Antonio Fernández-Caballero
María Gracia Manzano Arjona
Enrique Alonso González
Sergio Miguel Tomé (Eds.)

# Una Perspectiva de la Inteligencia Artificial en su 50 Aniversario

Campus Multidisciplinar en Percepción e Inteligencia, CMPI-2006
Albacete, España, 10-14 de Julio del 2006
Actas, Volumen II

Universidad de Castilla-La Mancha

Departamento de Sistemas Informáticos

Impreso en España. Printed in Spain.

ISBN 84-689-9560-6 (Obra completa)
ISBN 84-689-9562-2 (Volumen II)

Depósito Legal: AB-315-2006

Imprime: Gráficas Quintanilla. La Roda

Diseño de la cubierta: UGSC (Unidad de Gestión Sociocultural)

# Presentación

El 31 de Agosto 1955, *J. McCarthy* (Dartmouth College, New Hampshire), *M.L. Minsky* (Harvard University), *N. Rochester* (I.B.M. Corporation) y C.E. Shannon (Bell Telephone Laboratories) lanzaron una propuesta para reunir en el verano de 1956 a un grupo de investigadores que quisieran trabajar sobre la conjetura de que cada aspecto del aprendizaje y cada característica de la inteligencia podían ser tan precisamente descritos que se podían crear máquinas que las simularan. El encuentro, celebrado en 1956 y ahora conocido como la conferencia de Dartmouth, se llevó a cabo con tal éxito que el evento acuñó el término *Inteligencia Artificial* y con él una nueva área científica de conocimiento. En el año 2006 se cumplen cincuenta años de la Conferencia de Dartmouth. Pero a pesar del tiempo transcurrido, el problema de encontrar las minuciosas descripciones de las características del cerebro y de la mente que fue mencionado en la propuesta de 1955 sigue tan vigente hoy, como ayer, a pesar del variado abanico de ciencias que lo abordan y estudian.

Albacete (España) ha sido en la semana del 10 al 14 de Julio la sede del evento internacional más importante en lengua castellana con el *Campus Multidisciplinar en Percepción e Inteligencia, CMPI-2006*. El Campus Multidisciplinar en Percepción e Inteligencia 2006 es un evento internacional en el que investigadores de diversas áreas relacionadas con la Percepción y la Inteligencia se encontrarán del 10 al 14 de Julio en el Campus Universitario de Albacete con el ánimo de recuperar el espíritu entusiasta de aquellos primeros días de la Inteligencia Artificial. En nuestra intención está el objetivo de crear un ambiente heterogéneo formado por especialistas de diversas áreas, cómo la Inteligencia Artificial, la Neurobiología, la Psicología, la Filosofía, la Lingüística, la Lógica, la Computación,, con el fin de intercambiar los conocimientos básicos de las diferentes áreas y de poner en contacto investigadores de los diferentes campos. El facilitar la creación de colaboraciones e investigaciones multidisciplinares es un objetivo prioritario de la propuesta.

El *Congreso Multidisciplinar en Percepción e Inteligencia*, que ha dado lugar a esta publicación, se engloba como parte fundamental en el Campus Multidisciplinar sobre Percepción e Inteligencia. Este Congreso Multidisciplinar en Percepción e Inteligencia va dirigida a todas aquellas personas que tengan interés por conocer qué es la Percepción y qué es la Inteligencia, vistas ambas desde una perspectiva claramente multidisciplinar. El Congreso Multidisciplinar contará con la presencia de destacados especialistas del campo de la investigación. Todos ellos, así, y desde su propia experiencia, podrán proporcionar a los asistentes una visión muy clara del estado actual de las distintas ciencias que se ocupan de la Percepción y la Inteligencia. Estas charlas invitadas o tutoriales complementan a la perfección las ponencias que se impartirán por las mañanas durante la Escuela de Verano sobre Percepción e Inteligencia.

El Campus Multidisciplinar en Percepción e Inteligencia ha contado también con la *Escuela de Verano en Percepción e Inteligencia: 50 Aniversario de la Inteligencia Artificial*, que se ha ofertado en el seno de la XIX Edición de Cursos de Verano de la Universidad de Castilla-La Mancha, y que se ha correspondido con la XVI Escuela de Verano del Departamento de Sistemas Informáticos. Pensada fundamentalmente para los alumnos de la Universidad de Castilla-La Mancha del Campus de Albacete, la Escuela de Verano sobre Percepción e Inteligencia ha cubierto aspectos de gran interés para las carreras de Informática, Medicina, Humanidades y Magisterio. Las clases magistrales de la Escuela de Verano han estado a cargo de importantes y reconocidos investigadores a nivel internacional. Todos ellos, así, y desde su propia experiencia, han proporcionado a los asistentes una visión muy clara del estado actual de las distintas ciencias que se ocupan de la Percepción y la Inteligencia.

Fruto de las contribuciones más importantes del evento han nacido dos libros[1]. El primero de ellos, formado por dos volúmenes, denominado *Una Perspectiva de la Inteligencia Artificial en su 50 Aniversario. Actas del Campus Multidisciplinar en Percepción e Inteligencia, CMPI-2006*, es el que tiene en sus manos. Contiene las contribuciones de los congresistas expuestas oralmente o presentadas como pósters en el Congreso. El otro, llamado *50 Años de la Inteligencia Artificial. XVI Escuela de Verano de Informática*, recoge las ponencias invitadas de prestigiosos investigadores que han asistido al Campus Multidisciplinar en Percepción e Inteligencia.

Julio del 2006

Antonio Fernández-Caballero
María Gracia Manzano Arjona
Enrique Alonso González
Sergio Miguel Tomé
*Comité Organizador CMPI-2006*

[1] La publicación de estos libros ha sido financiada en parte por la Acción Complementaria del Ministerio de Educación y Ciencia TIN2005-25897-E y la Acción Especial de la Junta de Comunidades de Castilla-La Mancha AEB06-023.

# Organización

El congreso internacional *50 Años de la Inteligencia Artificial: Campus Multidisciplinar en Percepción e Inteligencia, CMPI-2006* ha sido organizado por el Departamento de Sistemas Informáticos (DSI) y el Instituto de Investigación en Informática de Albacete (I3A) de la Universidad de Castilla-La Mancha (UCLM) en cooperación con el Parque Científico y Tecnológico de Albacete (PCyTA) y el Excmo. Ayuntamiento de Albacete.

## Comité Organizador

| | |
|---|---|
| Antonio Fernández-Caballero | (Universidad de Castilla-La Mancha) |
| María Gracia Manzano Arjona | (Universidad de Salamanca) |
| Enrique Alonso González | (Universidad Autónoma de Madrid) |
| Sergio Miguel Tomé | (Universidad de Castilla-La Mancha) |

## Comité Local

| | |
|---|---|
| De la Ossa Jiménez, Luis | (Universidad de Castilla-La Mancha) |
| Díaz Delgado, Carmen | (Universidad de Castilla-La Mancha) |
| Fernández Graciani, Miguel Angel | (Universidad de Castilla-La Mancha) |
| Flores Gallego, María Julia | (Universidad de Castilla-La Mancha) |
| García Varea, Ismael | (Universidad de Castilla-La Mancha) |
| Gómez Quesada, Francisco J. | (Universidad de Castilla-La Mancha) |
| López Bonal, María Teresa | (Universidad de Castilla-La Mancha) |
| López Valles, José María | (Universidad de Castilla-La Mancha) |
| Mateo Cerdán, Juan Luis | (Universidad de Castilla-La Mancha) |
| Miranda Alonso, Tomás | (Universidad de Castilla-La Mancha) |
| Parreño Torres, Francisco | (Universidad de Castilla-La Mancha) |
| Ponce Sáez, Antonio | (Universidad de Castilla-La Mancha) |

## Comité de Programa

| | |
|---|---|
| Adán Oliver, Antonio | Dept. de Ingeniería Eléctrica, Electrónica y Automática - Universidad de Castilla-La Mancha en Ciudad Real (ES) |
| Alvarez Sánchez, José Ramón | Dept. de Inteligencia Artificial - Universidad Nacional de Educación a Distancia (ES) |
| Arce Michel, Javier | Sociedad Iberoamericana de Psicología, Bolivia - SIAPSI (BO) |
| Areces, Carlos | Langue et Dialogue - Laboratoire Lorrain de Recherche en Informatique et ses Applications (FR) |
| Armengol i Voltas, Eva | Institut d'Investigació en Intel.ligència Artificial - Centro Superior de Investigaciones Científicas (ES) |

| | |
|---|---|
| Armero San José, Julio | Dept. de Lógica, Historia y Filosofía de la Ciencia - Universidad Nacional de Educación a Distancia (ES) |
| Augusto, Juan Carlos | School of Computing and Mathematics - University of Ulster at Jordanstown (UK) |
| Barber Sanchís, Federico | Dept. de Sistemas Informáticos y Computación - Universitat Politècnica de Valencia (ES) |
| Barro Ameneiro, Senén | Dept. de Electrónica y Computación - Universidade de Santiago de Compostela (ES) |
| Bayro-Corrochano, Eduardo José | Computer Science - CINVESTAV, Guadalajara (MX) |
| Bel Enguix, Gemma | Research Group in Mathematical Linguistics - Universitat Rovira i Virgili (ES) |
| Blanco Mayor, Aquilino Carmelo | Dept. de Filosofía - Universidad de Castilla-La Mancha en Albacete (ES) |
| Botella Beviá, Federico | Dept. de Estadística, Matemática e Informática - Universidad Miguel Hernández de Elche (ES) |
| Bustos Guadaño, Eduardo de | Dept. de Lógica, Historia y Filosofía de la Ciencia - Universidad Nacional de Educación a Distancia (ES) |
| Caminos Benito, María Elena | Centro Regional de Investigaciones Biomédicas - Universidad de Castilla-La Mancha en Albacete (ES) |
| Casacuberta Nolla, Francisco | Dept. de Sistemes Informàtics i Computació - Universitat Politècnica de València (ES) |
| Casals Gelpí, Alicia | Dept. d'Enginyeria de Sistemes, Automàtica i Informàtica Industrial - Universitat Politècnica de Catalunya (ES) |
| Castejón Costa, Juan Luis | Dept. Sociologia II, Psicología, Comunicación y Didáctica - Universitat d'Alacant (ES) |
| Chinellato, Eris | Robotic Intelligence Laboratory - Universitat Jaume I (ES) |
| Cordón García, Oscar | Dept. de Ciencias de la Computación e Inteligencia Artificial - Universidad de Granada (ES) |
| Cortés, Ulises | Knowledge Engineering and Machine Learning Group - Universitat Politècnica de Catalunya (ES) |
| Cuadros-Vargas, Ernesto | Computer Science - Universidad Catolica San Pablo, Arequipa (PE) |
| Deco, Gustavo | Institució Catalana de Recerça i Estudis Avançats - Universitat Pompeu Fabra (ES) |
| Del Pobil, Angel Pasqual | Robotic Intelligence Laboratory - Universitat Jaume I (ES) |
| Delgado García, Ana | Dept. de Inteligencia Artificial - Universidad Nacional de Educación a Distancia (ES) |
| Díaz Delgado, Carmen | Centro Regional de Investigaciones Biomédicas - Universidad de Castilla-La Mancha en Albacete (ES) |
| Duro, Richard J. | Dept. de Computación - Universidade da Coruña (ES) |
| Faríñas del Cerro, Luis | Institut de Recherche en Informatique de Toulouse - Université Paul Sabatier (FR) |

| | |
|---|---|
| Feliú Batlle, Vicente | Dept. de Ingeniería Eléctrica, Electrónica y Automática - Universidad de Castilla-La Mancha en Ciudad Real (ES) |
| Fernández Graciani, Miguel Angel | Dept. de Sistemas Informáticos - Universidad de Castilla-La Mancha en Albacete (ES) |
| Fernández Moreno, Luis | Dept. de Lógica y Filosofía de la Ciencia - Universidad Complutense de Madrid (ES) |
| Fuentes Melero, Luis | Dept. de Psicología Básica y Metodología - Universidad de Murcia (ES) |
| Gámez Martín, José Antonio | Dept. de Sistemas Informáticos - Universidad de Castilla-La Mancha en Albacete (ES) |
| García Pupo, Mauro | Sociedad Cubana de Matemática y Computación - Universidad de Holguín (CU) |
| García Varea, Ismael | Dept. de Sistemas Informáticos - Universidad de Castilla-La Mancha en Albacete (ES) |
| Garijo, Francisco | Telefónica Investigación y Desarrollo - Telefónica (ES) |
| Geffner, Hector | Institució Catalana de Recerca i Estudis Avançats - Universitat Pompeu Fabra (ES) |
| Godó Lacasa, Lluis | Institut d'Investigació en Intel.ligència Artificial - Centro Superior de Investigaciones Científicas (ES) |
| González López, Pascual | Dept. de Sistemas Informáticos - Universidad de Castilla-La Mancha en Albacete (ES) |
| Graña Romay, Manuel | Dept. de Ciencias de la Computación e Inteligencia Artificial - Euskal Herriko Unibertsitatea / - Universidad del País Vasco (ES) |
| Hernández Orallo, José | Dept. de Sistemes Informàtics i Computació - Universitat Politècnica de València (ES) |
| Herrera Viedma, Enrique | Dept. de Ciencias de la Computación e Inteligencia Artificial - Universidad de Granada (ES) |
| Huertas Sánchez, M. Antonia | Informática y Multimedia - Universitat Oberta de Catalunya (ES) |
| Insausti Serrano, Ricardo | Centro Regional de Investigaciones Biomédicas - Universidad de Castilla-La Mancha en Albacete (ES) |
| Jansana, Ramón | Dept. de Lògica, Història i Filosofia de la Ciència - Universitat de Barcelona (ES) |
| Jiménez López, María Dolores | Research Group in Mathematical Linguistics - Universitat Rovira i Virgili (ES) |
| Juiz Gómez, José Manuel | Dept. de Ciencias Médicas - Universidad de Castilla-La Mancha en Albacete (ES) |
| Kemper Valverde, Nicolás | Laboratorio de Sistemas Inteligentes - Universidad Nacional Autónoma de México (MX) |
| Latorre Postigo, José Miguel | Dept. de Psicología - Universidad de Castilla-La Mancha en Albacete (ES) |
| Llinás Riascos, Rodolfo | Department of Physiology and Neuroscience - New York - University Medical Center (USA) |
| Llopis Borrás, Juan | Centro Regional de Investigaciones Biomédicas - Universidad de Castilla-La Mancha en Albacete (ES) |
| López Bonal, María Teresa | Dept. de Sistemas Informáticos - Universidad de Castilla-La Mancha en Albacete (ES) |

| | |
|---|---|
| López de Mántaras, Ramón | Institut d'Investigació en Intel.ligència Artificial - Centro Superior de Investigaciones Científicas (ES) |
| López-Poveda, Enrique A. | Instituto de Neurociencias de Castilla y León - Universidad de Salamanca (ES) |
| Luján Miras, Rafael | Centro Regional de Investigaciones Biomédicas - Universidad de Castilla-La Mancha en Albacete (ES) |
| Marin Morales, Roque | Dept. de Ingeniería de la Información y de las Comunicaciones - Universidad de Murcia (ES) |
| Martín-Vide, Carlos | Research Group in Mathematical Linguistics - Universitat Rovira i Virgili (ES) |
| Martínez Galán, Juan Ramón | Centro Regional de Investigaciones Biomédicas - Universidad de Castilla-La Mancha en Albacete (ES) |
| Martínez Tomás, Rafael | Dept. de Inteligencia Artificial - Universidad Nacional de Educación a Distancia (ES) |
| Mastriani, Mario | Misión SAOCOM - Comisión Nacional de Actividades Espaciales (AR) |
| Matellán Olivera, Vicente | Robotics Lab. - Universidad Rey Juan Carlos (ES) |
| Mira Mira, José | Dept. de Inteligencia Artificial - Universidad Nacional de Educación a Distancia (ES) |
| Miranda Alonso, Tomás | Dept. de Filosofía - Universidad de Castilla-La Mancha en Albacete (ES) |
| Moreno-Díaz, Roberto | Instituto - Universitario de Ciencias y Tecnologías Cibernéticas - Universidad de Las Palmas de Gran Canaria (ES) |
| Moreno García, Juan | Dept. de Tecnologías y Sistemas de Información - Universidad de Castilla-La Mancha en Toledo (ES) |
| Moreno Valverde, Ginés | Dept. de Sistemas Informáticos - Universidad de Castilla-La Mancha en Albacete (ES) |
| Nepomuceno Fernández, Ángel | Dept. de Filosofía y Lógica - Universidad de Sevilla (ES) |
| Ojeda Aciego, Manuel | Dept. Matemática Aplicada - Universidad de Málaga (ES) |
| Oliver, Nuria | Microsoft Research - Microsoft Corporation, Redmond (USA) |
| Pascual Fidalgo, Vicente | Dept. de Sistemas Informáticos - Universidad de Castilla-La Mancha en Albacete (ES) |
| Pavón Mestras, Juan | Dept. de Sistemas Informáticos y Programación - Universidad Complutense de Madrid (ES) |
| Penabad Vazquez, Jaime | Dept. de Matemáticas - Universidad de Castilla-La Mancha en Albacete (ES) |
| Pérez Jiménez, Mario de Jesús | Dept. de Ciencias de la Computación e Inteligencia Artificial - Universidad de Sevilla (ES) |
| Pérez Sedeño, Eulalia | Instituto de Filosofía - Centro Superior de Investigaciones Científicas (ES) |
| Ponce Sáez, Antonio | Dept. de Filosofía - Universidad de Castilla-La Mancha en Albacete (ES) |
| Ponte Fernández, Dolores | Dept. de Psicoloxía Social, Básica e Metodoloxía - Universidade de Santiago de Compostela (ES) |
| Prieto Espinosa, Alberto | Dept. de Arquitectura y Tecnología de Computadores - Universidad de Granada (ES) |

| | |
|---|---|
| Puelles López, Luis | Dept. de Anatomía Humana y Psicobiología - Universidad de Murcia (ES) |
| Puerta Callejón, José Miguel | Dept. de Sistemas Informáticos - Universidad de Castilla-La Mancha en Albacete (ES) |
| Rechea Alberola, Cristina | Dept. de Psicología - Universidad de Castilla-La Mancha en Albacete (ES) |
| Rodríguez Ladreda, Rosa María | Asociación Andaluza de Filosofía (ES) |
| Ruiz Shulcloper, José | Centro de Aplicaciones de Tecnologías de Avanzada - CENATAV, Ciudad Cuba (CU) |
| Sánchez-Andrés, Juan Vicente | Dept. de Fisiología - Universidad de La Laguna (ES) |
| Sánchez Calle, Ángel | Dept. de Informática, Estadística y Telemática - Universidad Rey Juan Carlos (ES) |
| Sánchez Cánovas, José | Dept. de Personalitat, Avaluació i Tractaments Psicològics - Universitat de València (ES) |
| Sánchez Vila, Eduardo | Dept. de Electrónica y Computación - Universidade de Santiago de Compostela (ES) |
| Sanfeliú Cortés, Alberto | Institut de Robòtica i Informàtica Industrial - Universitat Politècnica de Catalunya (ES) |
| Solar Fuentes, Mauricio | Dept. de Ingeniería Informática - Universidad de Santiago de Chile (CL) |
| Silva Mata, Francisco José | Centro de Aplicaciones de Tecnologías de Avanzada - CENATAV, Ciudad Cuba (CU) |
| Sossa Azuela, Juan Humberto | Centro de Investigación en Computación - Instituto Politécnico Nacional (MX) |
| Sucar Succar, Luis Enrique | Dept. de Computación - Instituto Tecnológico y de Estudios Superiores de Monterrey (MX) |
| Taboada Iglesias, María Jesús | Dept. de Electrónica e Computación - Universidade de Santiago de Compostela (ES) |
| Toro-Alfonso, José | Dept. de Psicología - Universidad de Puerto Rico |
| Torra, Vicenç | Institut d'Investigació en Intel-ligència Artificial - Centro Superior de Investigaciones Científicas (ES) |
| Trillas Ruiz, Enric | Dept. de Inteligencia Artificial - Universidad Politécnica de Madrid (ES) |
| Tudela Garmendia, Pío | Dept. de Psicología Experimental y Fisiología del Comportamiento - Universidad de Granada (ES) |
| Vega Reñón, Luis | Dept. de Lógica, Historia y Filosofía de la Ciencia - Universidad Nacional de Educación a Distancia (ES) |
| Verdegay Galdeano, José Luis | Dept. de Ciencias de la Computación e Inteligencia Artificial - Universidad de Granada (ES) |
| Vidal Ruiz, Enrique | Dept. de Sistemes Informàtics i Computació - Universitat Politècnica de València (ES) |

## Entidades Organizadoras

Universidad de Castilla-La Mancha
Parque Científico y Tecnológico de Albacete
Excmo. Ayuntamiento de Albacete

## Entidades Patrocinadoras

Ministerio de Educación y Ciencia
Junta de Comunidades de Castilla-La Mancha
(Consejería de Educación y Ciencia)
Caja Castilla-La Mancha
Telefónica
Fundación Campollano
Instituto de Investigación en Informática de Albacete
Departamento de Sistemas Informáticos, UCLM
Revista "Mente y Cerebro"
Centro Regional de Investigaciones Biomédicas, UCLM
Excma. Diputación de Albacete

## Entidades Colaboradoras

Asociación Española para la Inteligencia Artificial
Asociación Andaluza de Filosofía
Associació Catalana d'Intel·ligència Artificial
Asociación Cubana de Reconocimiento de Patrones
Fundación Española para la Ciencia y la Tecnología
Instituto de Filosofía, CSIC
Instituto de Neurociencias de Castilla y León
Mexican Association for Computer Vision, Neurocomputing and Robotics
Sociedad de Lógica, Metodología y Filosofía de la Ciencia en España
Sociedad Chilena de Ciencia de la Computación
Sociedad Colombiana de Psicología
Sociedad Cubana de Matemática y Computación
Sociedad Española de Filosofía Analítica
Sociedad Española de Neurociencia
Sociedad Española de Psicología Experimental
Sociedad Interamericana de Psicología
Sociedad Mexicana de Ciencia de la Computación
Sociedad Peruana de Computación
Sociedad Venezolana de Filosofía
TECNOCIENCIA

# Índice general

## VOLUMEN I

### FUNDAMENTOS DE LA INTELIGENCIA ARTIFICIAL Y REPRESENTACIÓN DEL CONOCIMIENTO



### ONTOLOGÍAS Y GESTIÓN DEL CONOCIMIENTO

**SISTEMAS EXPERTOS Y DE AYUDA A LA DECISIÓN**



**FILOSOFÍA Y MODELOS DE LA MENTE**



**REDES NEURONALES**

## COMPUTACIÓN EVOLUTIVA Y ALGORITMOS GENÉTICOS



## ROBÓTICA Y SISTEMAS AUTÓNOMOS



## SISTEMAS MULTIAGENTE Y ARQUITECTURAS PARA LA INTELIGENCIA ARTIFICIAL

## VOLUMEN II

### PERCEPCIÓN E INTELIGENCIA BIO-INSPIRADAS



### CLASIFICACIÓN Y RECONOCIMIENTO DE PATRONES

**RAZONAMIENTO FORMAL**



**RAZONAMIENTO APROXIMADO Y RAZONAMIENTO BAYESIANO**



**HEURÍSTICAS Y METAHEURÍSTICAS**

## INCERTIDUMBRE Y LÓGICA DIFUSA



## LENGUAJE NATURAL

## TRADUCCIÓN AUTOMÁTICA



## TECNOLOGÍAS DE INTERACCIÓN INTELIGENTES



## PLANIFICACIÓN Y OPTIMIZACIÓN

# Una arquitectura bioinspirada para el modelado computacional de los mecanismos de atención visual selectiva

José Mira[1], Ana E. Delgado[1], María T. López[2], Antonio Fernández-Caballero[2] y Miguel A. Fernández[2]

[1] Departamento de Inteligencia Artificial, E.T.S.I. Informática,
UNED, 28040 - Madrid, España
{jmira, adelgado}@dia.uned.es
[2] Departamento de Sistemas Informáticos, Escuela Politécnica Superior de Albacete e
Instituto de Investigación en Informática de Albacete (I3A),
Universidad de Castilla-La Mancha, 02071 – Albacete, España
{mlopez, caballer, miki}@info-ab.uclm.es

**Resumen.** En este trabajo se propone una arquitectura ascendente-descendente que integra una gran parte de las propuestas actuales sobre modelos computacionales de la tarea de atención visual selectiva. Se proponen después dos mecanismos básicos para hacer operacional esta arquitectura (la inhibición lateral y la computación acumulativa) y se mencionan algunos de los trabajos del grupo en visión activa en los que se ha aplicado esta arquitectura y los mecanismos mencionados. Se concluye reflexionando sobre la conveniencia de proponer arquitecturas y mecanismos sintéticos, tanto para comprender los mecanismos biológicos como para sintetizar sistemas de visión artificial.

## 1 Introducción

La búsqueda de una arquitectura de la cognición (la "lógica de la mente") es una tarea de largo recorrido y resultados muy limitados que tiene sus raíces en la Grecia antigua y que desde entonces ha preocupado esencialmente a filósofos, neurofisiólogos, psicólogos, matemáticos y, más recientemente, a profesionales del campo de la computación en general y de la Inteligencia Artificial (IA) en particular. Nos encontramos inquietos ante un conocimiento parcial, fragmentado y sin estructurar y buscamos una estructura abstracta que nos permita acomodar ordenadamente esas piezas de conocimiento. La arquitectura arrastra organización y estructura, facilidad de indexación y búsqueda y, finalmente, uso eficiente de ese conocimiento en la inferencia y en el razonamiento.

En el campo de la robótica se mantiene explícitamente el nombre de arquitectura y se usan términos tales como "reactiva", "situada" o representacional. En IA se le suele llamar paradigmas y se distingue entre conexionista, situado y simbólico o representacional. En este trabajo presentamos una arquitectura de carácter híbrido, que combina una parte ascendente de tipo conexionista con otra descendente de tipo sim-

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 425-437, 2006.

bólico. Para fijar ideas nos centramos en el camino visual y en la tarea de atención visual selectiva pero lanzamos también la conjetura de su potencial validez en otras modalidades sensoriales y en la comprensión y síntesis de otras tareas en las que se combinan componentes reactivas con otras de naturaleza intencional.

El resto del trabajo está estructurado de la siguiente forma. En la sección segunda resumimos la tarea y comentamos algunas de las aproximaciones relevantes a su modelado computacional. A continuación presentamos nuestra propuesta a nivel conceptual (sección tercera). Después describimos dos mecanismos básicos seleccionados para hacer operacional esta arquitectura: la inhibición lateral (sección 4ª), y la computación acumulativa (sección 5ª). Finalmente, en la sección 6ª mencionamos algunos de los trabajos del grupo basados en esta arquitectura y en los dos mecanismos de cálculo asociados.

## 2 La tarea de atención selectiva visual

Bajo el nombre de Atención Selectiva Visual (ASV) se engloban un conjunto de mecanismos de procesado de imágenes encaminados a enfocar la mirada y/o un conjunto de efectores sobre aquella o aquellas regiones de la imagen en las que ocurren sucesos espacio-temporales relevantes y de carácter local. La mayor o menor relevancia de un suceso local tiene que ver: (1) Con su carácter diferencial en relación con su entorno. (2) Con su analogía o diferencia con el conjunto de características que definen los objetos de un conjunto previamente definido a los que hay que reconocer (identificar). (3) Con una combinación de ambos criterios. Estos mecanismos de ASV sirven a dos propósitos: (1) Filtrar la información relevante en cada momento de otra que no lo es y (2) modular (realzar) la información seleccionada de acuerdo con un propósito. Es decir, ayudan a encontrar, mediante un proceso de búsqueda activa, la información relevante en cada momento para llevar a cabo la tarea de interacción con el medio en la que está involucrado el sistema.

Planteada así la tarea de ASV, tiene las características de una doble selección, estática y dinámica, basada en una combinación de criterios guiados por datos (criterios "de abajo hacia arriba" y de selección pasiva) y por conocimiento (criterios "de arriba hacia abajo" y de selección activa, por búsqueda) tal como se ilustra en la Figura 1. El rol de entrada es el conjunto de píxeles que componen la escena visual en cada momento y el rol de salida la constituyen las coordenadas de los ítems seleccionados. Es decir, las coordenadas de aquellas regiones de la imagen que sobresalen en ese momento ("dónde" mirar) por presentar valores máximos de actividad en alguna propiedad o combinación de propiedades o en la medida de la analogía del objeto detectado ("qué" mirar) con alguno del conjunto preseleccionado. Los roles estáticos los constituyen los criterios de selección pasiva y activa.

Esta descripción de la tarea de ASV es de carácter estacionario, ya que implícitamente se supone que la escena visual permanece constante mientras que se aplican los criterios de selección. Sin embargo, en muchas situaciones de interés la selección de coordenadas y/o ítems relevantes se realiza sobre el tiempo (conducción, blancos móviles, vigilancia, etc.) lo que nos obliga a comparar la situación de la escena visual en instantes sucesivos de tiempo viendo primero "cuándo" cambia en algún aspecto

relevante y viendo después "dónde" cambia y "qué" cambia. Para ello, los mecanismos de ASV dinámica necesitan disponer de una memoria de trabajo en la que almacenar la constancia o novedad de la evolución temporal de la escena.

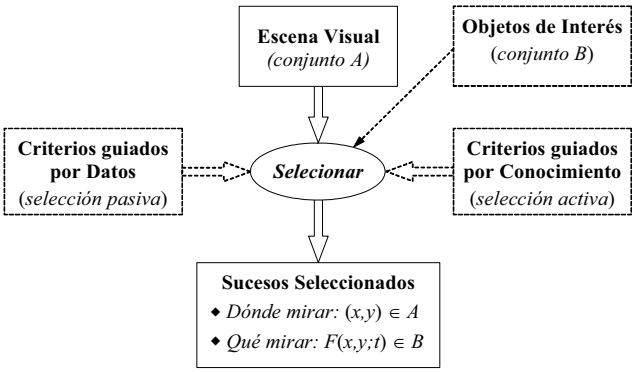



**Fig. 1.** Tarea de Atención Selectiva Visual

Si revisamos los trabajos previos sobre el modelado computacional de la ASV que pueden integrarse en esta visión de la atención como tarea de selección multicriterio, no es difícil detectar su agrupamiento en torno a alguno de los dos tipos de criterios mencionados: (1) Los modelos basados en la *escena*, asociados a los criterios guiados por datos (por señales) y (2) los modelos basados en la *tarea*, asociados a los criterios guiados por conocimiento.

El primer modelo guiado por datos fue propuesto por Koch y Ullman [1] y está basado en la conjetura de Anne Treisman [2] sobre la integración de características. En [3] se asocia este modelo ascendente con los datos biológicos sobre la anatomía y la fisiología del camino visual en su etapa más próxima al medio externo, antes de llegar a corteza, dando por supuesto que basta con una operación de selección de máximos sobre las respuestas de los circuitos neuronales encargados de extraer características [4]. La red usada (SLAM) tiene sin embargo una fuerte componente especulativa ya que no está claro que existan posiciones anatómicas concretas para representar palabras o colores, ni que la compatibilidad entre distintas visiones parciales de un objeto esté sólo asociada a procesos de excitación/inhibición. Otro modelo conexionista, SAIM [5], aunque de potencial validez en visión artificial, también es muy especulativo en su inspiración biológica. Supone que hay tres redes (de "contenidos", de "selección" y de "conocimiento") que interactúan como si fueran "agentes".

En cuanto a los modelos "descendentes", basados en criterios de selección asociados a la actividad interna del sistema (a sus objetivos), cabe destacar el modelo de búsqueda guiada propuesto por Wolfe [6] que usa la idea de "mapa de características relevantes" (Saliency Map) como punto de partida para cerrar el lazo y buscar los objetivos de interés usando el conocimiento del que se dispone sobre esos objetos. La atención se dirige así hacia los objetos que mejor se corresponden con las características que resaltan en cada intervalo temporal. El modelo de Barcker y Metsching [7], también abunda en esta necesidad de combinar criterios ascendentes (preatencionales o semiautomáticos) con otros criterios descendentes, intencionales, basados en la

búsqueda de objetos concretos. En todos los casos, de forma explícita o implícita, subyace la idea de "*lazo de control*" que enlaza la percepción con la acción. Se focaliza la atención (percepción) para ver qué hacemos y dónde y cuándo lo hacemos. Es decir, para manipular la escena (acción). De forma inseparable orientamos nuestros sensores (acción) para focalizar la atención dónde, cuándo y cómo la acción lo demanda (percepción).

## 3 Organización ascendente-descendente

Más allá del corto resumen del apartado anterior nos parece que hay evidencia experimental y consenso en los desarrollos teóricos sobre la existencia de una doble organización "ascendente" (de sensores a corteza) y "descendente" (desde corteza a efectores) en los mecanismos de ASV (Figura 2).

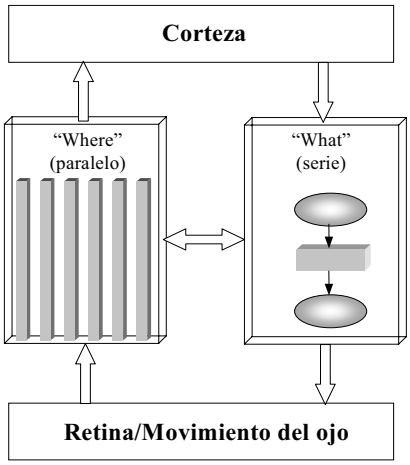



**Fig. 2.** Doble organización ascendente-descendente de la tarea de atención visual selectiva que integra las propuestas más relevantes del campo. Es crucial considerar la función de interacción entre ambas organizaciones, que existe prácticamente en todos los niveles corticales.

Esta doble organización de la ASV hace referencia al carácter espacial (estático) y/o temporal de la variable sobre la que se busca el máximo de actividad y a los dos criterios de selección: (1) por coordenadas y (2) por objetos que, lejos de excluirse, cooperan en función de los objetivos de la tarea global en la que se integran.

Las características que distinguen a la organización ascendente (figura 3) son las siguientes:

1. Es preatencional y automática (reactiva), aunque suele estar modulada por actividad voluntaria y por mecanismos reflejos dependientes de la novedad, conflicto, sorpresa e incertidumbre de los estímulos.
2. Computacionalmente tiene la arquitectura de un proceso paralelo sobre toda la escena visual, concurrente también con el resto de las modadlidades sensoriales.

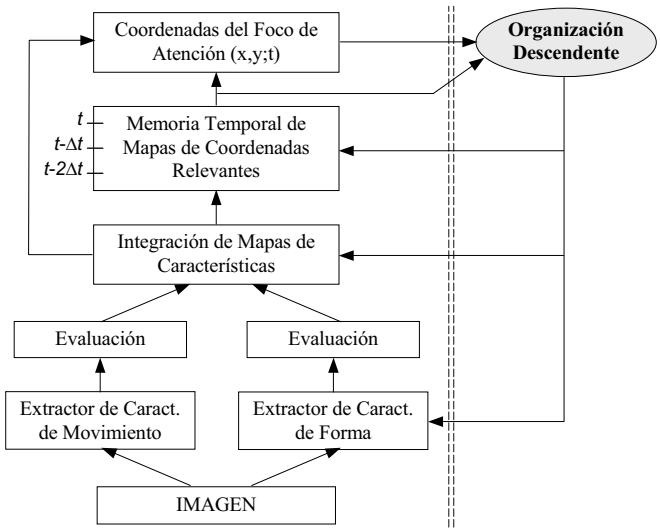



**Fig. 3.** Esquema conexionista de la organización ascendente.

3. Este proceso paralelo consiste en la extracción de un conjunto de características o propiedades de tipo local (intensidad, contraste en intensidad, color, orientación de las formas, velocidad, dirección y sentido del movimiento,...) en términos de las cuales construimos una representación de la escena.
4. Existen mecanismos transversales, de interacción lateral, que permiten integrar las distintas características para representar situaciones locales caracterizadas por determinadas configuraciones espacio-temporales entre los valores de dos o más características.
5. Estos extractores de características espacio-temporales de los estímulos deben de tener capacidad de adaptación ("aprendizaje") para permitir el ajuste de la forma y tamaño de las campos receptivos en función del factor de escala al que se quiere focalizar y de la representación en memoria de los objetos a cuya búsqueda contribuyen con la organización descendente.
6. En biología la mayor parte de estas características ("*esquemas perceptuales*") están "precalculadas" y se corresponden con aquellas propiedades del medio que se han mostrado relevantes para la supervivencia. Son los "canales" por los que el animal crea una representación abstracta, de dimensionalidad reducida, y por tanto eficiente para poder reaccionar en tiempo real.

En resumen, la *organización ascendente* de la percepción parece apropiada para crear una *representación interna del medio* y para su formulación analítica, lógica o inferencial en términos de "*circuitos neuronales*". En visión artificial, esta organización se corresponde con las tareas de *monitorización.*

En la figura 3 mostramos un posible esquema de organización ascendente que supone el cálculo en paralelo de características locales de tipo diferencial que detectan las coordenadas de los puntos en los que hay fuertes discontinuidades en el valor de cada propiedad y la combinación posterior de estos mapas individuales (integración

de características) en un único mapa de coordenadas relevantes en el que se selecciona la combinación de máximos individuales a la que hay que atender primero.

Conviene señalar que el carácter conexionista de esta organización nos exige que estas propiedades puedan calcularse de la misma forma sobre todas las coordenadas de la escena. Es decir, que las podamos formular como un operador de *convolución generalizada* en el que la forma y tamaño del núcleo especifican la función. Adicionalmente, pueden incluir también elementos no lineales o algorítmicos, pero también de tipo local e invariantes ante traslaciones.

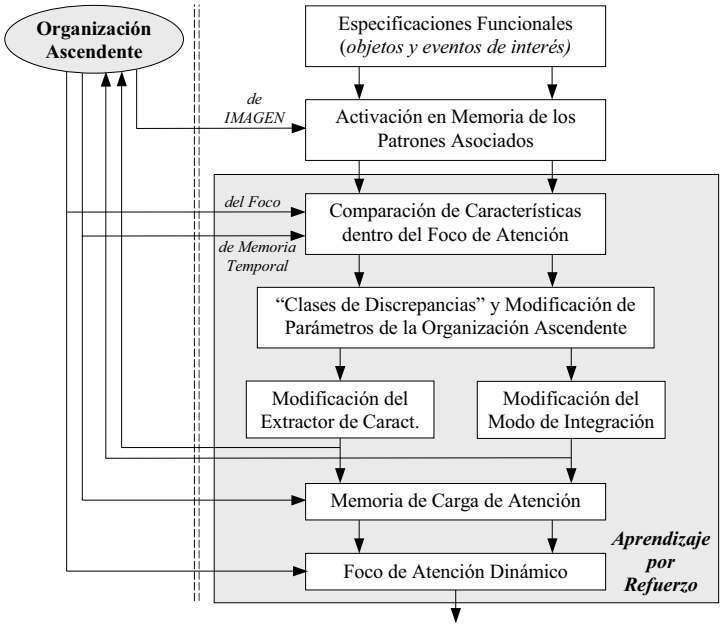



**Fig. 4.** Esquema de la organización descendente

La *organización descendente* de la ASV (la *búsqueda guiada*) está más próxima a la *visión simbólica* o representacional de la IA. En los sistemas biológicos se corresponde con el comportamiento intencional y consciente, guiado por propósitos, y en IA se corresponde con los llamados "*Sistemas Basados en Conocimiento*" (SBCs). Las características que distinguen esta organización descendente (figura 4) son las siguientes:

1. Tiene la estructura temporal de un proceso secuencial (serie).
2. Se acepta que el sistema conoce inicialmente, con distinto grado de precisión, la naturaleza y las características distintivas de los objetos y eventos en los que quiere centrar la atención. Conoce los "objetos y sucesos de interés" que quiere identificar.
3. El esquema inferencial de esta organización se corresponde con las tareas de selección activa e incluye inferencias de "*evaluación*", "*comparación*", "*selección*", "*diálogo*" y "*aprendizaje*" que conducen a reforzar o inhibir los focos de atención preseleccionados de forma cuasi-automática por la organización ascendente.

4. Es crucial considerar como parte de esta organización los procesos de aprendizaje por refuerzo que se encargan de comparar la descripción inicial de los objetos diana con la descripción de los objetos preseleccionados y que, como resultado de esa comparación, los etiqueta como "activadores", "inhibidores" o "neutros". El resultado de esta clasificación permite actualizar el contenido de la memoria de trabajo (la "respuesta" dinámica del SBC)
5. Hay, al menos, dos tipos de mecanismos biológicos que se han mostrado suficientes para implementar esta organización descendente: (1) La *inhibición lateral* (IL), ya mencionada en la ascendente, y la "*computación acumulativa*", que actúa como mecanismo de memoria dinámica que acumula la evolución temporal de una serie de sucesos del medio externo.
6. Esta organización descendente es la encargada de controlar la interacción entre las dos organizaciones y de refinar la selección del foco de atención a partir de las características específicas de un objeto-diana. Es decir, de una clase de combinaciones de propiedades.

En resumen la organización descendente tiene que ver con propósitos e intenciones y con el control en lazo cerrado de la actividad de los sensores y efectores encaminados a alcanzar esos objetivos. Aquí ya hablamos de "objetos", que son entidades de un grado de semántica superior a la de los vectores de propiedades de la organización ascendente (manchas, forma, color, velocidad,...). Se trata por tanto de un proceso de reconocimiento (identificación) de objetos que realza la combinación de propiedades que mejor se ajusta al propósito de la búsqueda. Este propósito se implementa biológicamente, por ejemplo, a través de la memoria del objeto buscado. En visión artificial, obviamente, somos nosotros quienes especificamos los "propósitos", las características de los objetos buscados y, a través de un mecanismo de aprendizaje por refuerzo, *refinamos* la selección inicial establecida por la organización ascendente ajustando los parámetros del filtrado inicial y eliminando de la "memoria de trabajo" los mapas que menos se parecen al patrón de búsqueda [8].

## 4 Inhibición lateral algorítmica (ALI)

Para hacer operacional una arquitectura hay que especificar los mecanismos y/o los procesos que subyacen a cada una de las subtareas e inferencias mencionadas en esa arquitectura. La conjetura de nuestro grupo es que toda la organización ascendente puede operacionalizarse usando un método inspirado en la biología al que llamamos "interacción lateral algorítmica" (ALI) por ser una generalización de los circuitos anatómicos de inhibición lateral [9].

Hay ciertas estructuras neuronales que se repiten en todos los niveles de integración del sistema nervioso y que vuelven a aparecer al nivel del comportamiento global de los seres vivos. Esto nos hace pensar que la evolución los ha sedimentado porque eran adaptivamente útiles en la interacción con el medio.

Si miramos estas estructuras desde una perspectiva electrónica y computacional podríamos decir que son los "módulos funcionales básicos" en términos de los cuales la evolución va diseñando la mejor arquitectura para que el Sistema Nervioso (SN)

"procese información" o, siguiendo a Maturana, "para que el sistema se acople estructuralmente a su medio". Este es el caso de los circuitos de IL, por la gran diversidad de funciones que pueden sintetizarse con ellos, por su capacidad de explicación en neurofisiología y por la facilidad de abstracción que aceptan al mantener invariante su estructura ante cambios de nivel en la semántica.

Nos encontramos con esquemas de IL en niveles tales como la neurogenesis, los contactos dendro-dendríticos, los circuitos neuronales en retina, cuerpo geniculado lateral y, en corteza cerebral, en la interacción entre grupos de neuronas (columnas) [10]. A nivel físico, en términos de circuitos interconectados que se describen usando un lenguaje de señales, hay dos esquemas de conectividad básicos: (1) IL no recurrente, y (2) IL recurrente, con realimentación (línea de trazos en la figura 5). En la IL no recurrente, la modificación de la respuesta de una unidad depende de las entradas a las unidades vecinas y en la recurrente depende de las salidas de las unidades vecinas.

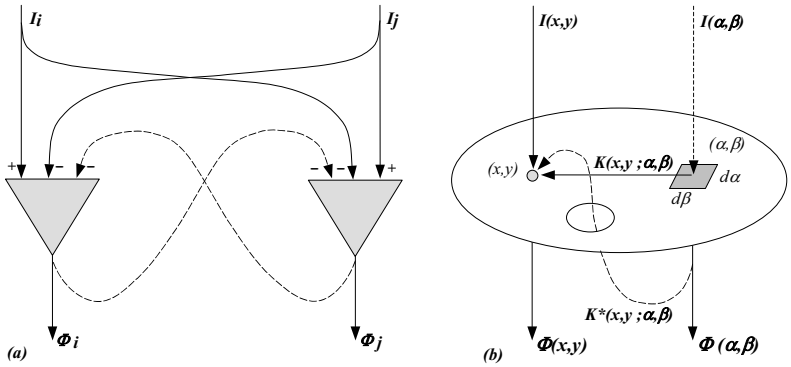



**Fig. 5.** Esquema de conectividad a nivel físico de la IL recurrente (trazo discontinuo) y no recurrente (trazo continuo). (a) Caso discreto. (b) Caso continuo.

Si extendemos la formulación al caso continuo (figura 5.b), los términos de interacción, se convierten en núcleos de una integral de convolución de forma que la salida, $\Phi(x,y)$, es el resultado de acumular la excitación directa de cada unidad, $I(x,y)$ con la inhibición procedente de modular la excitación recibida por las unidades vecinas conectadas, $I(\alpha,\beta)$, a través de los factores de peso $K(x,y;\alpha,\beta)$. El núcleo en diferencias, $K(x,y;\alpha,\beta)$, es ahora el responsable de la forma específica del cálculo que. en todos los casos, actúa como un detector de contrastes.

En la IL recurrente es la salida directa, $\Phi(x,y)$, la que se acumula a la inhibición procedente de las respuestas de las unidades vecinas, $\Phi(\alpha,\beta)$, ponderada por un coeficiente de interacción $K^*(x,y;\alpha,\beta)$, en principio diferente del de la vía directa (K).

De nuevo aquí, en IL recurrente, la forma y tamaño del campo receptivo (del núcleo de interacción K*) especifica la conectividad de la red (zona de cooperación-competición) y los detalles del cálculo (sintonía, orientaciones, formas, velocidades, etc.) La forma más usual en K y K* es una aproximación de la Laplaciana de una gaussiana, obtenida restando dos gussianas. Se obtienen así las estructuras de cálculo propias de toda red de IL: (1) Una zona central (ON ó OFF). (2) Una zona periférica (OFF ó ON). (3) Una región excluida (fuera del campo receptivo).

El modelo de IL en el nivel físico no puede ir más allá del lenguaje de señales. La primera abstracción posible, que nos permite pasar de un circuito a un algoritmo, se obtiene al reescribir los procesos de acumulación y de inhibición en términos de reglas (condicionales "if–then") como generalización de la suma ponderada y del umbral. El campo de condición de una regla generaliza la suma ponderada y el condicional generaliza la no linealidad del umbral. Por otro lado, la estructura del modelo de IL se mantiene. Es decir, ahora también definimos campos receptivos (campos de datos sobre una memoria FIFO espacio-temporal) con parte central excitadora y periferia inhibidora para el espacio de entradas (IL no recurrente) y para el espacio de salidas (IL recurrente), sólo que estos espacios son de representación y que la descripción de los cálculos realizados sobre los datos de la zona central y sobre los de la periferia la hacemos usando reglas en cuyo campo de condición pueden intervenir operadores lógico-relacionales distintos de la suma, resta y producto que eran los usados hasta ahora [9,10].

Con esta interpretación de la ALI, cada elemento de cálculo muestrea sus datos en la parte central y periférica del volumen que especifica su campo receptivo en el espacio de entradas y muestrea también en la parte central y periférica del volumen que especifica su campo receptivo en el espacio de salidas. Al especificar la naturaleza de las *reglas de decisión* obtenemos los distintos tipos de cálculo realizables por una red de IL algorítmica.

Finalmente, todavía podemos realizar un nuevo proceso de abstracción, pasando del nivel de los símbolos al nivel de conocimiento y generalizando las reglas en términos de inferencias [10,11]. Con este proceso de abstracción es posible considerar a los circuitos de IL como el soporte anatómico de un esquema inferencial (figura 6).

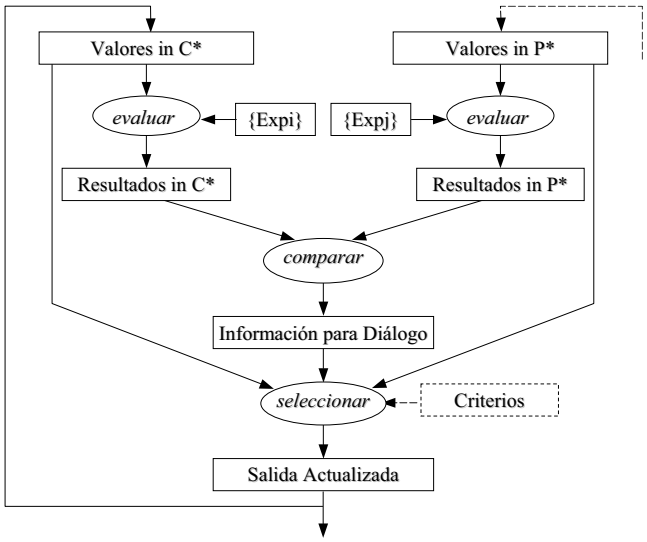



**Fig. 6.** Esquema inferencial de ALI recurrente. Se comparan los resultados de la evaluación en la zona central de realimentación (C*) y en la periférica (P*) y se selecciona dinámicamente la respuesta.

## 5 Computación acumulativa

Junto a los modelos neuronales convencionales en los que el tiempo entra en el cálculo a través de la simulación del retardo sináptico, dando lugar a sistemas de ecuaciones diferenciales de primer orden ("integra y dispara"), hay otras muchas formas en las que el SN incorpora el tiempo (carga y descarga del potencial de membrana, retardos analógicos, ciclos de oscilación en redes recurrentes, etc.). En nuestro grupo hace tiempo que nos fijamos en los procesos de computación en membranas y en la forma de acumular los potenciales presinápticos y de disparar la unidad cuando en el proceso dinámico de carga y descarga se alcanza un cierto valor umbral o cuando se satisfacen ciertas condiciones específicas en la historia reciente de los estímulos [12,13,14]. Al modelo resultante le llamamos "*computación acumulativa*" (CA). Independientemente de su origen biológico, la CA es un modelo de computación paralela y distribuida en el que la información local se transforma de acuerdo con un conjunto específico de reglas. Combinando la CA con el modelo de IL descrito en el apartado anterior, y encargado de la interacción ("diálogo") entre módulos CA vecinos, obtenemos un marco teórico y formal adecuado para ayudar a explicar el cálculo biológico y para resolver problemas en IA.

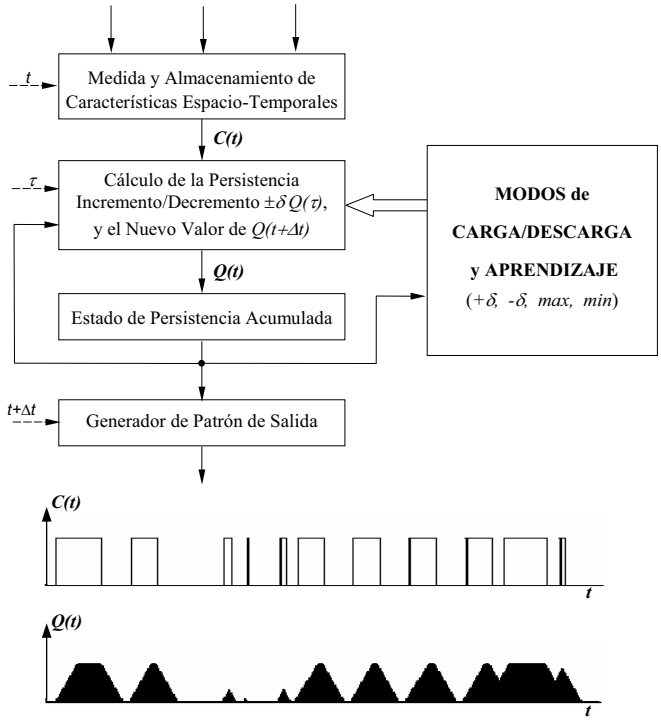



**Fig. 7.** (a) Modelo de Computación Acumulativa. (b) Ejemplo de evolución temporal del estímulo, C(t), y del estado de carga de una celda de CA, Q(t).

La figura 7.a muestra el diagrama de la CA que trabaja en dos escalas de tiempo, una macroscópica, t, asociada a la secuencia de datos externos que va a procesar la red y otra microscópica, interna, $\tau$, asociada al conjunto de procesos internos que tienen lugar mientras que los datos externos (una imagen, por ejemplo) permanecen constantes. El modelo consta de los siguientes elementos:

- Un módulo de extracción de características seguido de una memoria de trabajo que nos permite acceder al valor de las entradas en varios instantes sucesivos de tiempo.
- Un módulo que calcula el valor del incremento o decremento, ($\pm\delta Q$) del estado de actividad de esa propiedad en función de su valor en ese instante, Q(t), del valor acumulado en los instantes anteriores y del modo de acumulación.
- Un módulo de acumulación, tipo condensador (cálculo analógico) o contador reversible (cálculo digital), que almacenan el nuevo estado de persistencia de la característica seleccionada.
- Un módulo de programación, control y aprendizaje que controla la operación de cambio del estado de carga y ajusta los parámetros del módulo extractor de características espacio-temporales. La parte (b) de la figura 7 ilustra, para una dimensión, una posible evolución temporal del estado de carga en una celda de CA ante una secuencia de estímulos concreta.

## 6 Conclusiones

Posiblemente las arquitecturas de la cognición no puedan formularse de la forma estructurada, axiomática, deductiva, formal y precisa que requiere un modelo computacional. Por otro lado, al analizar con cierto rigor cualquier tarea biológica nos encontramos con que su complejidad es mucho mayor de lo esperado por la variedad y diversidad de mecanismos asociados y por la propia naturaleza de sus elementos constituyentes. Así pasa con el caso de la atención, que involucra procesos de memoria, alerta, vigilancia, atención selectiva y atención dividida, aprendizaje por refuerzo, intencionalidad y consciencia, entre otros.

Sin embargo, desde el punto de vista de la ciencia y la ingeniería quizás no tengamos otra alternativa que aceptar ambos retos: (1) Intentar dotar de una estructura sintética (una "arquitectura") a los procesos cognitivos, aún a sabiendas de que es insuficiente y (2) aceptar con humildad la pobreza de nuestras soluciones (en AVS en este caso), en comparación con las soluciones aportadas por la biología.

Y lo mismo ocurre en relación con los *mecanismos* de cálculo usados para hacer operacionales nuestras arquitecturas. En esta propuesta hemos explorado la posibilidad de que con sólo dos mecanismos sintéticos, la inhibición lateral y la computación acumulativa, sea suficiente para hacer computacional una parte importante de los procesos de información constituyentes de ambas organizaciones, la ascendente y la descendente. Algunos de los trabajos del grupo avalan la validez de esta aproximación metodológica.

En primer lugar, la CA [15] fue aplicada al problema de la clasificación de objetos móviles en secuencias indefinidas de imágenes [12], mostrándose incluso su capaci-

dad para ser implementada en tiempo real [16]. Más adelante la combinación de la CA y la IL se usó en la resolución del problema de la segmentación de siluetas móviles en secuencias de vídeo [17,18], describiéndose en extenso su naturaleza neuronal [19], el modelo de detección del movimiento [20] y la influencia de cada uno de los parámetros combinados de la CA con la IL [21]. A partir de los buenos resultados obtenidos mediante estos métodos en análisis del movimiento en visión artificial, el siguiente paso fue el de afrontar el reto de la atención visual selectiva (dinámica) por medio de una nueva línea de investigación abierta [8,22], en la que se ha mostrado, entre otras, la importancia de los parámetros de velocidad a la hora de mejorar la captura de la atención sobre los objetos en movimiento [23].

En paralelo con los trabajos anteriores, se vuelve a recurrir a la CA fundamentalmente, y a la IL en menor medida, para mejorar la segmentación de los objetos móviles, introduciendo la estereoscopía, con la finalidad de añadir un parámetro que puede resultar de una enorme importancia para cualquier tarea de seguimiento en el mundo tridimensional, a saber la profundidad [24]. Se han mostrado en la actualidad los primeros resultados aplicados a la potencial utilidad en robótica móvil [25]. Cabe señalar, por último, que el modelado de la IL a nivel de conocimiento, la denominada inhibición lateral algorítmica (ALI) [26] se ha mostrado como un método útil para resolver de forma modular, paralela y distribuida, problemas de cooperación y diálogo. En la actualidad estamos intentando construir físicamente, mediante lógica programable, un prototipo de esta arquitectura para aplicarla a problemas que demandan una solución en tiempo real.

## Agradecimientos

Este trabajo ha sido financiado en parte por los proyectos coordinados de investigación CICYT TIN2004-07661-C02-01 y TIN2004-07661-C02-02, integrados en el proyecto AVISA.

## Referencias

1. Koch, C., Ullman, S., “Shifts in selective visual attention: towards the underlying neural circuitry”, Human Neurobiology 4 (1985) 219-227.
2. Treisman, A.M., Gelade, G., “A feature-integration theory of attention”, Cognitive Psychology, vol. 12, (1980): 97-136.
3. Itti, L., Koch, C., Niebur, E., “A model of saliency-based visual attention for rapid scene analysis”, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (1998): 1254-1259.
4. Phaf, R.H., van der Heijden, A.H.C., Hudson, P.T.W., “SLAM: A connectionist model for attention in visual selection”, Cognitive Psychology 22 (3) (1990): 273-341.
5. Heinke, D., Humphreys, G.W., diVirgilo, G., “Modeling visual search experiments: Selective Attention for Identification Model (SAIM)”, Neurocomputing 44 (2002): 817-822.
6. Wolfe, J.M., “Guided Search 2.0. A revised model of visual search”, Psychonomic Bulletin & Review 1 (1994): 202-238.
7. Backer G., Mertsching, B. “Two selection stages provide efficient object-based attentional control for dynamic vision”. Proc. of the Int. Workshop on Attention and Performance in Computer Vision, WAPCV (2003) 9-16.
8. López, M.T., “Modelado computacional de los mecanismos de atención selectiva mediante redes

de interacción lateral". PhD Thesis, UNED, Madrid, 2003
9. Delgado, A.E., Mira, J., Moreno-Díaz, R., "A neurocybernetic model of modal co-operative decision in the Kilmer-McCulloch space", Kybernetes, vol. 18, no. 3, (1989): 48-57.
10. Mira, J., Delgado, A.E., "What can we compute with lateral inhibition circuits?", Lecture Notes in Computer Science, vol. 2084 (2001): 38-46.
11. Delgado, A.E., Mira, J., "Algorithmic lateral inhibition as a generic method for visual information processing with potential applications in robotics", Computational Intelligent Systems for Applied Research, World Scientific: Singapore, (2002): 477-484.
12. Fernandez, M.A. "Una arquitectura modular de inspiración biológica con capacidad de aprendizaje para el análisis de movimiento en secuencias de imagen en tiempo real". PhD Thesis, UNED, Madrid, España (1995).
13. Fernández, M.A., Mira, J., López, M.T., Alvarez, J.R., Manjarrés A., Barro, S., "Local Accumulation of Persistent Activity at Synaptic Level: Application to Motion Analysis". In From Natural to Artificial Neural Computation. J. Mira and F. Sandoval, (eds.) LNCS, 930. 137-143. Springer-Verlag, Berlin, (1995).
14. Mira, J., Fernández, M.A., López, M.T., Delgado, A.E., Fernández-Caballero, A., "A Model of Neural Inspiration for Local Accumulative Computation". In Computer Aided Systems Theory. R. Moreno-Díaz and F. Pichler (eds.) LNCS 2809: 427-435. Springer Verlag. Berlin (2003)
15. Fernandez, M.A., Mira, J., "Permanence memory: A system for real time motion analysis in image sequences". IAPR Workshop on Machine Vision Applications MVA'92, (1992): 249-252.
16. Fernández, M.A., Fernández-Caballero, A., López M.T., Mira, J., "Length-Speed Ratio (LSR) as a characteristic for moving elements real-time classification". Real-Time Imaging, 9 (1), (2003).49-59.
17. Fernández-Caballero, A., "Modelos de interacción lateral en computación acumulativa para la obtención de siluetas". PhD Thesis, UNED, Madrid, (2001).
18. Fernández-Caballero, A., Mira, J., Fernández, M.A., López, M.T., Segmentation from motion of non-rigid objects by neuronal lateral interaction. Pattern Recognition Letters, 22 (14), (2001): 1517-1524.
19. Fernández-Caballero, A., Mira, J., Fernández, M.A., Delgado, A.E., "On motion detection through a multi-layer neural network architecture". Neural Networks, 16 (2), (2003) 205-222.
20. Fernández-Caballero, A., Mira, J., Delgado, A.E., Fernández, M.A., "Lateral interaction in accumulative computation: A model for motion detection". Neurocomputing, 50C, (2003): 341-364.
21. Fernández-Caballero, A., Fernández, M.A., Mira, J., Delgado, A.E.,."Spatio-temporal shape building from image sequences using lateral interaction in accumulative computation". Pattern Recognition, 36(5), (2003):1131-1142.
22. López, M.T., Fernández-Caballero, A., Mira, J., Delgado, A.E., Fernández M.A., "Algorithmic lateral inhibition method in dynamic and selective visual attention task: Application to moving objects detection and labelling". To appear, Expert Systems with Applications (2006).
23. López, M.T., Fernández-Caballero, A., Fernández, M.A., Mira, J., Delgado, A.E., "Motion features to enhance scene segmentation in active visual attention". Pattern Recognition Letters, 27(5), (2006): 469-478.
24. López-Valles, J.M., "Estereopsis y movimiento. Modelo de disparidad de carga: Un enfoque con inspiración biológica". PhD Thesis, Univ. de Castilla-La Mancha, Albacete, (2004).
25. López-Valles, J.M., Fernández, M.A., Fernández-Caballero, A., López, M.T., Mira J., Delgado, A.E., "Motion-based stereovision model with potential utility in robot navigation". Proc. of the IEA/AIE 2005, LNAI, 3533, 16-25. Springer (2005).
26. Mira J., Delgado, A.E., Fernández-Caballero, A., Fernández, M.A., "Knowledge modelling for the motion detection task: The algorithmic lateral inhibition method". Expert Systems with Applications, 27 (2), 169-185. (2004).

# Interacción con seres simulados: Nuevas herramientas en psicología experimental

Carlos González Tardón

Universitat de Barcelona, Ramón Rocafull 94 bis 1º 1ª, 08033, Barcelona, España
carlos@carlosgonzaleztardon.com
http://www.carlosgonzaleztardon

**Resumen.** ¿Por qué usar seres simulados para estudiar la interacción? Esta es la pregunta que trataré de responder a lo largo del artículo. Partiendo de un repaso histórico de la relación de la psicología y psiquiatría con estos seres llegaremos hasta tres investigaciones que he realizado utilizándolo. Finalizaré el artículo con un repaso de las ventajas e inconvenientes que tiene la simulación para el estudio experimental en psicología.

## 1 Introducción

En este artículo pretendo demostrar que el uso de seres simulados para el estudio de la interacción humana es un método válido que posee una serie de ventajas difíciles de conseguir de otra manera.

El artículo comenzará con un marco histórico sobre la relación de la psicología y la psiquiatría con los seres simulados y virtuales. A continuación presentaré algunas de las investigaciones que he realizado respecto a este tema en los últimos años. Concluiré explicando las ventajas e inconvenientes que tiene el uso de los sujetos simulados para la psicología.

En el apéndice encontrarán un glosario, de recomendable lectura antes de comenzar el desarrollo del artículo, con algunos términos que aparecen y tienen más de un significado o uso pero que se utilizaran de forma muy concreta para este escrito, con el fin de simplificar las explicaciones.

## 2 Sujetos simulados, seres virtuales, psicología y psiquiatría

La relación de los sujetos simulados con la psicología se remonta al principio de la inteligencia artificial. En 1966 Joseph Weizenbaun, ingeniero del MIT, creó *ELIZA,* que era un "chatbot", un robot de conversación, que se comunicaba de forma verbal, por medio de frases en la pantalla, y entendía, o al menos eso intentaba, lo que se le escribía, contestando acorde a la conversación.

Fue concebido como un ejercicio para probar los límites de la capacidad de conversación que podía tener un ordenador. Consistía en una estructura de interacción

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 438-449, 2006.

verbal reactiva, el programa ELIZA, y una serie de personalidades o script que eran, en palabras de Weizenbaum, "[el conjunto de] Palabras claves y sus transformaciones asociadas, […] para un particular tipo de comunicación" [1]. El más conocido era el denominado *"Doctor",* que simulaba ser un psicoterapeuta rogeriano.

Este movimiento tiene su origen en Carl Roger que fue el fundador de la corriente humanística dentro de la psicología. Su técnica de intervención psicológica está orientada hacía la autocuración del paciente por medio del uso de preguntas reflejas del terapeuta. El terapeuta sólo debe intentar que el paciente hable y de esa forma irán saliendo los temas que le preocupan, y al irlos desarrollando los irá solucionando.

Utilizó este script porque pensaba que no era necesario tener conocimientos del mundo real, al ser el script totalmente reactivo, y por lo tanto, más fácil de programar.

Cuál fue su sorpresa cuando encontró que las personas rápidamente identificaban el programa como si fuera de verdad humano. Al interaccionar se acababan olvidando de que se trataba de un ser simulado sin ninguna inteligencia y comenzaban a contar sus secretos, miedos y problemas personales al igual que harían con un psiquiatra de verdad. Es decir, se produjo la inmersión al aceptar el simulador como un ser real e interactuar con él de forma "normalizada" o natural. Según Turkle "Cantidades muy pequeñas de interactividad nos provocan proyectar nuestra propia complejidad en un objeto que no lo merece" [2].

Esto le horrorizó porque para él era sólo un ejercicio de informática y consideraba que no tenía validez como terapeuta. El Doctor no entendía absolutamente nada de lo que se le decía y tal vez pudiera causar un problema psicológico serio a alguna persona.

Renegó de él y se convirtió en ferviente detractor de la inteligencia artificial llegando a escribir varios libros en contra de su uso.

Sin embargo, ese no fue el final del "Doctor". Kenneth Colby, un psiquiatra de la universidad de Stanford, lo percibió como un avance en psicoterapia y al poco tiempo presentó su programa *SHRINK*, un simulador de un psicoterapeuta real. Instaba a las personas a que pensaran que estaba interactuando con él, es decir, que vieran a Shrink como una extensión del psiquiatra Kenneth Colby dentro del ordenador.

En 1972 creó a *PARRY*, otro chatbot pero de características muy distintas al primero. Si Doctor y Shrink eran terapeutas, Parry era un simulador de un esquizofrénico paranoide. Estaba basado en las investigaciones que realizaba Colby en ese momento. Este programa, a parte de palabras claves, tenía estados internos como eran "furia", "miedo" y "desconfianza", cuyos niveles iban variando según lo que ocurriera en las interacciones y condicionaban las posibles respuestas siguientes. Hubo multitud de ocasiones que pusieron a interactuar a Parry con Doctor y se obtuvieron conversaciones de lo más rocambolescas.

En 1992, Colby presentó *Depression 2.0.* Fue recibido con gran expectación por la sociedad americana. Estaba especializado en el tratamiento de la depresión desde un marco cognitivo-conductual. Turkle lo probó con un grupo de personas y llegó a las siguientes conclusiones: "durante la mayor parte del tiempo dejan en suspenso la incredulidad y quedan absortos en lo que ocurre en la pantallas. En otras palabras, la cultura emergente de la simulación" [2] y "el éxito de las interacciones de Roger [uno de los sujetos] con el programa dependía de la tolerancia de sus limitaciones" [2]. Por lo tanto, según Turkle, el funcionamiento de los chatbot depende de la inmer-

sión y a su vez esta dependía de la predisposición del humano que interactuaba con él.

Todos estos chatbot pretendían ser un reflejo de un humano, simulando la parte verbal de las personas, ya fuera actuando como terapeuta o paciente. Como se interactuaba con ellos de forma escrita, nos encontramos con una situación en la que el ordenador sería un ser completo que interactúa con un humano. Por lo tanto, la persona que interactúa no lo hace inmerso en un mundo simulado, sino desde el mundo real. Es una interacción social humano-ser simulado.

Las personas también pueden estar inmersas en mundos simulados. A continuación prestaremos atención a aquéllos que están poblados por otros seres virtuales controlados por humanos. Veremos alguna de las características esenciales de esos lugares y de las personas que los habitan, por tanto estaremos describiendo las interacciones sociales mediadas por seres virtuales.

Los principales programas donde se realizan este tipo de interacción son los llamados *MMORPG* (Massive Multiplayer Online Role Playing Game, Juegos de rol con gran cantidad de jugadores en red) que nacieron en Korea en 1995 con la publicación de *"The Kingdom of the Wind"*. A principios del 2005 se calculaba que había unos 10 millones de personas que jugaban de manera habitual. En ellos cada humano elige su avatar, su representación en el mundo simulado, y comienza a interactuar con otros avatares (sujetos virtuales), tanto de forma comportamental como verbal, ya que se pueden enviar mensajes y hacer llamamientos a los personajes cercanos. También existen seres simulados, pero estos habitualmente hacen un papel secundario: los monstruos a los que matar o elementos del decorado. Su ambientación principalmente esta basada en la época medieval con un alto contenido fantástico. Los participantes toman estos juegos como una vía de escape o una forma de experimentación social. Como dice Turkle "los juegos son los laboratorios para la construcción de la identidad" [2] y "El mundo de la simulación es el nuevo escenario para jugar con nuestras fantasías, tanto emocionales como intelectuales" [3]. Ciertamente, esto es común a todos los juegos pero la peculiaridad del videojuego es que no son sólo los niños y adolescentes los que lo utilizan para construir su personalidad, sino que también lo hacen los adultos. Actualmente el mundo adulto tiene socialmente vedado los juegos clásicos como jugar a las familias, los de imaginación…, por estar catalogados para niños, por ello ha encontrado el sucedáneo en los videojuegos porque están socialmente aceptados.

Hay un nutrido grupo de artículos cuyo objetivo es descubrir cuáles son las características psicológicas de los jugadores, cómo actúan dentro del juego, cuál es el volumen de horas semanales que dedican, qué es lo que sacrifican para poder jugar y un largo etcétera [4][5][6]. Según Griffith y colaboradores "Con la llegada de los nuevos […] mundos virtuales online, existe la oportunidad, tanto de explorar la psicología de los jugadores que encaja con esta nueva forma de ocio, como también la psicología de los jugadores dentro del juego mismo" [7]. Es decir, no se trata de saber qué tipo de persona juega a los MMORPG sino también como son dentro de ese mundo. En palabras de uno de los participantes de otra investigación de Turkle respecto a este tema: "Los MUD [Multi User Domain, el precursor de los MMORPG] me hacen ser en mayor medida lo que realmente soy. Fuera del MUD, soy en menor medida yo mismo" [2].Tal vez los ambientes simulados sean el nuevo lugar de encuentro social,

por encima incluso de los Chat (salas de conversación online), ya que las personas que juegan a los MMORPG dicen fiarse más de la gente que en los chat porque ven "como actúan" no sólo "cómo hablan". Es decir, observan el comportamiento del ser virtual para conocer la personalidad del humano que hay detrás.

En los artículos consultados sobre el tema, hay consenso en que un alto porcentaje de jugadores los usan para hacer nuevos amigos e incluso buscar pareja. Su máximo interés se centra en el contenido social de estos programas.

¿Qué es lo que ocurre cuando la red social consiste en el jugador y todos los demás son sujetos simulados, es decir, cuando la interacción social es sólo con seres controlados por el ordenador?

Los primeros pasos para los simuladores de redes sociales se dieron a través del Commodore 64, a mediados de los ochenta, con el juego *Little Computer People,*. Escrito por David Crane, se presentaba al jugador como una investigación del comportamiento de la "pequeña persona que habitaba dentro del ordenador". Debíamos averiguar cuáles eran sus preferencias y necesidades, suministrarle agua y alimentos para que no enfermara e interactuar con él a través de sencillas instrucciones escritas para que no se sintiera solo. Básicamente era una pequeña mascota humana como su nombre original indicaba *"Human Pet".* El interés de este programa es ser el precursor de los videojuegos sobre "lo doméstico", es decir, la simplificación de la vida real.

En 1999 aparece *Babyz,* programado por Andrew Stern y Michael Mateas. En él adoptas un bebé simulado y tu tarea es enseñarle y criarle. Anteriormente ya habían creado *Dogz*, *Catz*… pero la acogida de Babyz fue distinta. Aunque la tecnología y la estructura del programa era la misma, las personas se comportaban de otra forma porque era un "bebé", no un "perro".

Los autores reportan el caso de una pareja que estaba utilizando al bebé simulado para saber si serían buenos padres en el futuro y otro caso de una pareja que les preguntó, al final de una conferencia, si su hijo simulado era "retrasado" porque les parecía que aprendía muy despacio [8]. Saquen ustedes conclusiones del nivel de "inmersión" que se puede llegar a conseguir con un programa de este tipo.

*Los Sims*, creado por Will Wright y publicado en el 2000, ha sido número uno en ventas desde su lanzamiento. Es el juego más vendido de todos los tiempos y tiene un porcentaje de jugadoras del 60%. Está considerado un fenómeno social.

Consiste en crear un grupo de "Sims", personajes que están en el ambiente simulado, y crear un hogar para ellos. Después es necesario mantener tanto la casa como a los propios seres, mandándoles que limpien, vayan a trabajar o al cuarto de baño. Pero la parte más importante de las actividades son las sociales. Tus Sims no están solos: existe una compleja red social formada por otros Sims que no están controlados por ti, cuya función es ser tus amigos, pareja o enemigo, todo depende de tu comportamiento hacia ellos. Una interacción seres simulados-ser virtual.

Pero, ¿dónde está el interés de conseguir amigos simulados que ni hablan ni sienten? Turkle argumenta que "Si a alguien le asusta la intimidad y a la vez le asusta estar solo, hasta un ordenador aislado (no en red) parece ofrecer una solución. Interactivo y reactivo, el ordenador ofrecerá la ilusión de compañerismo sin la demanda de una amistad" [2] y, por lo tanto, no debe entenderse como un juego, sino como una forma de pertenecer a una red social, pero sin las presiones sociales que existen en el

mundo real. Son ”Mundos autocontingentes en los que las cosas son más simples que en la vida real, y donde, si todo lo demás falla, puedes retirar tu personaje y empezar simplemente una nueva vida con otro” [2].

Además de ser el programa preferido de los jugadores, se ha convertido en el juego de la comunidad científica, como afirma Miguel Sicart [9]. Existen multitud de artículos sobre él, desde la ideología implícita [9] [8] hasta el estudio de los espacios domésticos que pueden ser creados [10] y de reseñas en periódicos y revistas, alguno realmente curioso como el escrito por Mark Boal [11]

La revista Psychology Today le dedicó un amplio reportaje escrito por Clive Thompson [12] donde desvela la base del modelo de toma de decisiones que tienen los seres simulados. Por una parte está basado en la pirámide de necesidades de Abraham Maslow (que curiosamente es el cofundador del movimiento humanista con Carl Roger). Es el sistema de prioridades que debe seguir el agente a la hora de satisfacer sus apetencias. Por otro lado, utiliza la teoría de la elección cuasi-económica desarrollado por David Friedman. Con este método elige las acciones que hará contrabalanceando los costos y beneficios posibles. También nos presenta el caso de un niño adoptado de origen rumano que explicó cómo había sido su traumática vida antes de llegar a la familia americana, simulando tanto su familia biológica como el orfanato donde vivió. Trabajó su experiencia traumática a través del juego.

Como dice Gonzalo Frasca “Yo creo que puedes aprender más de la naturaleza del hombre a través de un juego que de todas las película de Ingmar Bergman” [8]. Podemos considerar Los Sims como el gran “laboratorio social” de principios del siglo XXI.

Para finalizar utilizaremos el programa *Façade,* publicado en 2005, el “Drama interactivo sobre relaciones humanas” [13], como lo describen sus programadores, Mateas y Stern, creadores también de Babyz. Desde el punto de vista de primera persona, narra la interacción del jugador con una pareja de seres simulados que están pasando por un “bache” en su matrimonio.

Comienza cuando llegas a su casa y oyes que están discutiendo. Al llamar a la puerta dejan de discutir y te abren con una sonrisa… El objetivo del juego es intentar “intervenir” en la situación, ya sea para mejorar la situación o empeorarla. Como escribe Bosco y Caldana en un artículo de *El País* “Obliga a replantearse cuestiones fundamentales sobre las relaciones sentimentales, la vida en pareja y las nuevas opciones de núcleos familiares alternativos. Al vivir los hechos en primera persona, el jugador es consciente de que el desenlace depende de él” [14].

Tienes total libertad para realizar todo tipo de conductas como moverte por la habitación, abrazar, coger un vaso… pero es diferente a los anteriores porque además incorpora toda la parte verbal. Puedes interactuar con los personajes “hablándoles” (tecleando frases). En el fondo es una simulación de una terapia de pareja pero hecha juego. Al incorporar la parte verbal obtiene un realismo cercano a la práctica clínica y podría fácilmente ser aplicada al entrenamiento de psicólogos especializados en intervención familiar y de pareja.

Es muy similar a un programa de finales de los años ochenta, “The Erving programs” [15], una herramienta de enseñanza que pretendía que los alumnos predijeran cómo se sentirían y qué harían dos seres simulados, un hombre y una mujer, a los que observaban interactuar.

Lo más interesante de este programa es que simula una situación que está muy cargada emocionalmente. Además, al poder interactuar por medio del teclado y del movimiento, crea una sensación de inmersión bastante peculiar que te lleva hacía una sensación de "incomodidad" ante la situación, algo "embarazosa", en la que te encuentras. Es decir, te provoca cierto malestar el problema que existe entre "ellos".

Con este último programa, que es una síntesis de chatbot, MMORPG e interacción con seres simulados, espero haber conseguido dar una idea general de cuál ha sido y puede ser la relación de la psicología con los seres virtuales y simulados.

## 3 Ejemplos de investigación

Con la presentación de estas dos investigaciones y el proyecto de una tercera no pretendo dar resultados ni conclusiones de los temas tratados, sino proponer tres ejemplos del posible uso de los seres simulados como herramienta de investigación en psicología, en distintos campos y con distintas finalidades. Esta parte del artículo servirá de preámbulo al desarrollo de las ventajas e inconvenientes de su uso.

### 3.1 Credibilidad de los ChatBots: "Hola Soy su Ordenador, ya Puede Hablar…"

Esta investigación estaba centrada en si actualmente mantenemos o no comunicación con nuestros ordenadores, a partir del análisis de los componentes necesarios para considerar una situación como un acto comunicativo.
Se realizó un experimento con 10 sujetos a los que se les pedía que interactuasen con un ChatBot, el programa *Dr. ABUSE 6.10*, de Juan José Boronat y Vicente Barrés. Posteriormente se pedía a los participantes que valorasen el nivel de comunicación que hubo entre ellos.

En una segunda etapa observaban la trascripción de su interacción y valoraban como "acertada", "fracasada", "repetición" y "muy acorde con la conversación" las diferentes transiciones comunicativas entre ambos, con el objetivo de observar el grado de coherencia comunicativa asignado por el sujeto al Dr. Abuse

Lo más relevante de esta investigación fue que los sujetos afirmaban, al finalizar la interacción con el programa, que el nivel de comunicación era muy bajo. Sin embargo, una vez transcrito y haciéndolo de transición, en transición valoraban de una forma bastante alta la coherencia del programa y se sorprendían del tipo de comunicación que habían entablado con el programa hacía el final de la interacción.

### 3.2 Creación de situaciones imposibles: "*¿Por qué no Soy Bobby Fischer?*"

El tema principal era, como habrán imaginado, el ajedrez. El objetivo era crear un método de evaluación del juego para poder observar la diferencia entre un jugador amateur (sujeto experimental) y un gran maestro en partidas contra los mismos sujetos y así crear un método de entrenamiento personalizado para mejorar el juego del sujeto experimental.

Ante la imposibilidad de contar con un gran maestro, y más aún de Bobby Fischer, y la necesidad de que ambos jugadores hicieran sus partidas en las condiciones más similares posibles, se recurrió al programa de Ajedrez *ChessMaster 10000.*

Se utilizó este programa para:

1) Conseguir rivales idénticos en las partidas del gran maestro y el sujeto experimental, al tener predefinidos seres simulados de distintos niveles.
2) Poder contar con un gran maestro, y poder repetir las partidas las veces que fueran necesarias.
3) La posibilidad de grabar las partidas automáticamente y posteriormente ser visualizadas de una forma sencilla para poder categorizar los movimientos.

Se comparó la forma de juego de ambos sujetos y, a partir de las diferencias encontradas, se creó un programa de entrenamiento personalizado.

El interés de esta investigación respecto al artículo se concentra en el hecho de que habría sido difícil haber conseguido la colaboración de un gran maestro para hacer el experimento, así como unos rivales que se comportaran de la misma forma ante un profesional que ante un amateur, además, categorizar los movimientos habría sido más complicado y costoso ya que se tendrían que haber utilizado técnicas de grabación.

### 3.3 Interacción social y simulación de personas reales

Es el proyecto de investigación que actualmente estoy preparando, pretendo estudiar el comportamiento interactivo humano en un ambiente social por medio de un ser simulado. Para ello he programado un sencillo modelo de toma de decisiones con una serie de variables relevantes que serán obtenidas a través de un test a una persona. Con ello obtendré un sujeto simulado basado en datos reales de un ser humano concreto. La idea del modelo es similar a PARRY, ya que tiene una variable interna denominada “intimidad” que varía según el comportamiento en la interacción y modifica el universo de acciones que puede realizar el simulador en un momento concreto.

Está basado sólo en conductas, es decir, no se podrá interactuar con él de forma verbal por medio del teclado. No obstante, las conductas que se pueden realizar son tanto físicas como verbales. Por ejemplo, puedes marcar “abrazar” o “insultar”, abrazar es una conducta física mientras que insultar es verbal.

Una vez finalizado el modelo se pedirá a un grupo de participantes que interactúe con él y se buscarán patrones de comportamiento en la secuencia interactiva.

## 4 Conclusión: Ventajas e inconvenientes del uso de sujetos simulados en investigación psicológica de peatones

Sobre el uso de la simulación como método científico ha habido mucha controversia. Durante los años ochenta en el MIT hubo un gran debate en la Facultad de Física sobre la conveniencia de su uso. Mientras sus defensores esgrimían la ventaja de poder jugar con las variables y observar los cambios a tiempo real, a los detractores les parecía incoherente simular aquello que se podía medir de forma directa, llegando

a decir frases como la siguiente “La idea de hacer simulación… pido disculpas, pero es como la masturbación” [3].

El principal problema de la simulación, ya sean seres simulados o no, es la *validez ecológica* de los datos. Es decir, si los datos obtenidos son producto de estar utilizando la simulación o se puede considerar generalizable a otros contextos. Es muy difícil creer que una persona se comporta de igual forma interaccionando con un ordenador que haciéndolo en un contexto real con otro ser humano. Por ello, los datos que se recogen a través de la simulación siempre están en tela de juicio.

Realmente este inconveniente es casi infranqueable, el talón de Aquiles de la investigación con simuladores, pero la *inmersión* puede ser la manera de conseguir tener cierta validez ecológica con un ser simulado. Si el participante queda absorto por la realidad simulada en ese momento deja de ser consciente de que está interactuando con un ordenador y su comportamiento ya no está influido por ello. Por lo tanto se puede suponer que los datos son validos, son una medida real de su comportamiento. El problema reside en que no se ha operativizado el término “inmersión”, por lo menos hasta donde yo conozco, y no se puede saber con certeza en qué momento el sujeto está o no está absorto en la realidad simulada.

Por otro lado está la *aceptabilidad social* de los ambientes simulados que puede hacer aumentar la validez ecológica de los datos. Cuanto más familiar sea la situación experimental más natural será el comportamiento del sujeto. En el libro *Jóvenes y videojuegos*,[16], una investigación de la FAD (Fundación de Ayuda contra la Drogadicción) realizada a 3000 jóvenes de entre 14 y 18 años de todo el territorio nacional, encontramos los siguientes datos: el 58,5% de los encuestados actualmente utiliza videojuegos, el 36,7% los ha utilizado pero actualmente no los utiliza y sólo el 4,8% nunca ha jugado. Concluyendo, de una muestra de 3000 jóvenes españoles entre 14 y 18 años, el 95,2% ha tenido contacto con los videojuegos, por lo tanto es de suponer que hay una cierta familiaridad con los ambientes simulados en esa franja de edad.

Otros motivos para no usar sujetos simulados son que, al estar programados, tienen un *modelo implícito* [8][9] y que un mundo virtual siempre es una *simplificación del mundo real*. Como consecuencia de ambos factores, puede producirse que se guíe al participante hacia un conjunto de respuestas concreto y no responda de forma natural.

No cabe duda que los seres simulados, al ser artificiales, están impresos por las creencias de su programador, pero, precisamente por ello, se obtiene una de las principales ventajas. El investigador conoce perfectamente el por qué de su comportamiento, al predefinirlo él mismo Esto le da un *control absoluto sobre el comportamiento del ser simulado*.

Habitualmente, para hacer que el ser simulado parezca más real, se suelen usar en vez de un sistema de toma de decisiones determinístico, un sistema probabilístico, es decir, en vez de “si pasa A y luego B entonces siempre harás C” (determinístico), para que no se quede en un bucle eterno, se suele programar “si pasa A y luego B entonces harás C con una probabilidad x y D con otra probabilidad” (probabilístico). Esto le confiere cierta apariencia de ser “inconsistente” al sujeto simulado, lo que constituye un rasgo muy humano, pero es conocido y está limitado por el programador.

Es la principal ventaja en el estudio de la interacción. Al fijar el comportamiento de una parte del sistema, puedes observar mejor las diferencias características del

sujeto investigado y compararlo con otros sujetos, puesto que el comportamiento del ser simulado será idéntico con todos ellos, al no estar afectado por prejuicios ni por la "inconsistencia inexplicable" propia de los humanos.

Respecto a la simplificación del mundo real, creo que es una necesidad para poder *controlar las variables extrañas,* común tanto para los experimentos con seres simulados como aquéllos que no lo usan y, por lo tanto, no es un inconveniente propio de la simulación sino de las situaciones experimentales en general.

Otra de las reticencias es la *necesidad de una cierta habilidad en el uso del ordenador* para poder utilizarlos y puede que a ciertos grupos de personas no les sea factible utilizarlos.

Ciertamente, cualquier prueba que se pase con un ordenador, exige una cierta capacidad, pero con *un tutorial y algo de práctica* se puede solventar este problema menor. Ahora bien, las ventajas de hacer el experimento directamente sobre el ordenador son muy grandes, especialmente en el *ahorro de tiempo y dinero* porque la *categorización*, la *tabulación* y el *procesamiento de datos* se pueden *automatizar*. La categorización es especialmente relevante porque para que unos datos sean fiables es necesaria la concordancia de, al menos, dos observadores independientes.. Sin embargo, en un ambiente simulado y cerrado, como en el proyecto en el que estoy trabajando actualmente, el sistema de categorías es objetivo. La opción que marque el sujeto será registrada sin posible error de categorización, siempre y cuando esté bien programado el sistema, evidentemente.

También es posible que haya ciertas ventajas implícitas en el uso de un ordenador. Por ejemplo Turkle indica que "las personas tienen muchas razones para recurrir a la ayuda informatizada. Por el coste, la conveniencia, la constancia […] un ordenador psicoterapeuta no sería ni intimidatorio ni sentencioso" [2] Siguiendo esa línea de pensamiento, el artículo de Bainbridge y colaboradores [5] reseña un cierto número de investigaciones en las que una entrevista psicológica por ordenador puede obtener ventajas en temas como el comportamiento sexual, ideas de suicidio… También cabe reseñar que el ordenador, al no tener lo que el ser humano llama "sentido común", es una excelente mente científica sin prejuicio alguno.

Para concluir, creo que el principal inconveniente del uso de seres simulados para las investigaciones en psicología es la resistencia del ambiente académico a la simulación y la falta manifiesta de preparación de los psicólogos en temas relacionados con la informática. Todo ello provoca el rechazo irracional a todo lo que fluya de la simulación como método de investigación. Espero haber suscitado cierta curiosidad entre los lectores respecto a lo que pueden ofrecerse mutuamente el mundo de la psicología y de la informática y que se funden más grupos multidisciplinares que tienen mucho que aportar.

## Agradecimientos

Estoy especialmente agradecido por la colaboración, correcciones y consejos de Dionisio Cañas y Silvia Martín durante todo el proceso de elaboración del artículo. El apoyo y la ayuda tanto bibliográfica general como en los conceptos de "virtual", "artificial" y "simulado" de mi tutor de doctorado Vicenç Quera. Y las aportaciones

en el concepto “inmersión” de Pablo Cerviá, Arturo Ramírez, Jorge Santos y Adrián Scolari.

## Referencias


1. Weizenbaum, J.: ELIZA: A Computer Program for the Study of Natural Language Communication between Man and Machine. Communications of the ACM, Vol. 9, Num. 1 (1966) 36-45
2. Turkle, S.: La Vida en la Pantalla. Paidos, Barcelona (1997)
3. Turkle, S.: Seeing Through Computers. The American Prospect Vol. 9, Num. 31 (1997)
4. Leo Whang, S. M., Chang, G.: Lifestyles of Virtual World Residents: Living in the On-Line Game “Lineage”. CyberPsychology & Behavior, Vol. 7, Num. 5 (2004)
5. Yee, N.: The Norrathian Scrolls: A Study of EverQuest (Versión 2.5). http:/www.nickyee.com/eqt/report.html (2001)
6. Yee, N.: Through the Looking Glass: An Exploration of the Interplay Between Player and Character Selves in Role-Playing Games. http:/www.nickyee.com/daedalus/000755.php (1999)
7. Griffiths, M., Davies, M. N. O., Chappell, B.: Online Computer Gaming: A Comparison of Adolescent and Adult Gamers. Journal of Adolescence, Num. 27 (2004) 87-96
8. Frasca, G.: The Sims: Grandmothers Are Cooler than Trolls. Game Studies, Vol. 1, Num. 1 (2001)
9. Sicart, M.: Family Values: Ideology, Computer Games & The Sims. Conferencia. En: LevelUp. Universidad de Utrecht (2003)
10. Flanagan, M.: SIMple and Personal: Domestic Space and The Sims. MelbourneDAC (2003)
11. Boal, M.: Three Days in the Most Surreal Game on Earth: Me and My Sims. Village Voice. 29 de Marzo (2000)
12. Thompson, C.: Suburban Rhapsody. Psychology Today, November-December (2003) 32-40
13. Mateas, M., Stern, A.: Façade: An Experiment in Building a Fully-Realized Interactive Drama. http://www.interactivestory.net/papers/mateassterngdc03.pdf (2003)
14. Bosco, R., Caldana, S.: Las relaciones personales llenan el drama interactivo de “Façade”. El Pais, España, 12 de Septiembre (2005)
15. Bainbridge, W. S., Brent, E. E., Carley, K. M., Heise, P.R., Macy, M. M., Markovsky, B., Skvoretz, J.: Artificial Social Intelligence. Anna. Rev. Social. Vol. 20 (1994) 407-436
16. Rodríguez, E.: Jóvenes y Videojuegos. Espacio, Significación y Conflictos. FAD-INJUVE, Madrid (2002)
17. Baudrillard, J.: Cultura y Simulacro. Kairos, Barcelona (1996)
18. Dyaz, A.: Mundo Artificial. Temas de Hoy, Madrid (1998)
19. Wood, R. T. A., Griffiths, M., Chappell, B., Davies, M. N. O.: The Structural Characteristics of Video Games: A Psycho-Structural Analysis. CyberPsychology & Behavior, Vol. 7, Num. 1 (2004)
20. Funk, J. B.: Children and Violent Video Games: Are There “High Risk” Player?. Conferencia. En: Playing by the Rules: Video Games and Cultural Policy. Universidad de Chicago. (2002)
21. Thorngate, W. (1986). The Production, Detection and Explanation of Behavioral Pattern.. En: Valsiner, J. (ed.): The Individual subject and scientific psychology. Plenum, New York (1986) 71-93.

## Apéndice: Glosario

**Simulado**. Aparenta lo que no es. Jean Baudrillard en su escrito *La precesión de los simulacros* definió "Disimular es fingir no tener lo que se tiene. Simular es fingir tener lo que no se tiene." [17]. En nuestro contexto, lo sujetos simulados por ordenador son programas que *actúan* como si fueran sujetos reales, o al menos eso intentan.
**Virtual.** Que no es real. En el artículo se utiliza para aquellos programas que crean seres que son controlados por humanos. Por lo tanto la diferencia entre los seres simulados y virtuales es que los primeros son autónomos del control del ser humano, mientras que los segundos no lo son.
**Artificial.** Todo lo que no es natural. Antonio Dyaz, en su libro *Mundo Virtual,* define artificial como "Dícese de todo aquello incapaz de aparecer de manera espontánea fuera del hábitat del ser humano o de cualquier derivación de dicho hábitat" [18]. En mi caso definiré como artificial a todo aquello creado por el hombre. Tanto los seres virtuales, cómo los simulados, son artificiales, puesto que han sido creados por un programador.
**Avatar.** Ser virtual que constituye la representación de un ser humano dentro de un mundo simulado, por ejemplo un juego. Habitualmente puede "personalizarse" dicho avatar para que se identifique el jugador con su personaje. Se suele usar este término sobre todo en juegos de rol online.
**Inmersión.** Es el tiempo que una persona se integra en una realidad artificial, perdiendo la noción del tiempo y del espacio circundante. Wood y colaboradores, en su estudio sobre las características básicas preferidas por los jugadores de videojuegos, comentan "[Cuando] un jugador está absorto en la realidad virtual del juego, son menos conscientes del mundo real circundante" [19]. Aquí podemos obtener una primera definición de inmersión como el tiempo en el cual el jugador está absorto en la realidad del juego.

Preguntando a un grupo de personas que trabaja en temas de realidad virtual, inteligencia artificial y programación de videojuegos, concuerdan en que la virtud de ser "inmersivo" no recae principalmente en el juego sino que es el jugador, con su predisposición, el que hace posible la inmersión a esa realidad. No obstante afirman que los gráficos, la música, los efectos sonoros y el guión, así cómo su interactividad (la capacidad del juego de responder a los comportamientos del usuario) y su jugabilidad (la facilidad de aprender y controlar el juego) ayudan a que se pueda producir. Mientras los primeros (gráficos, música,…) lo hacen "realista", la interactividad crea la necesidad de una atención mantenida hacia lo que está ocurriendo en la pantalla y la jugabilidad dota de la sencillez necesaria para no tener que estar atento a "qué tecla tengo que tocar ahora" y no perder la "ilusión" de formar parte del mundo simulado.

Durante la revisión bibliográfica encontré un artículo muy interesante sobre la clásica controversia sobre la relación entre videojuegos y comportamiento violento, escrito por Jeanne Funk [20]. En él describe la inmersión en los mundos simulados como un método de evitación de los sentimientos negativos, como la ansiedad y la depresión. Para los jóvenes con problemas, esa situación les da sensación de poder y control sobre lo que está ocurriendo dentro de la pantalla, y que les suele faltar en su vida real, la inmersión en su caso puede ser un método terapéutico, o por lo menos positivo, para que estas personas disminuyan su ansiedad.

**Inconsistencia.** Cuando un organismo, ante una misma situación, realiza distintos comportamientos en dos momentos temporales diferentes. En palabras de Warren Thorngate "A menudo las personas son inconsistentes. No suelen producir la misma respuesta ante estímulos equivalentes, y, frecuentemente, no manifiestan una relación lógica entre lo que creen, dicen y hacen" [21]. Él no lo considera como un error o algo a extinguir sino que nos aporta la flexibilidad necesaria para poder vivir en los entornos cambiantes e inestables en los que se mueve el ser humano.

**Coherencia.** Cuando en una estructura todas sus partes tienen relación entre si. En el artículo se utiliza el término sobre todo en comunicación y comportamiento. Y se utiliza para medir el grado de relación causal o temática entro lo sucedido en un momento temporal antecedente y el momento temporal subsiguiente.

**Control Experimental.** Son las medidas que se toman en un experimento para asegurar que las variables estudiadas son el único motivo que produce los hechos medidos. Un ejemplo de ello podría ser una investigación sobre la relación entre comer chocolate y engordar. Le pedimos a un grupo de sujetos que tomen dos tabletas de chocolate al día y otro grupo que no coma ninguna y medimos su peso diariamente. Partimos de la hipótesis de que comer chocolate hace engordar. Si sólo controlamos el chocolate que comen, podríamos encontrarnos que el grupo que no come chocolate engorda y el que lo toma adelgaza. Esto indicaría que nuestra hipótesis es falsa. Ahora bien, podría haber otras explicaciones que no hemos contemplado. Por ejemplo, que el grupo que comen chocolate ingieren ese alimento exclusivamente mientras que el otro se comen tres pizzas cada uno. Para evitar estas explicaciones alternativas (dieta de los grupos), se "controlan", es decir, se igualan o aleatorizan, las posibles variables que puedan estar influyendo en el objeto medido (en este caso, el peso), para así asegurar que la variable experimental (tabletas de chocolate) sea la única explicación del resultado.

**Variables Extrañas.** Son todas las variables que no han sido "controladas" y que están influyendo en la variable medida. Este es el mayor riesgo de un diseño experimental ya que puedes obtener una conclusión totalmente errónea, como la obtenida en el ejemplo.

**Validez Ecológica.** Es la capacidad que tienen los resultados de una investigación de poder generalizarse fuera de la propia situación experimental. Es decir, si los datos obtenidos en un experimento son medidas realistas de lo que ocurre fuera de él. Habitualmente hay una relación inversa entre el control experimental y la validez ecológica ya que cuantas más variables extrañas se controlan menos natural es la situación y a la inversa.

# Niveles de descripción para la interpretación de secuencias de vídeo en tareas en vigilancia

Margarita Bachiller Mayoral, Rafael Martínez Tomás, José Mira Mira y Mariano Rincón Zamorano

Dpto. Inteligencia Artificial, ETSI Informática, Universidad Nacional de Educación a Distancia, c/Juan del Rosal 16, 28040 Madrid, Spain,
{marga, rmtomas, jmira, mrincon}@dia.uned.es

**Resumen.** En este trabajo se presenta una estructura de tareas para el modelado de la tarea global de vigilancia analizándola como tarea de control de sistemas. Parte fundamental de la tarea de vigilancia es la adquisición automática desde fuentes plurisensoriales de hallazgos relevantes de alto nivel de abstracción. Como el salto semántico entre las señales de entrada y la descripción en lenguaje natural de la escena es muy amplio, es aconsejable usar una jerarquía de niveles de descripción con grado creciente de semántica que sistematice el proceso de interpretación. En este trabajo se describe una propuesta metodológica que implica lo anterior y se ilustra sobre un escenario simplificado que se interpreta a varios niveles, dejando claras y explícitas las relaciones entre las distintas descripciones de una misma escena.

**Palabras clave:** Percepción Artificial: Visión Artificial y Visión Cognitiva

## 1 Introducción

La vigilancia es una macrotarea multidisciplinar que afecta a un número cada vez mayor de escenarios, servicios y usuarios. Su objetivo es la detección de amenazas a través de la observación continua de áreas importantes y vulnerables de un escenario considerado de valor económico, social o estratégico por ser susceptibles de robos, incendios, vandalismo o atentados. La gama de escenarios, y por tanto de necesidades, es muy diversa y de muy diferente complejidad, desde la simple detección de movimiento que hace saltar la alarma, hasta todo un sistema integral de control que monitoriza plurisensorialmente, diagnostica la situación y planifica una serie de actuaciones coherentes. En cualquier caso, implica la observación de objetos móviles (personas, vehículos, etc) en un ambiente determinado para proporcionar una descripción de sus acciones e interacciones. Esto supone, a su vez, la detección de los objetos en movimiento y su seguimiento, la clasificación de los objetos, análisis del movimiento particularizado para humanos e interpretación de la actividad. En los últimos años ha habido una cantidad significativa de trabajos relacionados con el análisis de la actividad a través de eventos en diferentes dominios de aplicación (por ejemplo ver [1,2]). Una buena revisión de las propuestas más significativas se puede encontrar en [3]. La mayoría de las aproximaciones para el análisis de la actividad se basan en

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 450-462, 2006.

modelos definidos para un tipo específico de actividad en un determinado dominio de aplicación, siendo altamente dependientes de los resultados obtenidos durante el seguimiento. El problema fundamental es el enorme salto semántico que hay entre el nivel físico de las señales y el nivel de conocimiento. Para facilitar este salto es necesario segmentarlo e inyectar una gran cantidad de conocimiento.

Así, hay consenso en el área, dentro de cierta variedad y dispersión en la nomenclatura, en aceptar distintos niveles de descripción con grado de semántica creciente [4,5]. En nuestro trabajo se consideran cuatro niveles de descripción: *píxeles, blobs*, *objetos* y *actividades-comportamientos*, de forma que permitan manejar modular e independientemente, cada uno de estos niveles, y en donde la información de cada nivel se obtiene de la información suministrada por el nivel inferior.

En la sección 2 de este artículo se analiza la tarea de vigilancia y en la sección 3 se introducen los diferentes niveles de descripción orientados a dicha tarea, a través de ejemplos en un escenario limitado y de un prototipo implementado. Este prototipo es de marcada modularidad y permite la inclusión, a medida que se complique el escenario, de nuevas técnicas y operadores, y la evaluación y comparación de los correspondientes resultados.

## 2 La tarea de vigilancia

La tarea de vigilancia en seguridad sigue una estructura común con tareas de control en general, tales como el control industrial y la medicina (particularmente en la medicina clínica). Comparten los procesos de 1) *monitorización* de una serie de variables críticas, cuya desviación de la normalidad es signo (o síntoma en medicina) de alguna disfunción, 2) *diagnóstico* del problema, con una mayor o menor interacción con el sistema y 3) la *decisión* (sistema consejero) de las acciones pertinentes para la resolución del problema. Este último proceso puede ser sencillo, por ejemplo una mera tabla de asignación de diagnósticos-acciones, o puede ser más complejo, incluyendo una planificación temporal de acciones. En cualquier caso, optamos por una aproximación semiautomática en la que, al final, es el operador de una central de alarmas quien toma las decisiones [6].

La salida del proceso de *monitorización,* que tiene consecuencias en el proceso posterior, es el aviso de que algo anómalo se ha detectado a través de la observación del sistema. Es, por tanto, la alarma que provoca la activación de un proceso de diagnóstico. La monitorización que se pretende en AVISA [7] es de tipo reactivo (con esquemas sensoriales precalculados), sin procesos de razonamiento computacionalmente costosos que retrasen la emisión de la alarma. Pero caben dos posibles relaciones con el *diagnóstico* que determinan el significado de este en el proceso global:

a) La alarma dispara el proceso de diagnóstico de "lo ocurrido". Es decir, hay un análisis de lo sucedido a partir de la grabación de las imágenes (y datos de otros sensores) utilizando cuanto recursos computacionales se requieran. Este proceso de análisis implica un retardo (razonable) entre el suceso y la conclusión del diagnóstico. La monitorización detecta que ha ocurrido ya algo, por ejemplo, se ha roto un cristal, y el diagnóstico intenta determinar la causa.

b) La alarma dispara el proceso de diagnóstico de lo que está ocurriendo. Hay una asignación de recursos computacionales (se desvían de otros usos) que permiten la interpretación on-line. El diagnóstico clasifica lo que ocurre a partir de la *prealarma* confirmando o descartando una situación de respuesta determinada.

Esta diferenciación entre monitorización y diagnóstico es coherente con el trabajo de Howarth y Buxton [8], por ejemplo, en el que se habla de *monitorización* como "pasivamente pasar de imágenes a descripciones conceptuales", mientras que la *vigilancia* (watching) es "más activa, orientada a la tarea", con realimentación entre un nivel *preintencional* desde el que emergen comportamientos (a partir de las imágenes) y un nivel *intencional* de control, que focaliza en el proceso de la tarea de vigilancia.

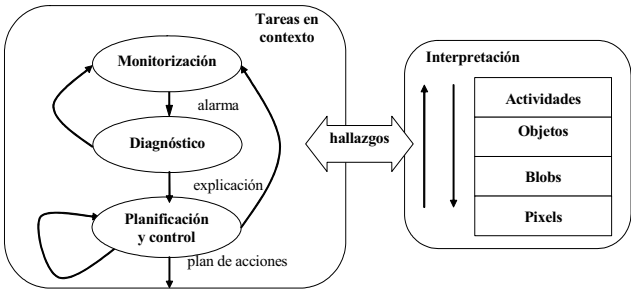



**Fig. 1** Descomposición en tareas de la macrotarea de vigilancia en AVISA. En cualquiera de las subtareas se precisa la interpretación de la escena desde la estructura de niveles de descripción.

Hablamos por tanto de unas tareas de monitorización, de diagnóstico y de planificación basadas en la interpretación, esto es, que hacen uso de los hallazgos y/o observables más o menos abstractos. La monitorización usa estos hallazgos y/o observables de una forma más directa, reactiva, como ya hemos dicho, y el diagnóstico y la planificación de una forma más elaborada. La interpretación o navegación por la estructura de niveles, admite las dos direcciones: a) ascendente (*bottom-up)*, en la que se identifican (componen, emergen) actividades desde los píxeles, b) descendente (*top-down)*, en la que se plantean actividades como hipótesis que se confirman o descartan. Diferenciaríamos, así, como ocurre en medicina, síntomas y signos procedentes de la exploración clínica de los procedentes (emergentes) de aquellos que provienen de exploraciones complementarias (en que se focaliza en determinadas características, por ejemplo, en una radiografía).

## 3 Niveles de descripción

El principal problema está en la necesidad de adaptar la información sensorial a las exigencias de los roles de entrada y salida de las *inferencias* en términos de las cuales se modelan las tareas. Este problema de análisis de los datos y obtención de un conjunto de descripciones concurrentes con distintos grados de semántica queda implíci-

to en las descripciones de las tareas. Así, para hablar de "blobs" y de separación fondo-formas, hablamos de segmentación y cuando hablamos de configuraciones espacio-temporales de eventos estamos hablando de objetos y sus relaciones. Pero antes de poder hablar de tareas de vigilancia en escenarios concretos con objetivos concretos, es decir, de tareas en contextos, necesitamos definir los distintos niveles de descripción necesarios para poder usar, en esas tareas, las señales procedentes de los sensores y para poder interpretar las secuencias de imágenes.

A medida que la semántica aumenta es más necesaria la contribución de una organización complementaria (*organización descendente*) para manejar el incremento de la complejidad y la incertidumbre, que se corresponde con el concepto usual de SBC o con la *búsqueda guiada* por conocimiento, propia de los mecanismos de atención visual selectiva. Este proceso descendente contribuye a especificar lo que cada tarea demanda de cada nivel de descripción y, además, cierra el lazo de realimentación, con lo que aumenta la estabilidad del sistema.

### 3.1 Escenario para un prototipo

En primer lugar, definimos un escenario específico que sea el banco de pruebas de nuestro desarrollo. El orden natural es especificar primero "*dónde y qué*" vigilar (escenario), después especificar "*para qué*" vigilar (objetivos) y, finalmente, especificar los *sensores y efectores* de los que disponemos para realizar esa tarea en ese *escenario*.

Para ejemplificar la propuesta metodológica desarrollada en este trabajo se define un escenario muy sencillo, tal como el que se ilustra en la Fig. 2, al que hemos llamado "*escenario del sofá*". Se trata de un espacio interior que es zona de paso de humanos que pueden llevar un maletín, sentarse en el sofá, levantarse y llevarse o no el maletín, sentarse y recoger el maletín que dejó otro humano, salir de la zona de observación, etc.

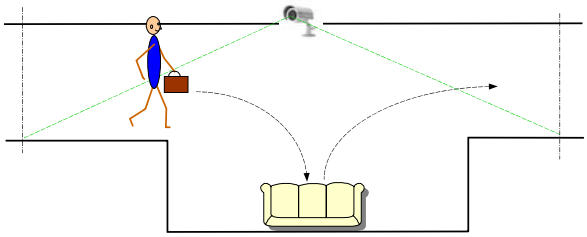


**Fig. 2** Versión inicial del escenario del sofá.

Así definido, estaríamos ante un escenario esencialmente de monitorización y muy reactiva, en la que la sucesión de una secuencia determinada de actividades elementales, normales cada una de ellas en principio, puede conducir a la interpretación como actividad anómala en conjunto. No hay lugar para un diagnóstico, entendido éste como la búsqueda de la explicación de hecho, y la planificación de la acción pertinente también se simplifica al máximo. Por ejemplo, avisando al guardia de seguridad o inspeccionando del maletín abandonado. El escenario, por tanto, está orientado casi

exclusivamente a la interpretación ascendente (*bottom-up*).

Este escenario nos acota el problema (no hay perros ni árboles, ni coches, etc.) y nos ayuda a especificar las otras dos componentes de un contexto (objetivos y sensores). Una señal de prealarma[1] surgiría si alguien abandona un maletín en el área bajo vigilancia. En cuanto a sensores, partimos también de la configuración más simple: una sola cámara fija enfrente, a distancia conocida, calibrada y cubriendo todo el área de interés.

El escenario se complica con facilidad si incluimos una puerta y/o una columna (oclusiones), o si se acepta que por delante del sofá pueden pasar un número elevado de sujetos con capacidad de interactuar e intercambiar maletines (necesidad de identificación, etiquetado y seguimiento), por ejemplo.

Del escenario planteado obtenemos las siguientes especificaciones:

- Los objetos de interés para nuestro sistema serán: el hombre y el maletín.
- Es necesario seguir a la persona (seguimiento) desde que entra en la escena hasta que sale.
- Los objetos trasportados no son distinguibles durante el movimiento de los humanos que los trasportan por la limitación de los algoritmos de segmentación utilizados, sólo ”aparecen” en la escena (en una posición intermedia) cuando los dejan los humanos, localizados por un cambio significativo y estático en el fondo de la escena.

### 3.2 Descripción a nivel físico

Este nivel corresponde a la información procedente de uno o varios sensores. En la mayoría de las aplicaciones, un único sensor es suficiente para localizar y seguir los objetos móviles en la escena. Sin embargo, existen sistemas en los que se utiliza un conjunto de sensores para aumentar el área de vigilancia [9] o incluso un par estéreo para conseguir información tridimensional de la escena. Es en este nivel donde se realiza la fusión e interpretación de las señales suministradas por los distintos sensores para obtener el conjunto de píxeles que definen la escena. Cada uno de estos píxeles tiene asociado una serie de características, tales como color, coordenadas de cámara, etc.

En nuestro caso, se utiliza una cámara fija perpendicular al sofá, lo cual simplifica la determinación de las coordenadas tridimensionales de un píxel. En este nivel la información solo puede estar en el nivel de color de los píxeles de la imagen en formato RGB y sus coordenadas de cámara. No se acepta otra semántica.

### 3.3 Descripción a nivel de blobs

La entidad en este nivel es el blob. Éste se define como zona estática o dinámica de una imagen o secuencia de imágenes en la que un conjunto de características se conserva dentro de un rango de valores. Podemos clasificarlas en blobs móviles y blobs

[1] Dejamos la señal de alarma propiamente dicha, que supondría el aviso a la policía del aeropuerto, a la identificación de la situación en la que la misma persona abandone el aeropuerto sin el maletín.

estáticos. Cada blob tiene asociado un vector de características obtenidas a partir del análisis de la forma, color, textura y movimiento.

En el prototipo desarrollado se utilizaron las características ofrecidas por la librería MIL de Matrox para la caracterización de las regiones. Partiendo de la imagen RGB captada por el sensor, se segmenta empleando un algoritmo basado en la sustracción del fondo [10]. El resultado de esta segmentación es un conjunto de blobs con un vector de características asociado (área, caja limítrofe, píxeles superior e inferior del contorno y nivel de gris medio). En la Fig. **3** se muestra un ejemplo de transición del nivel físico al nivel de blob.

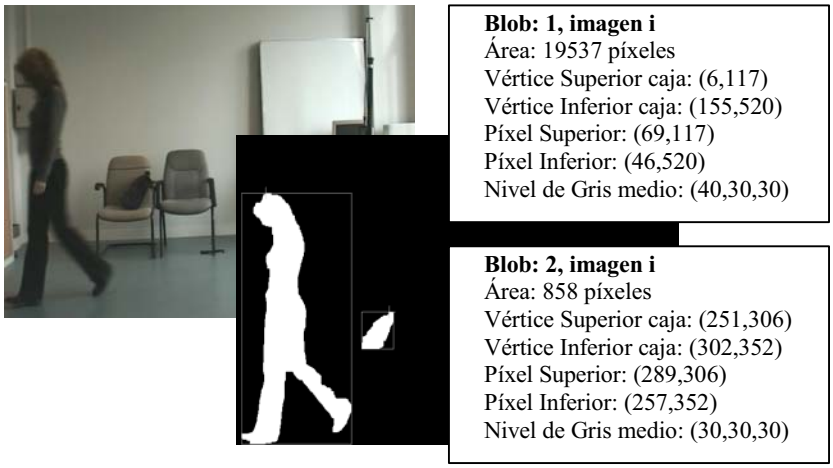



**Fig. 3** Transición del nivel físico al nivel de blobs: Imagen RGB del nivel físico e imagen y descripción de los blobs segmentados.

### 3.4 Descripción a nivel de objetos

En este nivel, se reorganiza la información asociada a los blobs para producir una descripción de los objetos de interés en la escena. Se pasa de una descripción de imagen a imagen a otra orientada a objetos. En la Fig. 4 se muestra la estructura de clases utilizada en nuestro prototipo, junto con instancias de tipo *humano* y *maletín* asociadas a la escena de la Fig. **3**. Obsérvese que a este nivel ya es necesario inyectar un grado de semántica razonablemente alto, en comparación con la del nivel de blobs.

En el nivel de objetos se describen los modelos de los objetos de interés. Estos modelos contienen: 1) la caracterización visual del objeto y su evolución espacio-temporal; 2) las relaciones de composición utilizadas para describir objetos complejos; 3) las relaciones entre objetos para la generación de la descripción geométrica de la escena orientada a la tarea.

El uso de modelos permite, por un lado, mejorar la segmentación al facilitar la detección de las partes constituyentes del objeto, determinar regiones de posible confusión entre objeto y fondo y mejorar el seguimiento, pudiendo establecer métodos de seguimiento basados en partes del objeto y métodos de asociación parcial entre blobs en instantes consecutivos. Además, el modelo del objeto permite clasificar los objetos

y gestionar patrones de características y de comportamientos dependientes del tipo de objeto, que pueden ser utilizados como realimentación en tareas de reconocimiento y de seguimiento.

Los requisitos impuestos por la tarea de vigilancia y el entorno en el que se desarrolla obligan a que exista, además, un modelo del fondo, que actualmente permite gestionar variaciones en la iluminación (controlar reflejos, sombras, contraluces, etc) y pasar de coordenadas en la imagen (2D) a coordenadas 3D mediante un proceso de calibración sencillo [11].

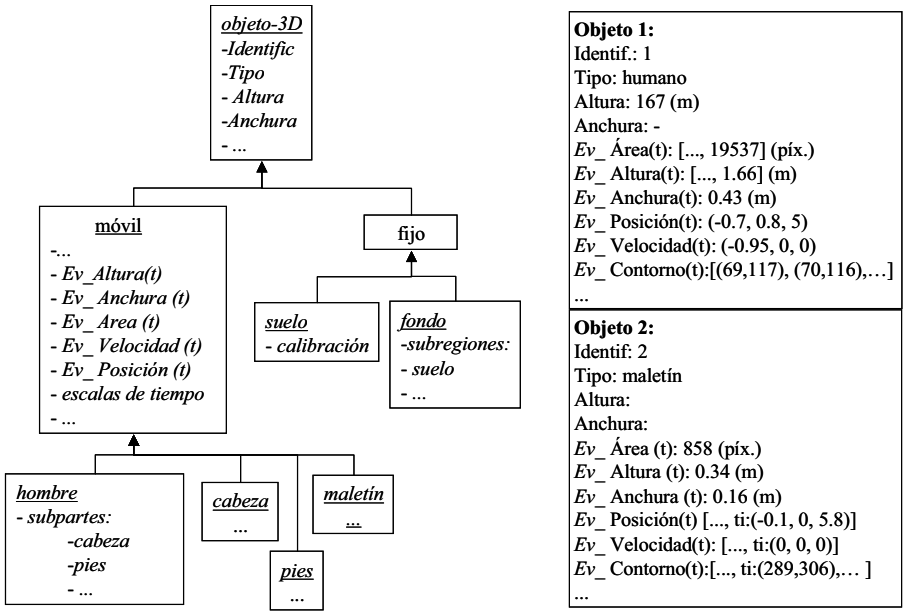



**Fig. 4** Estructura de clases utilizada para la representación del nivel de objetos e instancias del tipo *humano* y maletín asociadas a la escena representada en la Fig. **3**.

### 3.5 Descripción al nivel de actividad

En este nivel se describen los estados y los eventos utilizados para describir la actividad, orientada a la tarea, presente en la escena, esto es, la realizada por los objetos de interés. Los eventos están organizados de forma jerárquica, distinguiéndose entre *eventos primitivos,* asociados a transiciones (en el tiempo) en las variables visuales y espaciales de los objetos, y *eventos compuestos,* los cuales se definirán a partir de los anteriores y en orden creciente de complejidad.

En la ontología de *eventos*, cada *evento* viene caracterizado por una serie de atributos (Tabla 1): unos genéricos de la clase *evento,* como son el *identificador*, *el tipo* y el *contexto*, y otros dependientes del tipo de evento (altura de la transición, medida del cambio de velocidad, duración, ...).

**Tabla 1** Marco básico para la descripción de eventos

| Evento | |
|---|---|
| ***identificador*** | Identificadorespecífico del evento |
| ***tipo*** | Identificadordel tipo de evento |
| ***contexto*** | localización:<br>asociadoa-objetoX<br>posición espacial<br>tiempo |
| ***valor*** | dependientes del tipo de evento |

**Eventos primitivos**

Dentro de la jerarquía de eventos, los *eventos primitivos* [12] se determinan a partir de cambios de estado de atributos visuales. Previamente, se ha definido una clase abstracta de *eventos de base* asociados a una sola variable unidimensional, que facilita la definición de *eventos primitivos* por instanciación de los *eventos de base*. No obstante, otros *eventos primitivos* podrán obtenerse a partir de un espacio de parámetros más complejo mediante otros métodos de clasificación.

Los eventos en tareas de vigilancia están asociados con transiciones en el tiempo. Si analizamos la generación temporal de estos eventos, observamos que están relacionados con (1) el estado anterior a la transición, (2) con el estado posterior, (3) con la relación entre el estado anterior y posterior o (4) con características de la propia transición. En un sistema de vigilancia, en el que es interesante adelantarse a los acontecimientos, puede ser clave fijarse en eventos de los tipos 1 y 4, pues es posible evaluarlos en el mismo instante en el que se produce la transición.

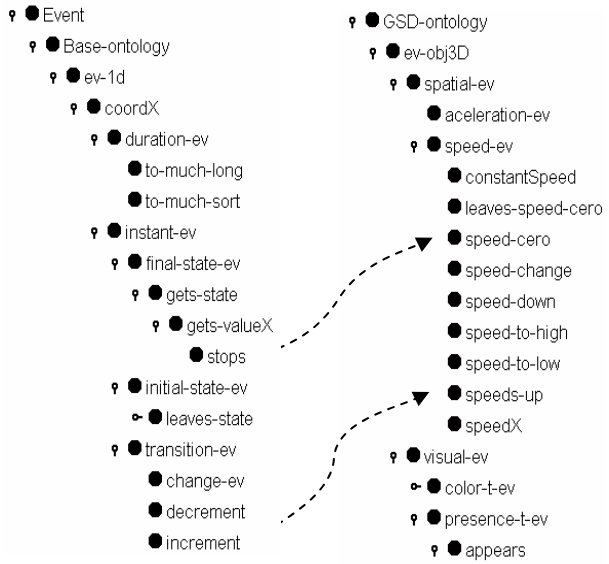



**Fig. 5** Ontología de eventos para variables unidimensionales

En la Fig. 5 se muestra la jerarquía de eventos de base organizados de acuerdo con este criterio y un ejemplo de su relación con eventos primitivos obtenidos al asociar la ontología de base con las características de los objetos del dominio (velocidad, en este caso).

Hay que destacar también que existe otro tipo de eventos no relacionados con una transición de estados, sino justo con lo contrario, con una duración inadecuada de un estado. Este tipo de evento está relacionado con una predicción implícita, que adelanta un determinado tipo de evento por medio de alguna relación causal entre una duración anormal de un estado y el evento real. Por ejemplo, si un coche va a llegar a un atasco (menos de 50mts.), va a una velocidad mayor que 50 y no frena, entonces ...

**Eventos compuestos**

La combinación de *eventos* más simples (primitivos o no) genera *eventos compuestos*. Las operaciones para combinar eventos se basan en relaciones espacio-temporales, por ejemplo, por simple coincidencia total o parcial basada en una lógica de intervalos [13] espacio-temporal o por secuenciación de acuerdo con un autómata finito [12,14]. La estructura jerárquica termina cuando se enlaza con aquellos eventos relacionados directamente con la tarea objetivo (con la vigilancia o con la descripción de la escena en el nivel de abstracción adecuado). En nuestro trabajo, como en [12,14], no se ha diferenciado entre estados y eventos.

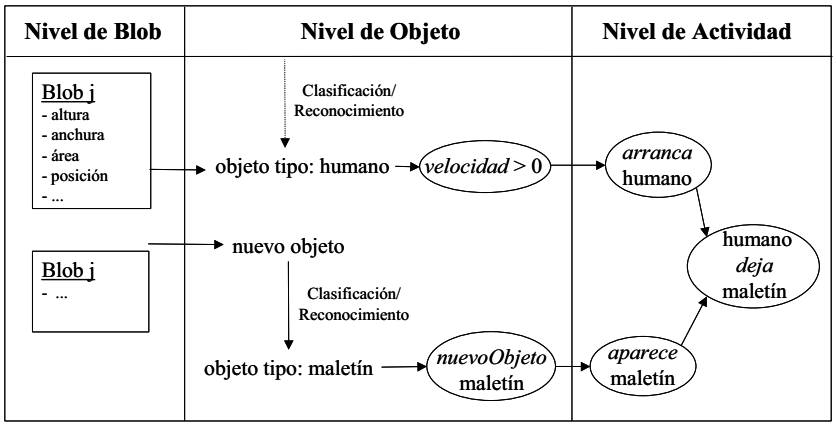



**Fig. 6** Ejemplos en que se transmiten eventos desde el nivel de blob al nivel de objeto y desde éste al nivel de actividad, hasta conformar eventos complejos como interpretación a alto nivel de la secuencia de imágenes.

La Fig. 6 muestra, a través de dos ejemplos sencillos, la evolución desde el nivel de blob hasta la descripción de actividades. En primer lugar, a partir de las características de los blobs obtenidos, se realiza la identificación de los objetos que intervienen en la escena. Una vez clasificado el objeto, se va actualizando la descripción del mismo con la información contenida en blobs de instantes sucesivos asociados al mismo objeto y de acuerdo con las escalas de tiempo adecuadas. A continuación, se generan una serie de eventos primitivos a partir de la evaluación de las características

temporales asociadas al objeto. Determinados eventos del nivel de objetos emergen hasta el nivel de actividades como eventos simples con una implicación semántica mayor en la descripción de la escena. Por ejemplo, el inicio del desplazamiento de un humano se transmite al nivel de actividades como el evento simple `arranca(objetoTipo=humano, posicion=`$x_o,y_o,z_o$`)`, que afecta a un único objeto. Si coincide con la aparición de un maletín (se produce otro evento simple `aparece(objetoTipo=maletín, posición=`$x_1,y_1,z_1$`)` y éste está en la zona cercana al humano (relación de cercanía entre los respectivos vectores de posición) entonces se genera el evento de tipo *deja(humano, maletín, posición)*.

**Eventos en el prototipo**

En la Tabla 2 se muestra la descripción de eventos en el desarrollo del prototipo a partir de las características de la escena planteada y en la

Fig. **7** el modelo de transición de estados asociado a la descripción de la escena considerando estos eventos como entradas. El prototipo se ha desarrollado inicialmente definendo las clases *eventos* y *estados* y, dentro de cada una de ellas, las subcategorías definidas en la. La transición de estados se produce como consecuencia de la generación de una instancia de la clase *eventos* de acuerdo al autómata de la

Fig. **7**.

En la Fig. 8 se muestra un volcado de la ventana del entorno Clips [15] en el que se ha desarrollado el prototipo; en la parte izquierda aparece la secuencia de eventos que llegan desde niveles anteriores y los que se componen en el nivel de actividades, y en la parte derecha aparece la secuencia de estados que implican según el autómata de la

Fig. **7**.

**Tabla 2** Relación de eventos utilizados en el nivel de actividad para la descripción de la escena del prototipo

| Nivel de actividad | Tipo | Descripción |
|---|---|---|
| Entra(h, (x,t)) | Simple | Aparece un humano en la escena |
| Sale(h, (x,t)) | Simple | Desaparece un humano de la escena |
| SeDetiene(h, (x,t)) | Simple | Un humano se detiene |
| Arranca(h, (x,t)) | Simple | Un humano se mueve |
| SeAgacha(h, (x,t)) | Simple | Un humano se agacha |
| SeLevanta(h, (x,t)) | Simple | Un humano se levanta |
| Deja(h, o, (x,t)) | Compuesto | Un humano abandona un objeto |
| Recoge(h, o, (x,t)) | Compuesto | Un humano recoge un objeto |

El objetivo de este prototipo no ha sido mejorar otros sistemas sobre escenarios similares y sobre tal o cual funcionalidad o capacidad, sino ilustrar la diferenciación de niveles propuesta y fijar la estructura consiguiente en un "banco de pruebas" de marcada modularidad sobre el que incluir, a medida que se complique el escenario, nuevas técnicas y operadores, y evaluar y comparar, entonces sí, los correspondientes resultados.

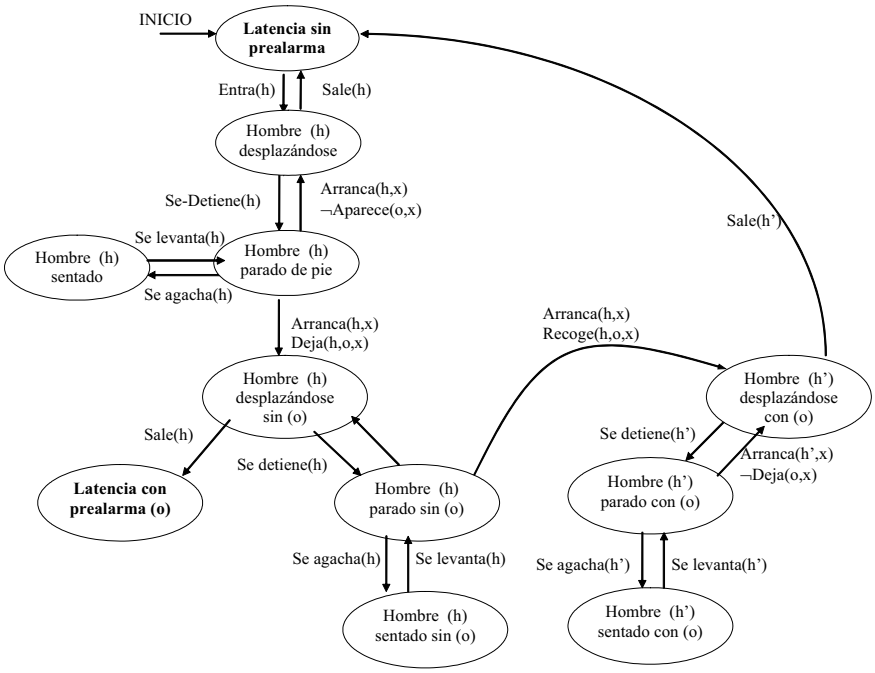



**Fig. 7** Diagrama de Transición de Estados

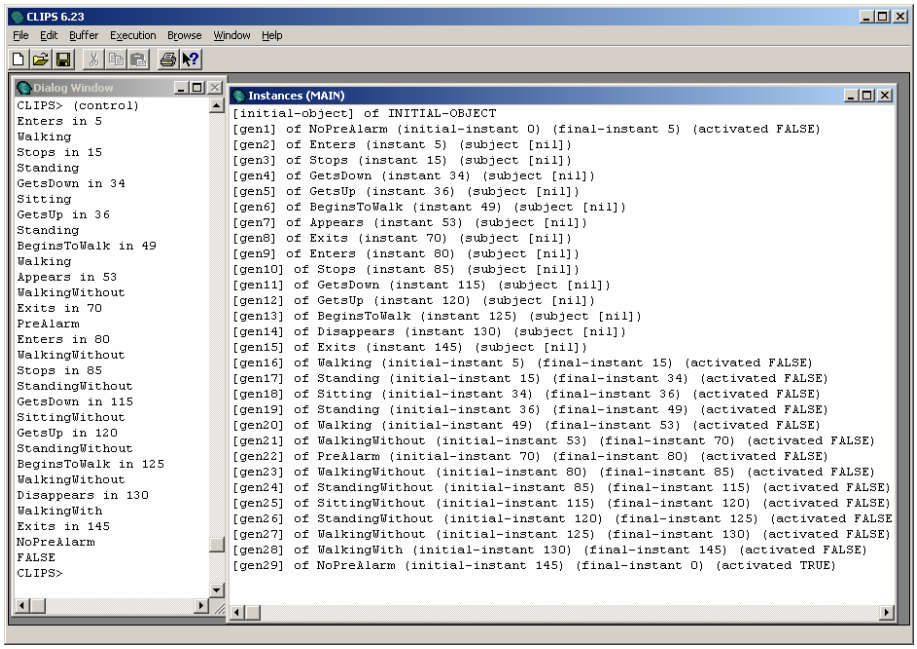



**Fig. 8** Volcado de pantalla del prototipo con la secuencia de eventos (izda.) y de estados (der.) por los que evoluciona el escenario

## 4 Conclusiones

En este trabajo se ha presentado una estructura para el modelado de la tarea global de vigilancia. Se han identificado las subtareas de monitorización, diagnóstico y planificación de acciones de respuesta, a semejanza de la descomposición habitual de la tarea de control de sistemas. Este modelo se ha mostrado coherente con las funcionalidades habituales de cualquier sistema de vigilancia.

La descripción de la tarea en lenguaje natural por parte del experto humano, y en particular, el papel de hallazgos, que son las entradas de las tareas de monitorización y diagnóstico, quedan muy alejadas del nivel físico. Es preciso, por tanto, un proceso de interpretación automático, especialmente en el caso de secuencias de imágenes, que permita salvar el salto semántico existente y transformar las observaciones (desde fuentes plurisensoriales pero en un nivel físico) en hallazgos de alto nivel de abstracción, en términos cercanos al lenguaje natural. Para ello se precisa una jerarquía de niveles de descripción con grado creciente de semántica que sistematice el proceso de interpretación.

En este artículo se ha descrito la propuesta metodológica que implica lo anterior y se ha ilustrado con un prototipo simplificado. Incluso en este ejemplo tan sencillo, que interpreta de forma orientada a la tarea de vigilancia una escena controlada, se puede observar la complejidad del problema de la interpretación de secuencias de vídeo en vigilancia y la necesidad de segmentarlo para facilitar la inyección de conocimiento del dominio, consiguiendo con ello soluciones más robustas y modulares.

## Agradecimientos

Los autores desean expresar su agradecimiento al apoyo económico de la CiCYT a través del proyecto TIN-2004-07661-C02-01.

## Referencias

1. Regazzoni, C. S., Fabri, G. and Breñaza, G. (Eds.): Advanced Video-Based Surveillance Systems, Kluwer Academia Publishers, Dordrecht (1999).
2. Colins, R. T., Lipton, A. J. and Kanade, T. (Eds.): Special Issue on Video Surveillance, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) 2000.
3. Aggarwal, J. K. and Cai, Q.: Human motion analysis: a review, Comput. Vision Image Understand. 73 (1999) 428-440.
4. Nagel, H. H.: Steps towards a cognitive vision system. AI Magazine 25 (2) (2004) 31-50.
5. Neuman, B. and Weiss, T.: Navigating through logic-based scene models for high-level scene interpretations. In Proc. 3er. Int. Conf. on Computer Vision Systems (ICV-2003), Springer (2003), 212-222.
6. Mira Mira, J. Memoria del proyecto TIN2004-07661-C02-01, AVISA: Diseño e implementación de un conjunto de agentes de diagnóstico, planificación y control, con capacidad de aprendizaje y cooperación con humanos en tareas de vigilancia. UNED (2004).
7. Mira Mira, J. and Fernández Caballero, J.A. (coords.) Documento interno D1-05 del proyecto AVISA. UNED (2005).

8. Howarth, R.J. and Buxton, H.: Conceptual descriptions from monitoring and watching image sequences. Image and Vision Computing, 18 (2000) 105-135.
9. Collins, R. T., Lipton, A. J. , Kanade, T., Fujiyoshi, H. , Duggins D., Tsin, Y., Tolliver, D. Enomoto, N. Hasegawa, O., Burt P. and Wixson L.: A System for Video Surveillance and Monitoring. CMU-RI-TR-00-12 (2000).
10. Haritaoglu, I., Harwood, D. and Davis, L. S.: W4: Real- Time Surveillance of People and Their Activities. IEEE Trans. On Pattern Analysis and machine Intelligence, 22(8) (2000).
11. Zhao, T. and Nevatia R. Tracking multiple humans in complex situations. IEEE trans. on Pattern Analisis and Machine Intelligence, 26 (9) (2004)
12. Bremond, F., Maillot, N. Thonnat, M. and Vu, V.-T: Ontologies for video events. INRIA: Report de investigación: 5189. (2004)
13. Allen, J.F. Towards a General Theory of Action and Time. Artificial Intelligence, 23 (1984) 123-154.
14. Bolles, B. and Nevatia, R.: A hieralchical video event ontology in OWL. Final report 2004 ARDA Project.
15. CLIPS Version 6.0. User's Guide & Reference Manual. NASA. L.B. Johnson Space Center (1993).

# Principios dinámicos en el estudio de la percepción

M.G. Bedia[1], J.M.Corchado[2] y J. Ostalé[3]

[1] Departamento de Informática
Universidad Carlos III de Madrid
Av. de la Universidad, 30, 28911, Leganés (Madrid), España
mgbedia@inf.uc3m.es
[2] Departamento de Informática y Automática
Universidad de Salamanca
Plaza de la Merced s/n, 37008, Salamanca, España
corchado@usal.es
[3] Departamento de Filosofía, Lógica y Filosofía de la Ciencia
Universidad de Salamanca
Campus Miguel de Unamuno s/n, 37007, Salamanca, España
ostale@usal.es

**Resumen.** En pleno debate entre computacionalistas y dinamicistas sobre cuál es la verdadera naturaleza de la cognición, nos proponemos en este artículo revisar algunos de los "modelos dinámicos sobre la percepción" que existen en la literatura. Desde una perspectiva dinámica, el fenómeno de la percepción se entiende como un proceso de formación de patrones, derivados del procesamiento de la información sensorial, y estrechamente vinculados al control de la acción. Se han planteado en los últimos años algunas propuestas, que llamaremos cognitivas, y que interpretaban la percepción como un fenómeno cognitivo en un régimen equivalente al de esquemas tradicionales de aprendizaje o de razonamiento. Algunas pocas arquitecturas artificiales se han implementado utilizando este tipo de modelos y sus resultados han generado importantes expectativas. En este artículo pretendemos explorar las propuestas y modelos dinámicos de la percepción con una inspiración biológica donde ésta pueda ser entendida como un proceso de codificación de información a lo largo de redes neuronales. En la última parte del artículo planteamos que considerar una red neuronal como un sistema dinámico no-lineal, podría proporcionar propiedades muy interesantes al diseño de redes neuronales artificiales (en términos de la capacidad de almacenado y de procesamiento de información) con el fin de diseñar modelos de percepción-acción artificiales más robustos, más sensibles a información con ruido y en sintonía con recientes teorías neurobiológicas.

## 1 Introducción

El punto de vista tradicional que la Inteligencia Artificial (IA) y las Ciencias Cognitivas (CC) han mantenido históricamente en sus modelos de percepción defiende una estructura secuencial: el sistema perceptivo procesa información sensorial construyendo una representación interna, ésta se manipula mediante el razonamiento y finalmente, a partir de las conclusiones obtenidas se selecciona la acción adecuada.

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 463-474, 2006.

Este tipo de descripción asume implícitamente en los agentes la existencia de un estadio intermedio entre la recepción de información y las acciones que ejecutan, donde, de alguna manera, el entorno queda "representado". Por tanto, la misión de la percepción es, para la IA clásica, producir y actualizar un modelo interno del entorno que será manipulado mediante un sistema de reglas de inferencia formales.

La primera cuestión que surge al aceptar esta perspectiva es si es necesario que el entorno sea "re-presentado" dentro de un agente. Esta idea fue introducida por primera vez por [Craik, 1943] explícitamente con el propósito de explicar cómo los agentes inteligentes eran capaces de actuar generando planes que respondiesen a los estímulos del entorno. El rasgo más importante de un modelo con representaciones internas es que ofrece la posibilidad de utilizar información "off-line", es decir, utilizar las representaciones sin necesidad de que se encuentre el estímulo presente.

Sin embargo, existen autores [Grush, 2003; Keijzer, 2002] que advierten cómo ese modelo tradicional no es útil en la descripción de muchos problemas de percepción en formas de vida simples como insectos (por ejemplo, la impresionante exhibición de rutinas de escape de una cucaracha es algo de lo que carecen los mejores sistemas artificiales, y seguramente no es gracias a las codificaciones explicitas ni a las deducciones lógicas), pero tampoco en humanos. Esta corriente (conocida como dinamicista [Port and van Gelder, 1995] por el uso que hace para la formalización de herramientas que provienen de la teoría de sistemas dinámicos) estudia los procesos perceptivos desde una perspectiva funcional: pretenda explicar cómo funciona el sistema perceptivo en animales y cómo puede ser usado en robots.

La crítica fundamental que los dinamicistas hacen del paradigma tradicional es que no puede entenderse la percepción como un problema de reconstrucción del entorno dejando de lado la función esencial de transformación de una señal en un comportamiento [Nolfi and Floreano, 2000; Husbands, 1994]. Los agentes, en situaciones hostiles, no quieren reconstruir la señal que reciben del medio sino usarla para producir una acción adecuada.

Esta forma de entender la percepción en términos de cómo puede ser usada (para controlar la acción) constituye la base, dentro del paradigma dinamicista, de los modelos "percepción para la acción", o modelos "percepción-acción" [Beer, 1995]. Siguiendo esta idea, y mediante técnicas basadas en la teoría de sistemas dinámicos, varios autores han propuesto modelos de percepción-acción que han implementado y que abren la puerta a aplicaciones interesantes. La existencia de un modelo interno, el único rasgo que parecía ser genuino de los modelos clásico-simbólicos y que hacía posible el uso de tareas "off-line", ha sido igualmente modelado a partir de sistemas dinámicos con copias eferentes [Woergoetter and Porr, 2005], haciendo de los modelos dinámicos de la percepción una alternativa definitiva a los modelos clásicos [Clark and Grush, 1999].

## 2 Teoría de sistemas dinámicos y percepción

Como señalábamos al inicio, en la última década se ha venido consolidando un enfoque en la investigación sobre las capacidades cognitivas que muestra un importante interés por el uso de la teoría de sistemas dinámicos frente a herramientas computa-

cionales/simbólicas. En este contexto, la percepción se entiende como el resultado de las interacciones entre dos elementos: por un lado el agente y sus sensores y, por otro lado, las fuerzas externas del entorno. La función del sistema perceptivo sería la de organizar la información externa en forma de comportamiento y las percepciones serían los patrones auto-organizados que emergen en esa interacción dinámica [Kelso, 1995].

La idea de la perspectiva dinamicista es relativamente simple: si las estructuras que configuran los sistemas perceptivos de seres vivos, simples o complejos, deben organizarse para resolver problemas relacionados con su comportamiento funcional para sobrevivir en entornos impredecibles y cambiantes, entonces estas acciones requieren actividad dinámica (implica tiempo/espacio) y no-lineal (requiere una rica estructura de respuestas no-proporcionales a los cambios).

Ambas cualidades son rasgos de los sistemas dinámicos que pueden modelarse mediante sistemas de ecuaciones diferenciales [Norton, 1995]:

$$\begin{aligned} \dot{x}_1 &= f_1(x_1,\ldots,x_n,t,u_1) \\ &\cdots\cdots\cdots\cdots \quad \text{donde} \quad x \in \Re^n, t \in \Re \;, u \in \Re^n, f \in \Re^n \\ \dot{x}_n &= f_n(x_1,\ldots,x_n,t,u_n) \end{aligned} \tag{1}$$

Se han aplicado los conocimientos y las herramientas de los sistemas dinámicos para estudiar la percepción fundamentalmente considerándola como un elemento del que dependen los patrones de comportamiento (en humanos/vertebrados/robots) para la adaptación a su entorno. Sin embargo, no existen demasiadas propuestas que hayan analizado, con un enfoque dinamicista, la percepción desde un régimen "micro", es decir, considerándola como un fenómeno de procesamiento de información administrado y distribuido por redes de neuronas.

A continuación consideraremos el tipo de ecuaciones dinámicas que modelan un sistema acoplado agente-mundo y que constituyen el marco para entender la percepción desde una perspectiva dinámica. Veremos las aplicaciones que se han desarrollado en este área. Posteriormente, introduciremos algunas cuestiones generales sobre redes neuronales y expondremos cómo, a partir de las propiedades dinámicas de los sistemas dinámicos no-lineales, puede ser interesante considerar la percepción también desde una perspectiva biológica como el resultado de la dinámica de activación en una red neuronal modelada como un sistema de ecuaciones diferenciales que simula la estructura perceptiva.

## 3 Modelo "macro": Percepción como fenómeno cognitivo

Existen numerosos ejemplos en la literatura donde la cognición solo puede ser explicada desde una perspectiva que entienda el fenómeno como parte de un sistema acoplado agente-mundo (por ejemplo, en problemas de control de movimiento [Schonerand, 1998], lenguaje [Elman, 1995], toma de decisiones [Townsend and Busemeyer, 1995], adaptación [Beer, 1995], etc.).

En estos casos se hace uso de la noción de "affordances" [Gibson, 1979] para la explicación dinámica de distintos fenómenos cognitivos. Las "affordances" pueden

ser definidas como "*las posibilidades para el uso, la intervención y la acción que un entorno físico ofrece a un agente concreto, a su estructura, capacidades y habilidades de intervención".* Son, por tanto, oportunidades para la acción en una situación particular. Gibson argumenta que en el caso de la percepción, el agente "extrae patrones de los datos sensoriales" que "resuenan" en el sistema perceptivo y son sintonizados. Estos patrones emergen durante la interacción del medio con el sistema perceptivo [Schoener et al., 1998] (por ejemplo, los procesos de sincronización perceptivo-motora en el movimiento).

Siguiendo esta interpretación, el esquema básico para modelar las capacidades perceptivas exige la consideración de un sistema "agente-entorno" formado por dos subsistemas dinámicos acoplados, *A* y *E*, ambos dependientes del tiempo de forma continua. En el sistema acoplado, consideramos a los parámetros de cada sistema como funciones de algunas de las variables de estado del otro. Representamos este acoplamiento con una "*función Sensor S"* del sistema agente que depende de variables del medio, y con una "*función Motora M"* del sistema entorno que depende de variables propias del agente:

$$\dot{x}_A = A(x_A; S(x_E); u_A) \qquad (2)$$
$$\dot{x}_E = E(x_E; M(x_A); u_E)$$

Usamos los términos *S(x) y M(x)* en un sentido amplio: S debe interpretarse como una función que representa todos los efectos que *E* provoca sobre *A*, ocurra o no a través de lo que normalmente consideramos un sensor. Del mismo modo, *M* representa el conjunto de efectos que *A* tiene sobre *E*, sean provocados o no, por mecanismos que normalmente conocemos como efectores.

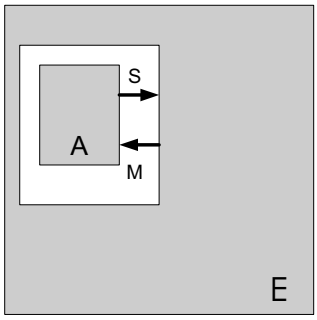



**Fig. 1.** Agente y entorno acoplados

¿Qué aporta este nuevo enfoque? La posibilidad de entender la percepción como un proceso diferente al de la "cognición pura" pudiendo ser interpretada como una estrategia de adaptación al entorno que busca economía computacional y eficiencia temporal explotando diferentes "trucos" en la relación entre la acción y el entorno local. Pasamos a ilustrar dos tipos de "estrategia" que solamente en un marco dinámico como el mostrado, pueden presentarse y modelar algunas de las características de los fenómenos de percepción.

En [Clark, 1997] el autor se refiere a la utilización de rasgos que no son identificativos de los objetos percibidos pero que son fáciles de detectar y usar, por ejemplo, el proceso de búsqueda visual de una taza a partir de su "color". Esta descripción sólo sería eficaz localmente (no se podría generalizar para ayudarnos a encontrar una taza en general) pero utilizaría características cuya detección es computacionálmente económica: el color se puede reconocer incluso en las periferias de baja resolución del campo visual.

Otros autores [Agre and Horswill, 1996] nos muestran como la percepción utiliza a menudo consultas repetidas al mundo exterior en lugar de contruir un modelo interno detallado. Adopta estrategias "sobre la marcha" y semiautomatizadas, rechazando procedimientos más intensivos que reflejaían conocimientos más profundos sobre los objetos.

La idea fundamental de estos mecanismos es que simplifican el tipo de representaciones para reducir posteriormente cargas computacionales y que sólo son posibles a partir de una interpretación dinámica entre el agente perceptivo y su mundo.

Otros autores como [Ballard, 1991] habla de representaciones 'just-in-time" mientras que, por ejemplo, [Brooks, 1991] ha acuñado el eslogan "el mundo es nuestro mejor modelo". Desde este punto de vista, la percepción se presenta como un mecanismo de procesamiento que auto-regula su actividad computacional (haciendo la tarea de selección de acciones más eficiente que mediante el uso de representaciones de escenas externas), en estrecha relación con el mundo y dependiente de las capacidades operativas del agente sobre el medio.

Los sistemas dinámicos se pueden interpretar como un estadio intermedio entre sistemas puramente reactivos (donde la salida motora del sistema queda determinada completamente por una entrada) y sistemas simbólicos (que pueden usar representaciones del medio para trabajar "off-line" sin necesidad de estímulos presentes): utilizarán sus variables internas y estímulos presentes en el medio para determinar combinadamente la salida motora (ver figura 2).

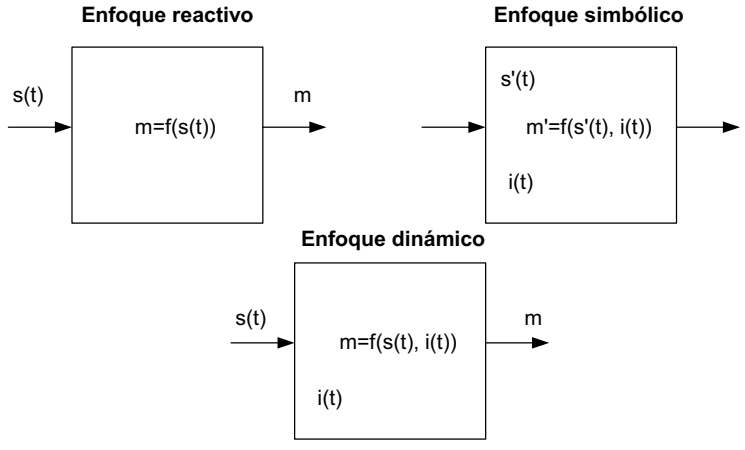



**Fig. 2.** Sistema reactivo, simbólico y dinámico

Desde el punto de vista práctico, nos interesa destacar el trabajo [Verschure, 2003] que ha logrado grandes progresos en el diseño de un robots utilizando un enfoque integrado en el contexto de la dinámica de sistemas. La robótica ha servido como campo de experimentación para observar y cuantificar la percepción (frente a mecanismos computacionales más restrictivos). El conocimiento adquirido por sus robots depende de los flujos de control que se dan en su arquitectura. El sistema que utilizan en sus trabajos se conoce como "Modelos de control adaptativo distribuido" y están basados en el acoplamiento dinámico entre capas reactivas, deliberativas y de control contextual.

## 4 Modelo "micro": Percepción como fenómeno neurobiológico

Hemos mostrado en la sección anterior qué cualidades presentan los modelos sobre la percepción al utilizar un punto de vista dinámico con respecto a la perspectiva tradicional: básicamente pasamos a entenderla como un elemento regulador del comportamiento adaptativo. En este artículo, sin embargo, queremos concentrarnos en las diferencias que se presentan en esquemas de más bajo nivel ¿Cuáles son las diferencias entre las redes neuronales analizándolas desde un enfoque dinámico con respecto a las redes neuronales clásicas?

Si observamos las RNA convencionales (por ejemplo, Redes de memoria asociativa [Hopfield, 1982] o redes de conexiones recurrentes [Elman, 1990]) comprobamos que presentan el mismo problema de representación interna que los modelos simbólicos clásicos. El mecanismo de computación neuronal clásico se basa en la creación de patrones estables bajo la actividad de un input sensorial y, sin embargo, las redes neuronales biológicas presentan una dinámica local y una evolución dinámica en las relaciones y conexiones. A pesar de las diferencias con el paradigma simbólico, el conexionismo no deja de ser una variedad del computacionalismo: los patrones clásicos son simbólico-lingüísticos y los conexionistas, patrones de activación distribuidos, pero ambas son representaciones discretas y, por tanto, independientes del contexto.

Durante varias décadas la mayoría de los biólogos, además de los neurocientíficos, han creído que los sistemas vivos (incluyendo el cerebro y sus propiedades) podrían ser entendidos mediante un enfoque reduccionista. Sin embargo, es posible observar fenómenos emergentes que surgen de la interacción entra los diferentes elementos de un estructuras neuronal y que sólo pueden ser entendidos desde un "punto de vista" integrado.

### 4.1 Redes neuronales como sistemas dinámicos

Las redes neuronales han sido estudiadas principalmente a partir de métodos estáticos que se reducen fundamentalmente al ajuste de los enlaces entre neuronas artificiales caracterizadas por una función de activación [Dorogovtsev *et al*., 2003]. El conocimiento reside en las conexiones que se establecen y la única dinámica que encontramos es la de la modificación de los valores asociados a cada enlace a partir de un

proceso de aprendizaje.

Sin embargo, muy poco se ha investigado aplicando otro tipo de herramientas matemáticas para abordar el estudio de la dinámica de las redes que simulan el comportamiento de los procesos neurobiológicos.

Creemos que debe profundizarse en la perspectiva dinamicista y en la consideración de una red como un conjunto de ecuaciones diferenciales que constituyen un sistema dinámico. Planteamos en esta y en la siguiente sección que algunas de las propiedades y resultados de la teoría de sistemas dinámicos no-lineales tienen una aplicación directa en el contexto de la neurocomputación y en la modelización de redes neuronales biológicas.

Durante más de dos décadas se han obtenido importantes resultados en el estudio de sistemas dinámicos, tanto teóricos [Katok and Hasselblatt, 1995] como numéricos [Kaneko, 1993] en estudios de multiestabilidad, caos, clasificación de patrones, etc. Sin embargo, para el caso de estructuras acopladas no-lineales, tenemos muy escasos resultados, salvo los obtenidos con métodos tradicionales y estáticos.

Un modelo dinámico del sistema nervioso deberá ser un proceso auto-organizado limitado por los requerimientos de su estructura y las fuerzas del entorno. Simplemente necesitamos tener bien caracterizadas dos componentes: por un lado, la *conectividad funci*onal (capacidad excitadora o inhibidora) junto a la *fortaleza de las conexiones,* y por otro lado, la *conectividad topológica*, que determina "qué neurona habla con quién" (Figura 3).

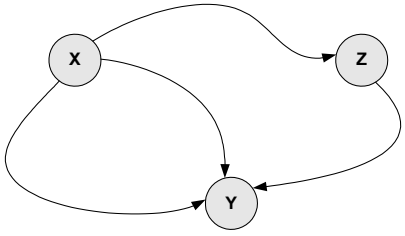



**Fig. 3.** Conectividad topológica de una red

¿Cuál sería el sistema de ecuaciones diferenciales más simple que nos permitiese modelar una red neuronal? El sistema más general, asumiendo una red neuronal de *N* elementos vendría representado por un sistema de ecuaciones *i:1,...N* del tipo,

$$\dot{x}_i = F(x_i) + \sum_{j=1}^{N} g_{ij}(x_i; x_j) + S_i \qquad (3)$$

donde $x_i$ representaría cada una de las neuronas, $g_{ij}$ sería la matriz que describiría los procesos de inhibición/excitación, $F(x_i)$ una función no-lineal que caracterizaría la dinámica de cada unidad de la red, y, por ultimo $S_i$ la contribución de un estímulo externo S a la neurona $x_i$.

La diferencia fundamental con respecto a una red neuronal clásica es que, para este modelo de computación, dado un estímulo, el sistema distribuye la información reci-

bida en el espacio (a lo largo de la red de neuronas) y en el tiempo (a lo largo de su dinámica) mediante la explotación de la dinámica espacio temporal de las redes no-lineales. La formación de clusters (correlación y desclusterización) se produce continuamente y es éste proceso el que la red utiliza para procesar la información que recibe del entorno.

Es sobre esta perspectiva sobre la que queremos profundizar con más detalle. Sólo recientemente en el contexto de la neurociencia computacional y a partir de los resultados del estudio dinámico sobre la percepción mostrado en la sección anterior, se ha empezado a mostrar interés por un nuevo tipo de sistemas de percepción inspirados en los principios básicos de "auto-organización" de sistemas dinámicos no-lineales.

La percepción se llevaría a cabo mediante sensores que procesarían señales distribuidas en el espacio con relaciones temporales dinámicas no lineales entre ellas, y el razonamiento (procesamiento) sería el resultado de un proceso de formación de patrones espacio-temporales, generados a partir de la información recibida por los sensores y que influiría directamente en el comportamiento del motor asociado.

Existen algunas propuestas computacionales de carácter dinámico, que muestran la manera en que una red neuronal simple, de carácter dinámico y recurrente, puede resolver de manera satisfactoria un problema adaptativo de elección simple. Pero, en general, los modelos actuales aún no reflejan la complejidad y heterogeneidad de muchos de los resultados empíricos que se conocen.

Destacamos, por sus aportaciones tanto experimentales como teóricas, el trrabajo de [Freeman, 2000; Freeman, 2004] sobre el sistema olfativo de los mamíferos. La importancia del trabajo de Freeman es doble, en la medida en que su modelo responde a un estudio experimental que permitió un conocimiento detallado del córtex olfativo y la existencia de un régimen de generación, coordinación y sincronización de secuencias neuronales emergentes como resultado de la interacción dinámica entre ellas. El modelo no sólo ofrece una aproximación detallada del sistema original sino que la interpretación de Freeman ha tenido grandes implicaciones para la comprensión de los orígenes neuronales de la percepción. Adicionalmente, su equipo ha diseñado y construido una red artificial que reproduce esencialmente las tesis teóricas dando al estudio mayor fortaleza.

### 4.2 ¿Qué se necesita para diseñar modelos neurobiológicos de la percepción desde una perspectiva dinámica?

El propósito de esta sección es proporcionar un conjunto de resultados, obtenidos del estudio de sistemas acoplados no-lineales, para el estudio de redes como sistemas dinámicos. Para tener todos los ingredientes que nos permitan adentrarnos en el estudio de las redes neuronales desde esta nueva perspectiva, necesitamos enfrentarnos a tres problemas: (1) En primer lugar, caracterizar el comportamiento dinámico de cada elemento del sistema, de cada nodo de la red, (2) en segundo lugar, caracterizar la dinámica de la topología de la red (los cambios en los enlaces a lo largo del tiempo) a partir de una ecuación diferencial, y (3) finalmente, caracterizar la interacción entre la topología de la red y la naturaleza dinámica de los procesos que pretende modelar.

En la literatura sobre los proceso neurobiológicos que subyacen en tareas de percepción se pueden encontrar numeroso artículos sobre estructuras de percepción-acción pero muy pocas han desarrollado realmente modelos matemáticos y propuestas serias, sino más bien especulaciones y argumentos heurísticos.

Sin embargo ciertas características de los sistemas dinámicos no-lineales parecen muy prometedoras si nos proponemos utilizar este tipo de sistemas para modelar la actividad perceptiva de una red neurobiológica. Nos centraremos en algunos de los resultados de los estudios teóricos sobre sistemas de N elementos acoplados con relaciones de excitación-inhibición [Rabinovich, 2001]. Estos sistemas, que podrían ser la base de un modelo de red neuronal, presentan:

1. Gran capacidad de codificación, el espacio de codificación en forma de clusters les permiten el almacenamiento y la recuperación de gran cantidad de información (del orden de *factorial de N* patrones diferentes, siendo *N* el número de elementos del sistema) [Rabinovich, 2001].
2. Patrones adaptativos: La inestabilidad inherente a los movimientos no-lineales facilita la habilidad de los sistemas para la rápida adaptación (pasando de un patrón a otro si el entorno es alterado) [Afraimovich, 2001].
3. Reconocimiento sensible a estímulos similares y, simultáneamente, capacidad de categorización: los sistemas neuronales pueden ser sensibles (muy discriminativos) y robustos a las perturbaciones al mismo tiempo.

¿Por qué la evolución habría de seleccionar los fenómenos dinámicos no-lineales como la base de los patrones de comportamiento de los organismos vivos? Por características como las que se acaban de presentar: cualquier red neuronal, entendida como un sistema dinámico de neuronas acopladas que presentase tales propiedades, sería un candidato excelente para poder interpretar la percepción como un producto de la dinámica de redes neurobiológicos.

Una vez encontrado un modelo dinámico de red neuronal para la percepción ¿Cuál sería la crítica más importante que quedaría sin resolver en los modelos de emergencia dinámicos? La incapacidad de su uso para la predicción (al contrario que las propuestas clásicas con los modelos de representación internos).

Algunos autores han planteado un mecanismo que podría resolver esta deficiencia situando a la perspectiva dinámica en una posición privilegiada para sustituir al paradigma clásico de carácter simbólico como paradigma de referencia en la Inteligencia Artificial [Wolpert *et al*., 1995, Wolpert and Ghahramani, 2000].

Podemos resumirlo de la siguiente manera: para proporcionar un comportamiento suficientemente sofisticado necesitamos más bucles de realimentación que sólo aquellos resultantes del acoplamiento entre agente (o su sistema sensorial) y el mundo (o los estímulos que provienen de él). Será sobre estos bucles sobre los que, según autores como [Hesslow, 2002, Cruse, 2003] se definirán las capacidades para la predicción. La posibilidad de que por encima de una colección de circuitos establecidos se puedan crear copias eferentes [Clark and Grush, 1999] que abran varias vías simultáneas de realimentación, permitirá que se generen flujos de control alternativos sobre los mismos componentes del sistema, y harán posible formas de control más sofisticado.

Este concepto está actualmente jugando un importante rol en las explicaciones neurocientíficas para el control de la acción dependiente del contexto. Varios autores

han sugerido que este mecanismo permite a los sistemas dinámicos y a la hipótesis dinamicista la posibilidad de una "forma de modelo interno del entorno". En [Clark and Grush, 1999] se sugiere que las interacciones sensomotoras facilitan copias eferentes de estímulos externos a los circuitos internos que los procesan y que pueden ser usados "off-line" para "emular" las salidas de diferentes posibles acciones.

## 5 Conclusiones y perspectivas

Las entidades biológicas tienen, por tanto, habilidades computacionales que residen en su estructura y en su dinámica. El uso de métodos de análisis y modelización estática para el estudio de las redes neurobiológicas no proporcionan los requisitos mínimos para poder entender, ni siquiera cualitativamente, la gran capacidad de computación y procesamiento de estas estructuras neuronales.

Este artículo ha proporcionado un análisis teórico orientado al estudio de la percepción-acción. El propósito ha sido mostrar cómo la noción de procesos perceptivos dinámicos permiten la generación de “representaciones” de la acción relevante dada la información que proviene del medio.

En concreto, hemos expuesto resultados teóricos que parecen sugerirnos que el uso de redes neuronales acopladas, modeladas como sistemas de ecuaciones diferenciales, pueden ser la base para una gran codificación de estímulos, representando el estímulo en forma de patrón espcio-temporal de actividad con propiedades como:

1. Robustez: estabilidad estructural de los patrones espacio-temporales generados frente al ruido.
2. Adaptabilidad: habilidad de la red para compensar cambios en las condiciones internas o externas y generar una salida apropiada.
3. Flexibilidad: habilidad para producir patrones espacio-temporales en respuesta a distintas señales externas.
4. Sincronización: capacidad de coordinación de patrones espacio-temporales en un conjunto de circuitos neuronales.

La idea que puede concluirse de los resultados de la aplicación de sistemas dinámicos al estudio de la percepción es que ésta puede ser entendida como un proceso de transformación de información con las siguientes características.

- Los procesos perceptivos adaptan sus respuestas hacia estímulos relevantes
- Existen conexiones múltiples y en paralelo entre sensores y efectores.
- Los sistemas dinámicos se consolidan como un marco de explicación alternativo a la computación clásica.
- Las arquitecturas neuronales de inspiración dinámica presentan un comportamiento inherente que es modulado por el contexto configurando su capacidad para procesar y codificar estímulos.

Consideramos que el enfoque dinámico para el estudio de la percepción y el posterior diseño de arquitecturas artificiales demuestra ser una alternativa sugerente a los modelos tradicionales. Nuestros proyectos actuales apuestan por el uso de propuestas dinámicas en los modelos de sistemas inteligentes que se encuentran en desarrollo en el contexto de las diferentes investigaciones de nuestro grupo. Creemos que los resul-

tados de diferentres análisis convergerán, sin duda, en nuevas herramientas teóricas que nos acercarán a un conocimiento más profundo sobre el funcionamiento de nuestras capacidades cognitivas y al diseño de artefactos que emularán de manera más precisa los recursos y las ventajas que caracterizan las redes neuronales responsables del comportamiento y la adaptación de los seres vivos.

Esperamos que estos sistemas presenten apropiados grados de autonomía y que también aprendan a partir de su interacción. Existen posibilidades prácticas claras en todas las aplicaciones donde se requieran sistemas autónomos o semi-autónomos, en campos como la inspección, monitorización o el control de sistemas complejos.

## Referencias

[Afraimovich *et al*. 2004] Afraimovich, V.S., M.I. Rabinovich, and P. Varona, Heteroclinic contours in neural ensembles and the winnerless competition principle. International Journal of Bifurcation and Chaos, 2004. 14: p. 1195-1208.

[Agree, 1996]. Agre, P.E. and Horswill, I. (1996). Lifeworld Analysis. Journal of Artificial Intelligence Research, No. 6.

[Ballard, 1991] Ballard, D. (1991). Animate vision. Artificial Intelligence, 48:57-86.

[Beer, 1995] Beer, R. (1995). A dynamical systems perspective on agent-environment interaction.Articial Intelligence, pages 173-215.

[Brooks, 1991] Brooks, R. (1991). Intelligence without representation. Artificial Intelligence, 47:139-159.

[Clark, 1997]. Clark, A. (1997). Being there. Putting brain, body and world together again. MIT Press.

[Clark and Grush, 1999] Clark, A. and Grush, R. (1999). Towards a cognitive robotics. Adaptive Behavior, 7:5-16.

[Craik, 1943] Craik, K. (1943). The nature of explanation. Cambridge University Press.

[Cruse, 2003] Cruse, H. (2003). The evolution of cognition - a hypothesis. Cognitive Science, 27:135-155.

[Dorogovtsev, 2003] Dorogovtsev, S. and Mendes, J.(2003). Evolution of Networks. From Biological Nets to the Internet. Oxford Univ. Press, Oxford.

[Elman, 1990] Elman, J. (1990). Finding structure in time. Cognitive Science, 14(2):179-211.

[Elman, 1995]. Elman, J. (1995), Language as dynamical system, in: R.F. Port, T. Van Gelder (Eds.), Exploration in the Dynamics of Cognition: Mind as Motion, MIT Press, 1995, pp. 195–225.

[Freeman, 2000] Freeman, W. J. Characteristics of the synchronization of brain activity imposed by finite conduction velocities of axons Int. J. Bifurcation Chaos 10 (10), 2307-2322 (2000)

[Freeman, 2004]. How and why brains create meaning from sensory information. Int. J. Bifurcation Chaos 14 (2), 515-530 (2004)

[Gibson, 1979] Gibson, J. (1979). The ecological approach to visual perception. Boston: Houghton Mi□in.

[Grush, 2003] Grush, R. (2003). In defense of some 'cartesian' assumptions concerning the brain and its operations. Biology and Philosophy, 18:53-93.

[Hesslow, 2002] Hesslow, G. (2002). Conscious thought as simulation of behaviour and perception. Trends in Cognitive Sciences, 6(6):242-247.

[Hopfield, 1982]. Hopfield, J. (1982). Neural networks and physical systems with emergent collective computational abilities. Proceedings of the National Academy of Sciences of the

USA, 79:2554-8.
[Husbands *et al*., 1994] Husbands, P., Harvey, I., and Miller, G. (1994). The use of genetic algorithms for the development of sensorimotor control systems. In Proceedings of the PerAc'94 Conference. IEEE Computer Society Press.
[Kaneko, K.,1993]. Kaneko, K. (ed.) (1993). Theory and Applications of Coupled Map Lattices. Wiley, London.].
[Katok, A. and Hasselblatt, B.,1995]. Katok, A. and Hasselblatt, B. (1995). Introduction to the Modern Theory of Dynamical Systems. Enciclopedia of Mathematics and its Applications, vol.54. Cambridge University Press.]
[Keijzer, 2002] Keijzer, F. (2002). Representation in dynamical and embodied cognition. Cognitive Systems Research, 3:275-288.
[Kelso, 1995] Kelso, J. (1995). Dynamic patterns: the self-organisation of brain and behaviour. Cambridge, MA: MIT Press.
[Nolfi and Floreano, 2000] Nolfi, S. and Floreano, D. (2000). Evolutionary Robotics. The Biology, Intelligence, and Technology of Self-organizing Machines. MIT Press.
[Norton, 1995]. Norton, A. (1995). Dynamics: An Introduction. In Port, R. and Van Gelder, T. (Ed.). Mind as Motion: Explorations in the Dynamics of Cognition. Cambridge University Press.
[Rabinovich, 2000]. Rabinovich, M. I., Varona, P. & Abarbanel, H. D. I. Nonlinear cooperative dynamics of living neurons. Int. J. .Bifurcation Chaos 10 (5), 913-933 (2000)
[Rabinovich et al. 2001] Rabinovich, M.I., et al., Dynamical encoding by networks of competing neuron groups: Winnerless competition. Physical Review Letters, 2001. 87 (6), (2001)
[Schoener et al., 1995] Sch□oner, G., Dose, M., and Engels, C. (1995). Dynamics of behaviour: theory and applications for autonomous robot architectures. Robotics and Autonomous Systems, 16:213-245.
[Schoener et al., 1998] Schoener, G., Dijkstra, T., and Jeka, J. (1998). Action-perception patterns emerge from coupling and adaptation. Ecological Psychology, 10:323-346.
[Townsend, J. Busemeyer,J., 1995] Townsend, J. Busemeyer,J.Dynamics representation of decision-making, in: R.F. Port, T. Van Gelder (Eds.), Exploration in the Dynamics of Cognition: Mind as Motion, MIT Press, 1995, pp. 101–120.
[Verschure, 2003] Verschure, P., Voegtlin, T. & Douglas, R. J. Environmentally mediated synergy between perception and behaviour in mobile robots. Nature 425, 620-624 (2003)
[Woergoetter and Porr, 2005] Woergoetter, F. and Porr, B. (2005). Temporal sequence learning, prediction and control. Neural Computation, 17:245-319.
[Wolpert et al., 1995] Wolpert, D., Ghahramani, Z., and Jordan, M. (1995). An internal model for sensorimotor integration. Science, 26:1880-1882.
[Wolpert and Ghahramani, 2000] Wolpert, D. and Ghahramani, Z. (2000). Computational principles of movement neuroscience. Nature Neuroscience, 3:1212-1217.

# Memoria y organoterapia

Juan Pablo Moltó Ripoll y Marcelino Llopis

Qimedica@hotmail.com

**Resumen.** La Homeopatía está reconocida como un instrumento para recuperar el equilibrio energético y químico del cuerpo cuando está afectado por una enfermedad. Aunque no se conoce muy bien como actúa la Homeopatía se está reconociendo que si que hay realmente una influencia notable sobre las células del organismo. En base a este principio nuestra intención es estudiar si ciertos preparados homeopáticos pueden tener influencia sobre ciertas actividades mentales. Dentro de las actividades mentales, vamos a empezar el estudio de la influencia de estos preparados sobre la memoria. La Organoterapia es la parte de la Homeopatía que prepara medicamentos en dosis homeopáticas a partir de ciertos extractos de tejido o sustancias orgánicas (sarcor). Dado que cuando se aplican estos medicamentos Organoterápicos se obtienen efectos demostrados sobre ciertas funciones orgánicas [1], proponemos analizar los efectos que tendrán sobre la memoria de los ratones, los preparados homeopáticos a partir de ciertos extractos de tejido orgánico (sarcor). Dado que las neuronas se caracterizan por su gran plasticidad, pensamos que la Organoterapia puede influir sobre algunas de sus funciones, y aquí proponemos es estudio de la influencia de la Organoterapia sobre la memoria de los ratones..

## 1 Introducción

### 1.1. Homeopatía y organoterapia

Dentro de las llamadas Medicinas Blandas la Homeopatía ha logrado el reconocimiento oficial en diversos países caracterizados por su rigor científico (Alemania, Inglaterra, EE.UU). La Homeopatía es un método terapéutico experimental que nació en 1760 y se desarrollo a partir de 1810 cuando Samuel Hahnemann publicó su libro Organon de la Medicina [2]. Los medicamentos homeopáticos parten de un principio distinto a la farmacología convencional. La farmacopea homeopática cuenta con más de 1500 cepas de origen vegetal, 1500 cepas de origen mineral o químico, y unas 200 de origen orgánico. Estas últimas son las que nos interesan en este trabajo [3]. Los principios de la Homeopatía son:

- Ley de semejanza [4]: existe cierta semejanza entre el poder tóxico de una sustancia y su poder terapéutico, dependiendo de su dosis. Algunas sustancias usadas en dosis ponderales (farmacopea convencional) produce síntomas patológicos en sujetos sanos. Si, en sujetos enfermos con esos mismos síntomas patológicos, usamos estas mismas sustancias podemos curar la afección manifestada por los mencionados síntomas. Este principio ya fue descrito por el médico danés George Ernst Stahl, en el siglo XVII.

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 475-481, 2006.

- Dosis infinitesimales: Los medicamentos Homeopáticos se preparan con a partir de diluciones muy bajas del principio activo. Para especificar el nivel de dilución se utiliza la nomenclatura CH o DH. Así una dilución 1CH se obtiene disolviendo íntimamente una gota del principio activo (tintura madre) en 99 gotas de diluyente (generalmente agua destilada). Una dilución 2CH se obtiene tomando añadiendo una gota de una dilución 1CH a 99 gotas de diluyente. Así sucesivamente se obtienen diluciones superiores.

Estas diluciones pueden ser tan elevadas, por ejemplo 24CH, que no tengan trazas químicas del principio activo inicial, y sin embargo, presentan notables efectos terapéuticos [5-6]. La Organoterapia sigue los principios de la Homeopatía obteniendo diluciones elevadas de tejidos orgánicos. Fue introducida en Europa por el Dr. Conan en Francia, y posteriormente desarrollada por Nevel, Martiny, Rouy G. Gomnod, Bergeret, y Tetaut desde 1936 [7]. La organoterapia se basa en la Ley trifásica: Las bajas diluciones (4CH) tonificaran la acción de un órgano, las diluciones moderadas (7CH) lo regularán y las diluciones altas (12CH) lo relajan o inhiben.

### 1.2. Sobre la memoria

El aprendizaje es el proceso por el cual es sistema nervioso adquiere nueva información, y la memoria es el almacenamiento y/o recuperación de esa información. La memoria según categorías cualitativas la podemos clasificar en:

- memoria declarativa que es el almacenamiento y recuperación del material disponible que hay en la mente consciente, como recordar un cumpleaños o la lista de la compra.
- memoria de procedimiento que no existe en la mente consciente e implica habilidades y asociaciones que son adquiridas y recuperadas a nivel inconsciente, como tocar un instrumento o montar en bici.

Otra clasificación de la memoria es según las categorías temporales,

- memoria inmediata: mantiene una experiencia durante unos segundos, en sentido visual, verbal o táctil).
- memoria a corto plazo: retiene en la mente la información durante un periodo de tiempo de minutos; por ejemplo recordar un número de teléfono. Aquí se englobaría la memoria de trabajo que permite mantener la información el tiempo necesario para llevar a cabo acciones secuenciales. Por ejemplo buscar un objeto perdido, te permite revisar lugares sin mirar en lugares donde antes habías mirado.
- memoria a largo plazo: retiene la información días, semanas o toda la vida. Implica la transferencia de información adquirida inicialmente a una estructura más permanente.

Hebb (1949) supuso que las bases fisiológicas son diferentes para la memoria a corto plazo y para la memoria a largo plazo. En la memoria de corto plazo el circuito de neuronas, al que el llamo “circuito reverbenante” se dispara según una pauta repetida, produciendo un trazo de memoria inestable y no causa un cambio en la estructura física del cerebro.

Para que esta experiencia o sensación pase de la memoria corto plazo a la memoria de largo plazo se requiere un verdadero cambio físico en la estructura del cerebro. Por ejemplo, en las autopsias se ha demostrado que las espinas dendríticas son más eleva-

das en cerebros con un CI alto que en cerebros con un CI mas bajo, y el numero de estas espinas dendríticas aumenta en correlación al aprendizaje (Purpura 1974, Crick 1982). Hay numerosas experiencias que avalan que el aprendizaje modifica la estructura neurológica del cerebro [8-15] (Van Harreveld y Fifkova 1975) (Sokolov 1977). Tomando como base estos resultados es importante conocer como se producen estos cambios en la estructura del cerebro, y como las moléculas de proteínas sirven como bloques de construcción celular se ha intentado investigar si las experiencias del aprendizaje producen cambios en la síntesis proteica.

## 2 Experimento inicial propuesto

**Descripción general**

Dado que el éxito para memorizar a largo plazo determinado acontecimiento radica en la capacidad de modificar la estructura de las células del cerebro, consideramos la posibilidad de que ciertos preparados Organoterápicos puedan estimular la plasticidad de las células neurológicas (espinas dendríticas) y con ello facilitar el registro de experiencias en la memoria y con ello el aprendizaje. Dado que las partes de la corteza cerebral que controlan la atención y la memoria no quedan totalmente mielinizadas hasta la edad adulta temprana, tendremos que considerar la edad de los sujetos del experimento.

Para desarrollar el experimento utilizaremos ratones a los que les suministraremos un producto organoterápico denominado Organolab Nervocen [16] a diversas diluciones suministradas en el alimento del día anterior al experimento. Para evaluar la capacidad de aprendizaje colocaremos en el final de un laberinto un cebo de 1 gramo de azúcar y observaremos el recorrido que realiza el ratón para llegar al final del laberinto. Repetiremos el experimento varias veces controlando el tiempo empleado por el ratón para conseguir su recompensa. El laberinto escogido para el experimento es el de Hampton Court como aparato experimental. El objetivo del expperimento es contrastar si la capacidad de aprendizaje de los ratones es influenciada por el suministro de preparados organoterápicos.

**Grupos de control**

El experimento se realizará sobre 4 grupos homogéneos de 20 ratones de raza “mus norvegicus albinus”, machos, de 1 mes de edad y temperamento similar. La asignación de los ratones a cada uno de los grupos se realizará aleatoriamente.

A cada grupo se les suministrará una dosis de Organolab Nervocen a diferentes concentraciones en la comida del día anterior (a las 21h de la noche anterior al experimento, para saciar el hambre del animal), según la tabla siguiente:

| Grupo 1 (control) | Grupo 2 | Grupo 3 | Grupo 4 |
|---|---|---|---|
| Sin medicamento | 4CH | 7CH | 12CH |

**Toma de datos**

Cada ratón efectuará el recorrido del laberinto 8 veces con un intervalo de una hora. El tiempo, en segundos, empleado por cada ratón será registrado en una tabla como la que se indica.

Primero situamos la rata en la zona de salida, y se mide el tiempo que tarda hasta conseguir llegar a la celda donde está el cebo. Este tiempo se registra en el cuestionario. Una vez la rata ha pasado el laberinto se le sitúa a la cola de sus compañeras, hasta el siguiente intento una hora después. Cada día se experimentará con 10 ratas de cada grupo. Es conveniente que haya varios laberintos iguales para evitar que los ratones puedan reconocer sus huellas anteriores en el laberinto.

A partir de los datos individuales, calcularemos para cada ratón el tiempo medio utilizado en todos los intentos, $\bar{x}$; el tiempo medio utilizado en los últimos tres intentos, $\bar{x}_f$; y el menor de los tiempos, $x_{min}$. Así mismo, para cada uno de los cuatro grupos se obtendrá el tiempo medio y desviación tipo para cada intento.

| Grupo | ratón | 1 | … | s | … | 8 | $\bar{x}$ | $\bar{x}_f$ | $x_{\min}$ |
|---|---|---|---|---|---|---|---|---|---|
| $i$ | 1 | $t_{i11}$ | … | $t_{i1s}$ | … | $t_{i18}$ | $\bar{x}_{i1}$ | $\bar{x}_{i1f}$ | $\bar{x}_{i1\min}$ |
| … | … | … | … | … | … | … | … | … | … |
| $i$ | j | $t_{ij1}$ | … | $t_{ijs}$ | … | $t_{ij8}$ | $\bar{x}_{ij}$ | $\bar{x}_{ijf}$ | $\bar{x}_{ij\min}$ |
| … | … | … | … | … | … | … | … | … | … |
| $i$ | n | $t_{in1}$ | … | $t_{ins}$ | … | $t_{in8}$ | $\bar{x}_{in}$ | $\bar{x}_{\inf}$ | $\bar{x}_{in\min}$ |
| $i$ | | $\bar{x}_{i1}$ | … | $\bar{x}_{i7}$ | … | $\bar{x}_{i8}$ | $\bar{\bar{x}}_i$ | $\bar{\bar{x}}_{if}$ | $\bar{\bar{x}}_{i\min}$ |
| $i$ | | $s_{i1}$ | … | $s_{i1}$ | … | $s_{i1}$ | $s_{i1}$ | $s_{i1}$ | $s_{i1}$ |

**Análisis de datos**

Una vez recogidos estos datos experimentales cabe hacer varios análisis diferentes:

1) estudiar para cada grupo la correlación entre los tiempos de cada uno de los intentos

   $Cov_{i,kl} = \dfrac{\sum_j \left(t_{ijk} - \bar{x}_{ik}\right)\left(t_{ijl} - \bar{x}_{il}\right)}{s_{ik}\ s_{il}}$, donde $i$ hace referencia al grupo, $j$ hace referencia al individuo, y $k,l$ hacen referencia a dos intentos diferentes.

   Un coeficiente de correlación elevado indicaría pautas de aprendizaje similares para todos los individuos del grupo.

2) analizar los gráficos de evolución de los tiempos para cada individuo con respecto a la media del mismo grupo, y comparar la evolución de las medias de cada uno de los grupos.

   Esta imagen nos puede dar una idea de cómo evoluciona el tiempo necesario para hacer el recorrido, y en base a esto proponer un modelo de regresión lineal que represente el proceso de aprendizaje.

De este modo se pueden comparar los tiempos de dos intentos consecutivos, estableciendo un coeficiente de aprendizaje, para el individuo $j$, del grupo $i$, entre los intentos $k$ y $k+1$, se tendría: $a_{ij,k} = \frac{t_{ij,k}}{t_{ij,k+1}}$; el coeficiente de aprendizaje medio del grupo $i$, entre los intentos $k$ y $k+1$, sería : $\overline{a}_{i,k} = \frac{\sum_j t_{ij,k}}{\sum_j t_{ij,k+1}}$.

Descontando efectos aleatorios, que cabe esperar más influyentes en los primeros intentos, la interpretación del valor de los coeficientes de aprendizaje es:

- $a_{ij,k} > 1$ hay un aprendizaje notable al reducirse significativamente el tiempo del recorrido.
- $a_{ij,k} \approx 1$ el tiempo se ha estabilizado, por lo que se considera que ya no se puede aprender más.
- $a_{ij,k} < 1$ hay una pérdida de motivación para realizar el recorrido.

El análisis de estos coeficientes revelará:

- Cuantos intentos son necesarios para que se estabilice el tiempo del recorrido, lo que indicará el mínimo número de intentos para el aprendizaje básico.
- A partir de cuantos intentos hay desmotivación.
- Durante cuantos intentos hay efectos aleatorios debido a los intentos de ensayo y error de los ratones para encontrar la salida del laberinto.

El número de intentos en este primer experimento debe de ser superior al estrictamente necesario para conseguir que el coeficiente de aprendizaje se estabilice en valores próximos a 1. De este modo se podrán distinguir dos fases una fase de aprendizaje básico y una segunda fase de refuerzo.

3) utilizar técnicas de análisis de la variancia para contrastar las hipótesis:
$H_0 = \{m_1 = m_2 = m_3 = m_4\}$ el tiempo de ejecución de la prueba es el mismo para todos los grupos,
$H_0 = \{$ "*alguna media es diferente*" $\}$
Para ello utilizaremos como valores representativos de la memoria de cada individuo $\overline{x}_{ij}$, $\overline{x}_{ijf}$ y $\overline{x}_{ij\min}$.

4) utilizar técnicas que comparen las observaciones de los grupos 2, 3 y 4 con el grupo 1 tomado como grupo de control (intervalo de confianza para observaciones pareadas) o mejor la pruebas de Tukey, de Duncan o de Dunnett [17]
También realizaremos estos análisis con $\overline{x}_{ij}$, $\overline{x}_{ijf}$ y $\overline{x}_{ij\min}$.

## 3 Experimento complementario

Se repetirá el experimento con 20 ratones del mismo tipo por cada tratamiento, y manteniendo las mismas condiciones que el experimento anterior, pero limitando el número de intentos al estrictamente necesario para alcanzar el aprendizaje básico.

De este modo, para cada tratamiento tenemos dos grupos de ratones uno que sólo ha realizado un aprendizaje básico y otro que además ha tenido un aprendizaje de refuerzo, con lo que tendremos 8 grupos de 20 ratones cada grupo.

| Aprendizaje | Grupo 1 | Grupo 2 | Grupo 3 | Grupo 4 |
|---|---|---|---|---|
| Básico | B1 | B2 | B3 | B4 |
| Con Refuerzo | R1 | R2 | R3 | R4 |

Los ratones de cada grupo y tratamiento se ordenan de mejor a peor tomando como criterio de ordenación alguno de los parámetros $\overline{x}_{ij}$ , $\overline{x}_{ijf}$ y $\overline{x}_{ij\,\min}$ .

Los ratones de cada uno de los grupos de la tabla anterior se dividen en dos subgrupos, *a* y *b*, de modo que ambos subgrupos sean semejantes en cuanto a la capacidad manifestada por los individuos. Es decir, el mejor iría a uno de los subgrupos, por ejemplo al subgrupo *a*, los dos siguientes al subgrupo *b*, los dos siguientes, de nuevo al subgrupo *a*, y así sucesivamente. El mejor individuo se asignaría alternativamente al subgrupo *a* y al subgrupo *b*. Cada subgrupo estaría integrado por 10 ratones. A los ratones de los grupos *a* se les daría una dosis diaria del medicamento organoterápico correspondiente, a los de los subgrupos *b* no se les suministraría medicamento. A todos los ratones se les colocaría una vez al día en el mismo laberinto con el correspondiente cebo como recompensa; se controlaría el tiempo empleado en hacer el recorrido. El experimento se repetiría durante 10 días y se analizarían los resultados para cada subgrupo siguiendo las directrices comentadas en el *experimento inicial*.

Para reforzar el análisis de los datos obtenidos en este último experimento se utilizaría una técnica de análisis de la variancia con tres factores, uno que representa el tipo de medicamento suministrado, el segundo factor que indica si el ratón ha tenido un aprendizaje reforzado con mayor número de intentos, y el tercer factor que indicaría si ha tenido dosis complementarias de medicamento organoterápico. Las conclusiones serían análogas a las que se pueden obtener en el experimento inicial, junto con las que se desprenden de la evaluación de la memoria a mayor plazo. Consideramos muy interesante la realización de este experimento para explorar las posibilidades que la homeopatía y la organoterapia pueden dar a la mejora de los procesos mentales, en particular, en este caso, a la memoria.

## Referencias

1. JOSE LUIS VAZQUEZ COLOMINA, (1995), Homeopatía y terapias afines, Hoedi homeopatía ediciones
2. DR. SAMUEL HAHNEMANN, (2002). Organon de la medicina, Editorial Porrua.
3. BOIRON (2003) Nomenclatura (3th ed.). autor.
4. DR PHILIPPE BELON, (1999), Investigación en homeopatía, Editions Borrón

5. ROBERTO MENDIOLA, (1999), Bases científicas de la medicina homeopática, Tomos I,II.
6. ENDLER, J. SCHULTE, (1997), Ultra High Dilution Physiology and Physics, edit; Kluwer Academic Publisher
7. Instituto Británico de Homeopatía (2003) temario Diplomatura homeopatía, tomos I,II,III. Autor
8. Dale Purves,George J, David F, Lawrence C, Kart, James O. (1987) Invitación a la neurociencia, editorial medica panamerica.
9. Diane E, Papalia ( 2001) desarrollo humano (8 ed) Mc Graw Hill.
10. Eric R Kandel, Thomas M. Jessell, James HSchwatr (2002) Neurociencia y conducta, Prentice hall.
11. Ardí Leahey,T. Jackson Harris, R (1997), Aprendizaje y cognioción (4 ed), prentice Hall.
12. John P.J. Pinel (2001) Biopsicología. (4ed), prentice hall.
13. Marcos Ruiz Rodriguez, (1999), Las caras de la memoria, Prentice Hall.
14. H.KLUWE, GERAD LÜE, FRANK RÖSER, BIRKHÄUSER VERLAG, (2003), Principles of learning and memory, Basilea
15. YADIN DUDAI,(2002), Memory from A to Z, Keybords, Concepts and beyond. Oxford University Press.
16. HOMEOLAB, (1999), Vademecum, Medicina ergocibernetica, edit, departamento técnico laboratorios homeolab.
17. WALPOLE, MYERS, MYERS. Probabilidad y Estadística. McGraw Hill. 1999
18. JUAN CARLOS AVILÉS, (2002). Prontuario de homeopatía y terapias biológicas, Edaff.
19. FONTES DE GRACIA, S. GARCÍA GALLEGO, C.GARRIGA, A,J.PEREZ-LLANTADA MªC. SARRIA,S.(2001), Diseños de Investigación en Psicología. UNED.
20. HAIR y otros; Análisis Multivariante; Prentice Hall, 1.999
21. MONTGOMERY, D.C.; Diseño y Análisis de Experimentos; Limusa Wiley, 2.004
22. C, AMENGUAL VICENS. J, ALEGRE VALLS. J, BAUR. M,A, SIERRA. G, RESCH. (1995),El medicamento homeopático, Phinter-Heel, Simile Homeoden

# De la neurociencia a la semántica: Percepción pura, cognición y modelos de estructuración de la memoria

Maite Fernández Urquiza

Universidad de Oviedo, Depto. Filología Española, Área de Lingüística General,
C/ Teniente Alfonso Martínez s/n, 33011 - Oviedo, Asturias, España
maitefu.uo@uniovi.es / maitefu@gmail.com

**Resumen.** El estudio de la interpretación de estímulos ostensivos no codificados de carácter visual requiere ocuparse de dos tipos de procesos que tradicionalmente han sido abordados desde disciplinas que emplean metodologías irreconciliables. Por un lado, es preciso describir el modo en que el cerebro procesa los inputs visuales para determinar hasta qué punto la estructura de la imagen es relevante en la interpretación del estímulo comunicativo. Por otro, habrá que examinar el papel que el conocimiento de alto nivel desempeña a la hora de dotar de sentido al mundo visual. Esta investigación pretende poner de manifiesto el carácter difuso de la frontera existente entre percepción pura y conocimiento, y destacar la importancia del estudio de la estructuración de la memoria humana desde la semántica empirista para completar nuestra comprensión del proceso interpretativo de la imagen, todo ello enmarcado en una perspectiva teórica relevantista y psicológicamente verosímil.

## 1 Introducción

El ser humano impone la característica de relevancia a los fenómenos externos en función de sus intereses y expectativas, además de en función de su conocimiento del mundo, y también, posiblemente, de algunas restricciones innatas.

Por otro lado, la percepción visual resulta paradigmática a la hora de ejemplificar el modo reflejo en que los mecanismos perceptivos humanos se enfrentan normalmente a una gran variedad de fenómenos ante los que actúan como filtro, procesando y seleccionando la mayor parte de la información a nivel subconsciente. Este filtrado libera al ser humano de la sobreestimulación caótica que se derivaría de la concesión del mismo rango informativo a todas las percepciones que cotidianamente lo asaltan.

Así pues, ciertas clases de acontecimientos (por ejemplo, ruidos fuertes, olores desagradables, movimientos violentos y repentinos) traspasan inmediatamente el filtro de atención selectiva y suscitan inferencias a nivel conceptual, lo que pone de manifiesto que el ser humano dispone de un mecanismo heurístico que maximiza su eficacia cognitiva en pro de la supervivencia, algo que sin duda constituye una ventaja evolutiva.

Sin embargo, existe otro tipo de información que procesamos de un modo que, en principio, podría parecer reflejo, dado que lo ejercemos de forma experta y eficaz, sin reparar en la importancia que la experiencia del mundo (el aprendizaje, al fin y al

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 482-487, 2006.

cabo) tiene en nuestra capacidad de realizar inferencias y razonamientos exitosos de forma consciente.

Dentro de este tipo de fenómenos se encuentran los estímulos comunicativos, es decir, aquellos fenómenos que cumplen la condición de haber sido designados por su emisor para que el receptor alcance determinados efectos cognitivos. En este trabajo nos ocuparemos de esta clase de fenómenos desde una perspectiva teórica relevantista, cuyo principio básico establece que un intercambio comunicativo se produce de forma óptima cuando la cantidad de información que un individuo es capaz de extraer de un estímulo que percibe como comunicativo es la mayor posible, y además está disponible con el mínimo coste de procesamiento requerido. La Teoría de la Relevancia proporciona un soporte coherente y psicológicamente verosímil para abordar el estudio de la conducta comunicativa humana, ya que sitúa la clave de su modelo cognitivo en una capacidad de inferencia espontánea, esto es, no estrictamente derivacional, que flexibiliza los modelos puramente formales del razonamiento humano, y que sitúa el desencadenante de tales procesos inferenciales en el reconocimiento, por parte del receptor, de la existencia de una intención comunicativa en el emisor.

## 2 Percepción y sistemas simbólicos

### 2.1. Estímulos ostensivos

Los fenómenos de tipo comunicativo son, en efecto, algo irrelevante si el destinatario no es capaz de reconocer su carácter ostensivo. En el caso de los enunciados lingüísticos esto se hace especialmente evidente, ya que de no ser tenida en cuenta la intención comunicativa del emisor, podrían ser considerados meros ruidos o simples marcas sobre el papel, sin significado alguno.

Sin embargo, lo que no podemos ignorar es que una parte importante de ese significado nos llega codificada, y que las características estructurales del signo lingüístico son, en gran medida, relevantes para completar un proceso exitoso de interpretación.

Ahora bien, existen también estímulos ostensivos no codificados como, por ejemplo, un movimiento corporal realizado de forma exagerada o un simple gesto. De lo que se trata es de que utilicemos con un fin comunicativo algo que, normalmente, no es utilizado con ese fin.

En este sentido, nos ocuparemos del estudio de la imagen como estímulo ostensivo no codificado (cuando se utiliza como soporte de fenómenos comunicativos de carácter publicitario y/o artístico) por cuanto que responde a la característica básica de hacer manifiesta una intención comunicativa a pesar de no responder a las leyes estructurales de un código explícito similar al lingüístico.

### 2.2. Sistemas simbólicos

Hasta el momento no hemos hecho sino proporcionar las coordenadas teóricas necesarias para que el propósito de nuestra investigación resulte comprensible. Sin embargo, no basta con apelar a nuestra capacidad de teorizar acerca de los estados men-

tales de nuestros congéneres y de atribuirles intenciones comunicativas para explicar en términos precisos el modo en que opera la mente humana a la hora de procesar un estímulo ostensivo de tipo visual. Es necesario, además, indagar acerca de la estructura de la imagen y del modo en que nuestro cerebro procesa la información visual que le llega a través de los módulos mentales de aducto.

Ocuparnos de la primera cuestión que acabamos de plantear supone hacerlo también de las características de los sistemas simbólicos. Como es bien sabido, los sistemas de símbolos constan de tres componentes básicos, a saber: 1) un conjunto de símbolos primitivos; 2) principios de ensamblaje de símbolos complejos a partir de símbolos primitivos, y 3) un método para relacionar los símbolos complejos con las entidades que simbolizan. El medio más simple para llevar a cabo esto último consiste en el emparejamiento unívoco y arbitrario de cada símbolo con un referente. Así funciona, por ejemplo, la notación alfabética que representa gráficamente algunas lenguas naturales humanas. Sin embargo, inmediatamente se nos plantea una subpregunta en este apartado, a saber: ¿Podemos manejar las imágenes en términos similares, es decir, es lícito hablar de un sistema simbólico visual? Nuestra respuesta es afirmativa, por las razones siguientes: 1) en primer lugar, porque toda imagen se compone de elementos primitivos susceptibles de descripción física; 2) luego, porque tales elementos son organizables en estructuras superiores por medio de varias fases perceptivas, y 3) finalmente, porque todo este proceso arroja una representación mental de tipo conceptual (simbólico) que se refiere al objeto físico percibido.

Ahora bien, si los símbolos complejos de tipo visual (imágenes) se ensamblan a partir de elementos primitivos de acuerdo con reglas estructurales, todavía debemos ser capaces de determinar dos cosas: 1) cuáles son esas reglas, lo que nos remite a la cuestión del modo en que nuestro cerebro procesa los inputs de tipo visual, y 2) si el modo de ensamblaje de los elementos primitivos y la estructura compleja resultante es relevante o no para su interpretación desde un enfoque comunicativo.

### 2.3. Estructura de la percepción y conocimiento

Comenzaremos, para responder a las cuestiones planteadas, por describir de modo esquemático qué es lo que sabemos que ocurre cuando ocurre algo tan habitual para nosotros como la percepción. Básicamente, la visión nos permite representar mentalmente el mundo a partir de patrones de luz. Sabemos que el cerebro está compuesto de células nerviosas que producen impulsos consistentes en cambios electroquímicos que se propagan a través de fibras nerviosas, sellando así la sinapsis entre un nervio y otro mediante otros procesos eléctricos o químicos. Así pues, en última instancia, podríamos decir que es este tipo de fenómenos el material primitivo del que se construyen los símbolos y, por tanto, de él emergerían también fenómenos mentales como la experiencia consciente y el procesamiento simbólico serial. Pero no explicaríamos gran cosa si no fuésemos un poco más allá para intentar salvar el abismo que en el plano de la investigación se abre entre anatomía y mente.

Las representaciones simbólicas procedentes de la percepción hacen explícita la información que necesitamos para desenvolvernos con seguridad en el mundo, y esto constituye una gran ventaja evolutiva. Sin embargo, esta información no se explicita

directamente en los patrones de luz que llegan a nuestra retina, sino que se recupera después de varios estadios de procesamiento visual. Así, por ejemplo, sabemos que el color es una característica prescindible a la hora de identificar objetos, y que lo verdaderamente importante en el procesamiento visual es la sensibilidad de las células retinianas a la intensidad lumínica. El contraste entre los puntos de mayor y menor intensidad lumínica de áreas contiguas es lo que parece permitir al ojo humano detectar elementos primitivos del tipo de barras, bordes, o manchas, es decir, localizar discontinuidades en las superficies físicas de las cosas de la escena. Existen modelos computacionales que describen esta fase perceptiva como una matriz de nivel de gris que haría explícita la intensidad de la luz en cada píxel de la imagen con respecto a una escala de valores arbitraria. En una segunda fase, el sistema visual detectaría los cambios de intensidad mediante el filtrado de la matriz para construir un esbozo primario, una representación organizada de las principales regiones de diferentes intensidades, a la que se llegaría mediante la agrupación de elementos similares para formar así líneas, puntos más grandes, y grupos estructurados. Posteriormente, el sistema visual ensamblaría la visión estereoscópica e identificaría el movimiento.

Ahora bien, lo que más nos interesa de todo esto es que se realiza sin que intervenga ningún tipo de esfuerzo consciente: nos movemos en el campo de la percepción pura, es decir, del conjunto de módulos visuales que utilizan un tipo de procesamiento que trabaja con información que podríamos llamar de bajo nivel. Se trata de un tipo de conocimiento implantado evolutivamente en el sistema nervioso y, por tanto, encapsulado en las computaciones de los módulos de nivel inferior. Es, por tanto, opaco a la introspección y al control consciente.

Dicho esto, hemos de enfrentarnos a un problema añadido, que es el siguiente: si la óptica determina el patrón que sigue la luz reflejada por las superficies mediante procedimientos matemáticos, a la visión le ocurre que debe afrontar la tarea de averiguar qué clase de objetos causaron los patrones de luz proyectados en la retina. Pero se trata de una labor que plantea demasiadas incógnitas, por cuanto que muchas disposiciones diferentes de objetos en una escena podrían dar lugar al mismo patrón de luz, dependiendo de variables como la intensidad lumínica, los ángulos de proyección, o las características de las superficies.

Sin embargo, capacidades humanas como el reconocimiento de formas tridimensionales a partir de contornos, y posiblemente también la estereoscopia, sugieren que la visión no depende únicamente de información innata de bajo nivel para llevar a cabo un procesamiento de abajo hacia arriba a partir de los inputs visuales. Por el contrario, llega un momento en que se hace necesario apelar al conocimiento de alto nivel para abordar el reconocimiento de objetos, escenas e imágenes que lleva a cabo nuestra mente. Dicho de otro modo: a pesar de toda la información que pueda haber en los patrones lumínicos que impresionan la retina, los procesos visuales dependen también de suposiciones acerca del mundo físico. La neuropsicología asume que la experiencia propia nos capacita para construir tal conocimiento y para emplearlo en procedimientos operativos que dotan de sentido al mundo visual. Lo que estamos diciendo es que todo apunta a la existencia de mecanismos mentales no encapsulados para recobrar las identidades de las cosas de una escena y aquellas propiedades de las mismas que la visión hace explícitas a la conciencia.

### 2.4. Consecuencias metodológicas

Realizar una distinción entre estos dos tipos de conocimiento pone de manifiesto el carácter endeble y difuso de la frontera que habitualmente suele trazarse entre la percepción visual pura, que se aborda como una cuestión de funcionamiento anatómico, y la cognición, considerada un fenómeno mental emergente de procesos fisiológicos cuya descripción no somos aún capaces de precisar por completo, lo que ha llevado a que ambas capacidades se aborden desde disciplinas que emplean, la mayor parte de las veces, metodología y terminología irreconciliables.

En cualquier caso, tanto el hecho de que exista esta dualidad epistemológica, como la necesidad de tender puentes interdisciplinares para intentar solventarla, arrojan las consecuencias metodológicas siguientes para nuestro estudio: 1) en primer lugar, nos permite aclarar el sentido en que la estructura del símbolo visual no es relevante para su interpretación desde un punto de vista comunicativo. En efecto, si la información que posibilita la construcción de un esbozo cuasi-tridimensional del mundo percibido está especificada en un módulo de arquitectura nerviosa relativamente fija pero, en cualquier caso, opaca a la conciencia, entonces no nos sirve para decir nada relevante acerca de cómo procesamos conscientemente este tipo de estímulos; 2) por otra parte, si la identificación de los objetos no puede producirse sin el uso de un conocimiento de alto nivel, imprescindible para generar representaciones de un mundo visual con sentido, esto orienta nuestro trabajo hacia el estudio de cuestiones semánticas y cognitivas relacionadas con la estructuración de la memoria para su aplicación a un análisis interpretativo de la imagen.

## 3 Semántica empirista

Apuntaremos, por tanto, muy brevemente, cuál es la línea que sigue nuestro trabajo actualmente. Para abordar el papel que la memoria desempeña en la interpretación de cualquier estímulo es necesario apelar a dos evidencias clave: 1) que el contenido semántico de los supuestos generados por tal estímulo afecta al razonamiento (lo que se manifiesta en el hecho de que los seres humanos nos negamos a deducir trivialidades, y en que pensamos de forma más adaptativa que lógica cuando, por ejemplo, resolvemos cotidianamente problemas que luego somos incapaces de afrontar si se nos plantean de forma abstracta), y 2) que las personas proceden de forma altamente estereotipada a la hora de procesar representaciones a las que se enfrentan con frecuencia, es decir, que infieren por defecto la mayor parte de las veces.

Lo anterior establece una relación directamente proporcional entre la frecuencia de procesamiento de los estímulos y la fuerza de confirmación de los supuestos que generan en nosotros, lo que condiciona el modo de estructuración y la accesibilidad de tales supuestos en nuestros patrones de memoria, así como de los esquemas conceptuales que los integran. Desde la lingüística podemos apoyarnos en teorías semánticas de corte empirista para explicar el modo en que esto ocurre. Tales teorías abarcan desde la noción de estereotipo desarrollada por H. Putnam, que se distancia de las mónadas no estructuradas de la semántica de prototipos para describir el significado de un concepto como una agrupación de rasgos típicos que son condición necesaria

de pertenencia a una clase, hasta la noción de concepto dinámico o <ad hoc> propuesta por R.Carston, según la cual el significado se construiría on-line, durante el proceso de interpretación del estímulo. Tal noción está basada en la que L.W.Barsalou desarrolla en el seno de su teoría de sistemas simbólicos perceptuales, la de <simulador>, que define como un ente psicológico componencial que se materializa, cada vez que un proceso interpretativo tiene lugar, en una conceptualización diferente de una misma categoría.

Creemos que el estudio en profundidad de los estereotipos conceptuales puede arrojar luz acerca de cómo el conocimiento de alto nivel opera en la interpretación de imágenes, y hacia este objetivo hemos enfocado una de las líneas principales de nuestra investigación actual.

## Referencias

1. Barsalou, L.W.: Perceptual Symbol Systems. En: Behavioral & Brain Sciences, 22 (1999) 577-660
2. Carston, R.: Metaphor, ad hoc concepts and word meaning – more questions than answers. En: Carston, R.: Thoughts & Utterances: The Pragmatics of Explicit Communication, Oxford, Blackwell (2002)
3. Hoffman, D. D.: Inteligencia visual. Cómo creamos lo que vemos, Barcelona, Paidós (2000)
4. Johnson-Laird, P.N.: El ordenador y la mente, Barcelona, Paidós (1990)
5. Marr, D.: Visión, Madrid, Alianza (1985)
6. Putnam, H.: El significado de significado. En: Valdés Villanueva, L.M., ed.: La búsqueda del significado, Madrid, Tecnos (1991) 131-194
7. Sperber, D. & Wilson, D.: La Relevancia. Comunicación y procesos cognitivos, Madrid, Visor (1994)
8. Varela, F., Thompson, E. y Rosch, E.: De cuerpo presente. Las ciencias cognitivas y la experiencia humana, Barcelona, Gedisa (1992)
9. Wilson, D. & Sperber, D.: Linguistic Form and Relevance. En: Lingua, 90 (1993) 1-25

# Clasificación de estímulos somatosensoriales basada en codificación temporal de la información

Juan Navarro[1], Eduardo Sánchez[2] y Antonio Canedo[1]

[1] Departamento de Fisioloxía, Facultade de Medicina,
Universidade de Santiago de Compostela,
{fsjna,fsancala}@usc.es
[2] Grupo de Sistemas Intelixentes (GSI)
Departamento de Electrónica e Computación, Facultade de Físicas,
Universidade de Santiago de Compostela,
15782 Santiago de Compostela, España
eduardos@usc.es

**Resumen** Este artículo explora la capacidad del código temporal generado por las neuronas de relevo de un modelo computacional realista del núcleo cuneatus en la vía de la sensibilidad somatosensorial para clasificar estímulos somestésicos. Para ello se realizan cuatro experimentos de clasificación con una red de aprendizaje *Perceptron* multicapa cuyo vector de entrada contiene la sucesión de estados de la capa de salida del modelo computacional del cuneatus. Los aciertos en cada tarea de clasificación varían con la dificultad del experimento pero en todos se observa un descenso del número de errores al aumentar la longitud del vector de entrada, a medida que la red de aprendizaje va tomando en cuenta un código especificamente elaborado por el modelo del núcleo. Ello permite concluir que el modelo computacional del cuneatus: 1) transmite la información necesaria para la clasificación, a lo largo del tiempo, y 2) ofrece una visión del cuneatus como un centro de procesamiento y codificación óptima de la información sensorial y no meramente un lugar de relevo o filtrado.

*Palabras clave:* Modelos computacionales del cerebro, reconocimiento de patrones, codificación temporal

## 1. Introducción

Nuestro interés se centra en el papel de la zona media del núcleo cuneatus, que recibe una entrada de fibras primarias que transmiten información táctil aferente procedente de receptores cutáneos localizados en el tronco superior y miembros anteriores en tetrápodos. Los tipos de células implicadas en la circuitería del cuneatus son: células de relevo o cuneotalámicas, interneuronas gabaérgicas e interneuronas glicinérgicas. Hasta el momento sólo se conoce la estructura del

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 488-499, 2006.

campo receptor de las cuneotalamicas, que consta de un centro excitador y una periferia inhibidora [3]. Esta disposición es generada a partir de una organización somatotópica de las aferencias, que contienen una entrada excitadora directa sobre las cuneotalámicas y una inhibición de áreas adyacentes a través de interneuronas gabaérgicas. Además hay evidencia experimental de que interneuronas gabaérgicas realizan una inhibición lateral recurrente desde campos receptores no contiguos y que interneuronas glicinérgicas producen una facilitación de las neuronas de relevo al inhibir a las interneuronas gabaérgicas [1]. Tras ser procesada por el núcleo cuneatus, la información sensorial pasa al núcleo lateral–posterior–ventral de tálamo antes de alcanzar la corteza somatosensorial primaria. El estudio de las primeras etapas de procesamiento en la vía somatosensorial, en animales superiores, se encuentra con dos graves problemas: por una parte la dificultad para registrar intracelularmente unidades en animales despiertos y activos y por otra, la dificultad para especificar y controlar los parámetros de estimulación con la exactitud que se consigue, por ejemplo, en la vía visual. Los modelos computacionales son un medio de eficacia conocida para sortear tales dificultades.

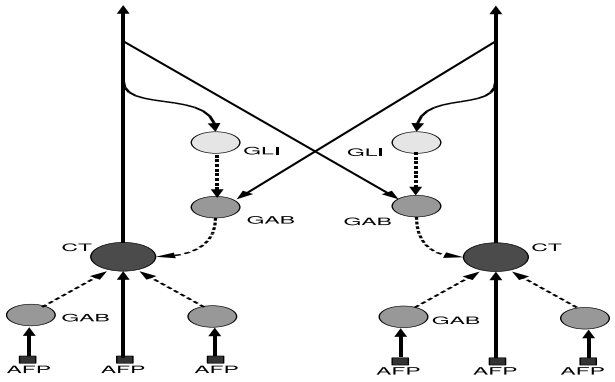



**Figura 1.** Modelo del núcleo cuneatus basado en hallazgos experimentales en el gato. AFP significa aferencias primarias, CT células cuneotalámicas, GAB interneuronas gabaérgicas y GLI interneuronas glicinérgicas. Las flechas indican el sentido en el flujo de información y las líneas discontinuas representan conexiones inhibidoras.

El modelo computacional del cuneatus, basado en hallazgos experimentales, propuesto por los autores con anterioridad [8,9] y utilizado en este trabajo, es una versión bidimensional del modelo que aparece en la figura 1, que especifica los tipos de neuronas, sus conexiones aferentes que proceden de receptores cutáneos que no adaptan y sus conexiones recurrentes. Las neuronas cuneotalámicas tiene un campo receptor aferente con un centro excitador y una periferia inhibidora y reciben una influencia recurrente inhibidora de interneuronas gabaérgicas. Estas interneuronas reciben una entrada excitadora de células cuneotalamicas

de campos no adyacentes. Finalmente las interneuronas glicinérgicas tienen un campo receptor que recibe entradas excitadoras de colaterales recurrentes de células cuneotalámicas de campos receptores próximos. Las neuronas del modelo son unidades MacCulloch–Pits en las cuales la salida de la j–ésima neurona es $y_j = \Psi \sum(w_{ji} \cdot x_i)$, donde $\Psi$ es una función tipo umbral. La contribución de la sinapsis entre la neurona i–ésima y j–ésima es modelada a través del peso $w_{ji}$. Los pesos que afectan a cada neurona forman una matriz que describe su campo receptor: con forma de sombrero mejicano para las cuneotalámicas, con forma de anillo para las recurrentes gabaérgicas y de meseta plana para las glicinérgicas. El modelo de neurona de las neuronas cuneotalámicas es realista, por lo que la condición de salida depende del potencial de membrana que es el resultado de la influencia de las entradas sinápticas y de la evolución de cuatro corrientes de membrana: una corriente de potasio y una corriente de sodio capaces de producir potenciales de acción; una corriente catiónica despolarizante que se activa por hiperpolarización y una corriente de calcio de bajo umbral activada por despolarización pero desinhibida por hiperpolarización. Existe respaldo experimental [2,6,7] para la inclusión de las citadas corrientes. La circuitería del modelo incluye un bucle recurrente sobre las neuronas cuneotalámicas por lo que la salida del modelo sufre transformaciones cada vez que se actualizan las salidas de todas las neuronas, es decir, en cada iteración. Si el estado de activación de las neuronas cuneotalámicas supera un umbral, estas transmiten potenciales de acción aislados o en ráfagas. La respuesta global o poblacional del modelo se obtiene a partir de una variable llamada *salida_global* cuyo valor se obtiene sumando el número de neuronas que transmiten un potencial de acción en esa iteración o el primero de una ráfaga.

En la figura 2 se muestran algunas características espaciotemporales de la información transmitida por el modelo en respuesta a dos estímulos de prueba que aparecen en los paneles A1 y A2. Los estímulos son dos cuadrados que contiene un patrón en forma de mosaico. Al que aparece en A1 se le ha aplicado cierto grado de desenfoque gausiano. En este trabajo se emplean otros estímulos con otros patrones, formas, tamaños y grados de desenfoque gausiano (o sin desenfoque como el del panel A2) que intentan reproducir algo de nuestra experiencia somatosensorial. Ante un estímulo estático presentado al modelo, la salida de las neuronas cuneotalámicas pasa por una sucesión de distintos estados, algunos de los cuales figuran, con el número de la iteración en que aparecen, a la derecha del estímulo, en los paneles A1 y A2. En la iteración 1 se transmite un mapa del estímulo, en la 7 un mapa de las zonas de mayor contraste espacial y en las otras iteraciones se aprecia una codificación progresiva de la superficie del estímulo empezando por las zonas de mayor contraste hacia las de menor contraste y que llamaremos "llenado". Nótese que las zonas de mayor contraste en el estímulo del panel A1 son los límites o perímetro del cuadrado, en cambio en el estímulo del panel A2, son las líneas de la retícula del mosaico. En el panel A1 el "llenado" es bastante evidente pero en A2 afecta a pequeños islotes, uno de los cuales ha sido marcado con flechas para facilitar su seguimiento. Cuando el "llenado" termina, el modelo alcanza un régimen estacionario con una oscilación residual en la que

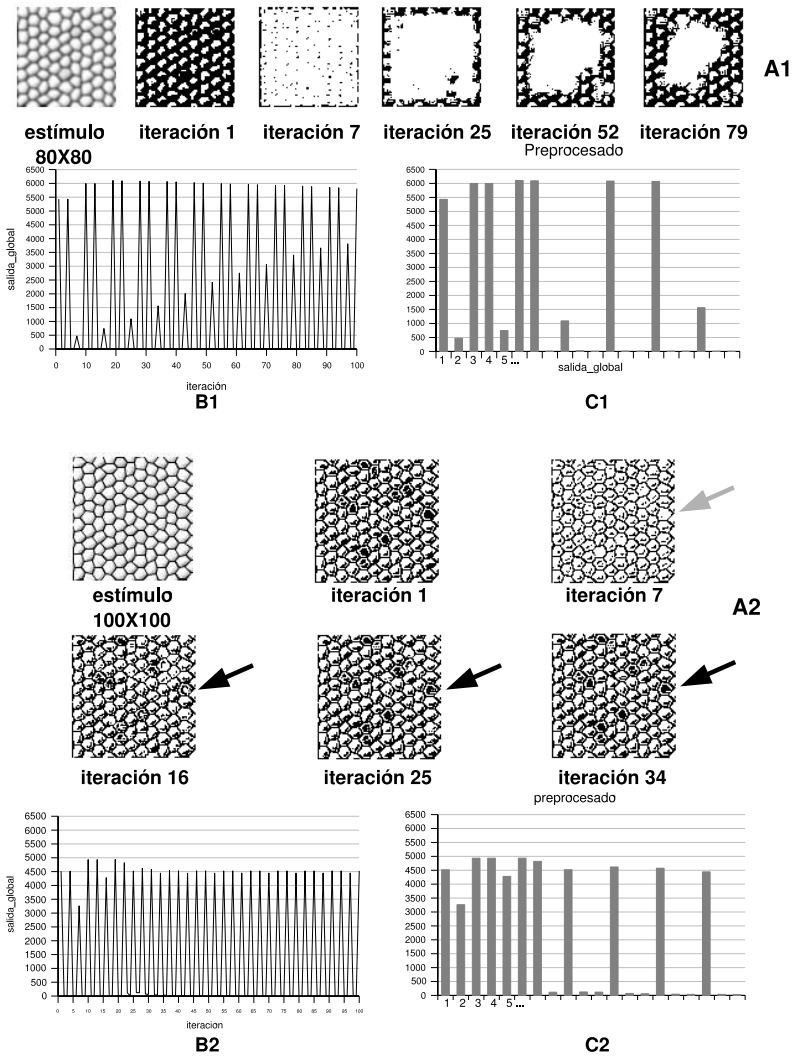



**Figura 2.** Los paneles A1 y A2 muestra sendos estímulos somestésicos, donde la intensidad de la presión está codificada en escala de grises. En los paneles A1 y A2 aparece también la salida correspondiente del modelo en varias iteraciones, donde cada punto negro representa una neurona cuneotalámica que transmite un potencial de acción en esa iteración. Los paneles B1 y B2 representan una serie temporal con la evolución de la variable *salida_global* a lo largo de las 100 primeras iteraciones. Los paneles C1 y C2 representa un conjunto de valores de *salida_global* extraídos de los paneles B1 y B2 tras eliminar ceros y repeticiones del primer valor. Las flechas en A2 apuntan a una de las pequeñas zonas donde puede seguirse el "llenado" de la codificación progresiva.

se reitera la transmisión de un mapa somatotópico del estímulo. Así pues, el modelo transmite una primera tanda de información con el "llenado" y después una tanda que repite de forma indefinida el mapa somatotópico, junto con otros estados. La duración de la primera tanda, el número de estados de la segunda y en general todas las particularidades de ambas, dependen de cada estímulo. Durante la primera tanda, cada neurona cuneotalámica transmite una información que depende de la activación de su campo receptor y de su distancia a las zonas de mayor contraste del estímulo, según se aprecia en el "llenado". En el panel B1 y B2 se aprecian distintas fases oscilatorias en la respuesta global a través de la evolución de la variable *salida_global* en las 100 primeras iteraciones. Es sabido [4,5] que tales oscilaciones en la respuesta poblacional están determinadas por las características geométricas, textura y desenfoque de las imágenes que representan a los estímulos.

El reconocimiento de formas y texturas es una función importante del sistema somatosensorial de animales superiores relacionada hasta ahora con la actividad de la corteza cerebral. El modelo descrito del cuneatus permite investigar la participación de estructuras inferiores en la vía somatosensorial en tal función de reconocimiento. En el presente trabajo se plantea la pregunta de si la respuesta global (a través de la variable *salida_global*) del modelo del cuneatus contiene suficiente información para la identificación o clasificación de los estímulos.

## 2. Métodos

Para responder a esa pregunta se proponen cuatro experimentos en los que se entrena una red de aprendizaje, tipo *Perceptron* multicapa, para que clasifique la respuesta global del modelo del cuneatus según el tamaño de los estímulos, el grado de desenfoque gausiano aplicado a las imágenes que los representan, su pertenencia a distintas formas (caracteres tipográficos) y su grado de segmentación. Las redes de aprendizaje permiten establecer una correspondencia entre grupos de datos y en este caso el intento se refiere a categorías de estímulos y al código elaborado por el modelo del cuneatus al procesarlos. Si tal correspondencia tiene "éxito" es porque el código contiene información sobre tales categorías. El "éxito" se cuantificará a través del porcentaje de errores en cada experimento. En la figura 3 se esquematiza el flujo de datos durante los cuatro experimentos tomando como ejemplo el de clasificación según tamaños. Los valores de *salida_global* cambian a lo largo de sucesivas iteraciones y constituyen una serie temporal. Dicha serie se preprocesa en todos los casos dos veces: la primera elimina los valores nulos y las repeticiones del primer valor y la segunda vez se *normaliza*. Los datos así preprocesados son los valores de entrada en la red *Perceptron* multicapa. Nótese que se trata de datos correspondientes a una serie temporal procedente del modelo computacional del cuneatus. Los paneles C1 y C2 de la figura 2 muestran dos ejemplos de la transformación que sufre la serie temporal tras el primer preprocesado. Este flujo se produce de forma supervisada durante la etapa de entrenamiento con el conjunto de estímulos de entrenamiento y para

estudiar la información contenida en la variable *salida_global*, durante la etapa de prueba con el conjunto de estímulos de prueba. Todo el proceso se repite para vectores de entrada de longitudes: 1, 2, 3, hasta 20 y se define y evalúa con un *script* de la librería de funciones para redes de MATLAB. Cada experimento se repite diez veces.

Nótese que la red de aprendizaje no se utiliza para clasificar o reconocer estímulos, ni para simular el comportamiento del sistema nervioso sino como una

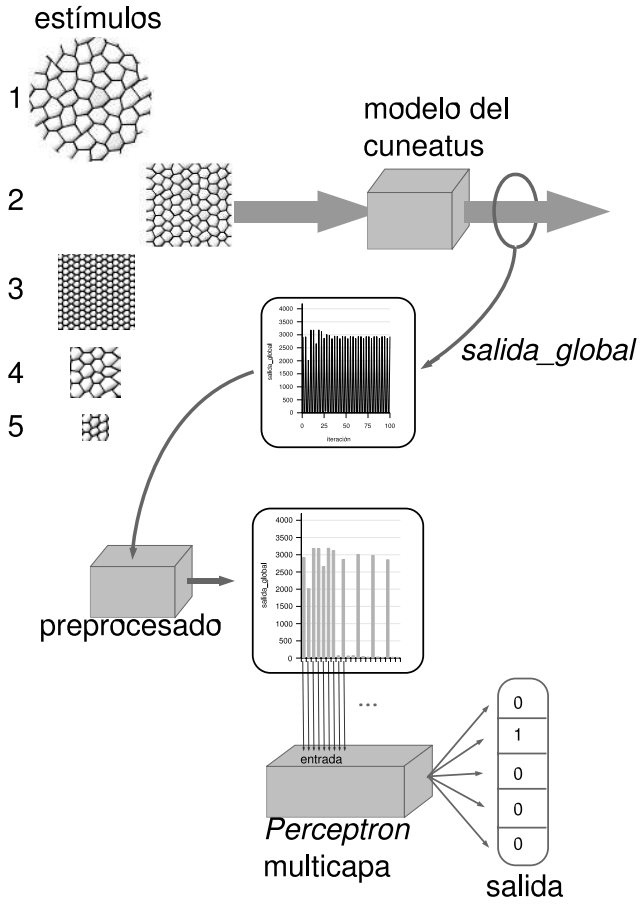



**Figura 3.** Las flechas indican el flujo de datos en los experimentos de clasificación, concretamente según el tamaño del estímulo. La serie temporal de *salida_global* es transformada para eliminar ceros y la repetición del primer valor. La serie resultante es el vector de entrada de la red de aprendizaje. Nótese que la coincidencia entre el estímulo del segundo tamaño y la posición del bit 1 en la salida de la red de aprendizaje constituye un acierto en la etapa de aprendizaje o en la de prueba.

erramienta para evaluar el código generado por nuestro modelo del cuneatus. La evaluación no solo se refiere a los valores aislados de la variable *salida_global* sino a la sucesión de su valores a lo largo del tiempo ya que el modelo transmite un código temporal. Por ello se prueba con varias longitudes del vector de entrada.

## 3. Resultados

El primer experimento utiliza 130 estímulos diferentes (65 de entrenamiento y

65 de prueba) de cinco tamaños y varias texturas rugosas, formas y grados de desenfoque (o sin desenfoque). La red debe identificar en la etapa de prueba cual de los cinco tamaños posibles es el del estímulo presentado. El panel A de la figura 4 contiene un ejemplo de cada uno de los cinco tamaños de estímulos presentados en la etapa de prueba y el panel B el número de errores en la identificación correcta de cada estímulo que desciende del 11.69 % al 1.69 %, cuando el vector de entrada a la red alcanza los tres datos.

El problema encargado a la red de aprendizaje es fácil pero no trivial. Véase por qué. La variable *salida_global* se definió como el número de neuronas cuneotalamicas que transmiten un potencial de acción en respuesta al estímulo. Cuántas más neuronas de relevo sufran el efecto del estímulo mayor será el valor de *salida_global*. Teniendo en cuenta que el primer valor de la serie temporal de

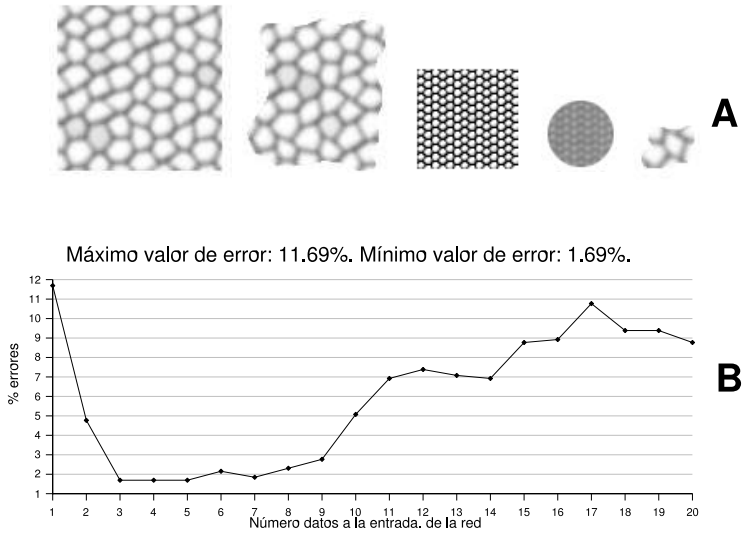



**Figura 4.** Ejemplos de varios estímulos y resumen de los resultados sobre los errores en la tarea de clasificación según tamaños.

*salida_global* contiene un mapa de la superficie estimulada, la red de aprendizaje debererría tener suficiente información con el primer valor de la serie. Con el primer valor de la serie el error es pequeño (11,69 %), pero en la figura 4 se ve que hay después errores menores, es decir el primer valor no es suficiente para que la red alcance su mejor rendimiento. Esto es así por dos razones. Primero porque a la red se le pide que categorice según tamaño, es decir la dimensión de los ejes del estímulo, no según superficie. La relación entre tamaño y superficie es constante sólo para el mismo tipo de figura geométrica. Dicho de otra forma: sabiendo la superficie se puede predecir el tamaño de los ejes sólo si se sabe el tipo de figura. El cálculo es distinto, por ejemplo, si la figura es un cuadrado o un círculo. Segundo, porque cada textura y cada desenfoque genera en el modelo distintos valores de *salida_global* por unidad de área. Así pues, sería necesario que todos los estímulos tuviesen la misma textura, desenfoque y la misma forma geométrica para que la red pudiese hacer una predicción sin errores del tamaño, sólo a partir del primer dato de la serie temporal de *salida_global*. Como ya se ha dicho, ese no es el caso: se han utilizado estímulos de distinto tamaño, pero también de distinta forma, textura y desenfoque. Por eso la red mejora su rendimiento, casi siete veces, si además se le suministran datos sobre las zonas de mayor contraste y al menos, el inicio de la codificación progresiva de la superficie del estímulo. En la figura 2 se muestra un ejemplo de como un estímulo mayor da un valor inicial de *salida_global* menor (4516) que el que da un estímulo mas pequeño (5429), sólo porque tiene mas contrate al estar menos desenfocado. Pueden compararse ambos valores en el eje vertical de los paneles B1 y B2 de ambas figuras. El segundo dato del vector de entrada se corresponde con el

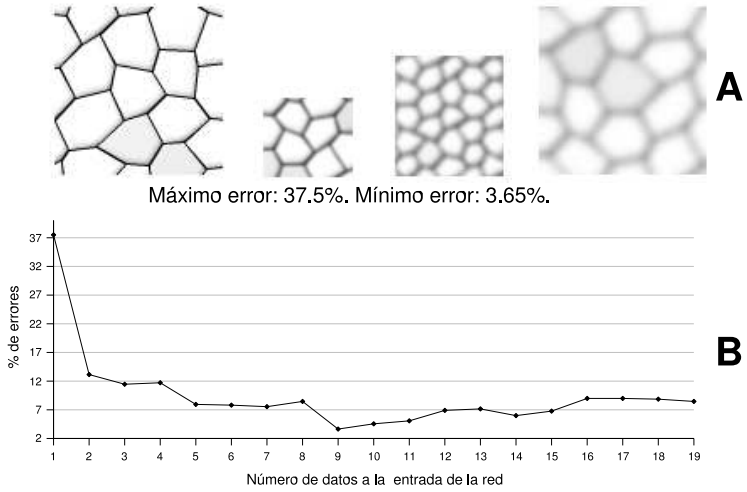



**Figura 5.** Ejemplos de varios estímulos y resumen de los resultados sobre los errores en la tarea de clasificación según contrastes.

valor de *salida_global* en la séptima iteración. Tal valor es el conteo de unidades que transmiten un potencial de acción en las zonas de mayor contraste. Para el estímulo menor, en esta pareja de ejemplos, ese valor es de 473 (un valor próximo al perímetro del cuadrado: $80X4 = 320$). En cambio, para el estímulo mayor ese valor de 3260 incluye muchos segmentos de la trama hexagonal (un valor muy distinto del perímetro del estímulo: $100X4 = 400$). Pueden apreciarse las diferencias comparando la séptima iteración en los paneles A1 y A2 de la figura 2. De alguna forma, la red de aprendizaje va utilizando esos datos elaborados por el modelo del cuneatus para su entrenamiento en la tarea de clasificar cada estímulo, según su tamaño. Tras el entrenamiento, la red de aprendizaje acierta clasificando adecuadamente ambos estímulos si cuenta con un vector de entrada de 3, 4 ó 5 elementos. Esos elementos recogen la oscilación inicial que la codificación progresiva impone a los valores de *salida_global*, así que esa propiedad oscilatoria de la respuesta global del modelo no es una mera curiosidad, sino un elemento fundamental en su capacidad de identificar estímulos, en este experimento y como se verá, en los tres siguientes.

El segundo experimento utiliza 128 estímulos diferentes (64 de entrenamiento y 64 prueba) de tres grados de desenfoque gausiano (2X2, 4X4 y 8X8) y sin desenfoque, pero con varios tamaños y texturas. La red debe identificar cual de los cuatro grados de desenfoque gausiano es el que afecta al estímulo presentado. El panel A de la figura 5 contiene un ejemplo de los 4 tipos de estímulos

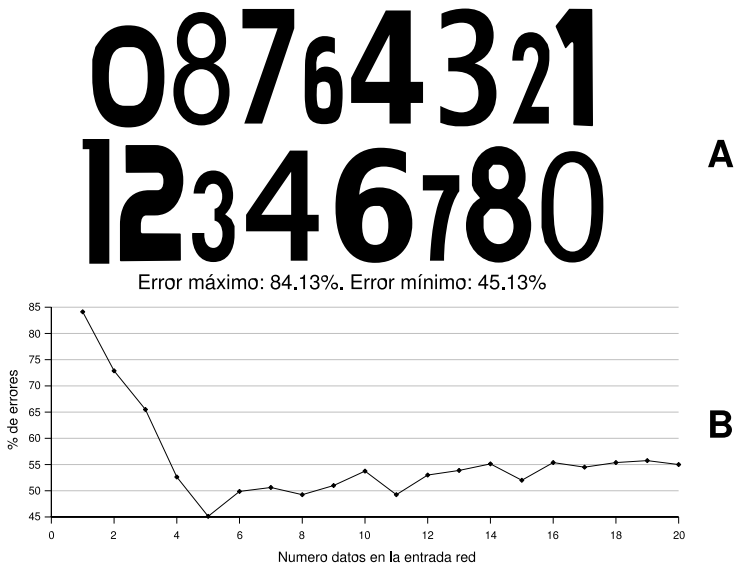



**Figura 6.** Ejemplos de varios estímulos y resumen de los resultados sobre los errores en la tarea de clasificación según formas.

presentados en la etapa de prueba (de izquierda a derecha: sin desenfoque, con desenfoque 2X2, 4X4 y 8X8) y panel B el número de errores en la identificación correcta de cada estímulo en el experimento desenfoque. Los errores pasan del 37.5 % al 3.65 % en los 9 primeros datos del vector de entrada. En este caso y el anterior el valor del error sufre un descenso casi de un orden de magnitud al alcanzarse cierto número de iteraciones.

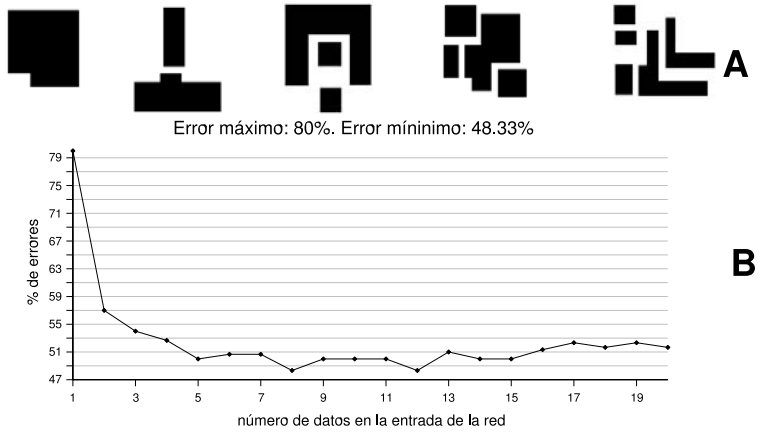



**Figura 7.** Ejemplos de varios estímulos y resumen de los resultados sobre los errores en la tarea de clasificación según la segmentación de los estímulos.

En el tercer experimento se presentan al modelo 180 estímulos (90 de entrenamiento y 90 de prueba) que son los caracteres numéricos 1, 2, 3, 4, 6, 7, 8 y 0 escritos en 20 variedades tipográficas distintas. En la etapa de prueba la red debe identificar cual de los ocho caracteres es el que se presenta. El panel A de la figura 6 contiene dos ejemplos (dos variedades tipográficas) de cada uno de los ocho estímulos presentados en la etapa de prueba y panel B el número de errores en la identificación correcta de cada estímulo que desciende, en la etapa de prueba, del 84.13 % al 45.13 % en los cinco primeros datos del vector de entrada.

En el último experimento se plantea al modelo un problema difícil: el reconocimiento del número de piezas o segmentos de los que consta el estímulo. Para ello se presentan 100 estímulos (50 de entrenamiento y 50 de prueba) en los que aparecen 1, 2, 3, 4 ó 5 piezas o segmentos de distintas formas y tamaños, pero con una superficie total parecida para que eso no resulte una pista en la clasificación. La superficie de todas las piezas o segmentos es homogénea y no se aplica desenfoque gausiano. La red debe identificar en la etapa de prueba cuantos segmentos contiene el estímulo. El panel A de la figura 7 contiene un ejemplo

de cada uno de los cinco tipos de estímulos presentados en la etapa de prueba y panel B el número de errores en la identificación correcta de cada estímulo que desciende del 80 % al 48.33 % en los ocho primeros datos del vector de entrada. En este experimento y el anterior cada estímulo genera una configuración de superficie total, perímetro y extensión del "llenado" que puede servir a la red para clasificar, teniendo en cuenta que la red va mejorando sus resultados a medida que va completando la recepción de dicho "llenado". Nótese que en estos dos últimos experimentos el error máximo, que es el valor inicial de *salida_global* que contiene sólo un mapa somatotópico del estímulo, es muy parecido al error cometido al hacer una elección aleatoria (87.5 % y 80 % respectivamente) en la tarea de clasificación.

## 4. Discusión

Los resultados en los cuatro experimentos son acordes con la dificultad que entraña cada uno de ellos, pero en todos el error disminuye desde un valor inicial hasta un mínimo ligado a un número limitado de iteraciones. Los datos contenidos en la primera iteración, que transmite un mapa somatotópico del estímulo, dan en todos los casos el mayor porcentaje de errores. El aumento en el número de aciertos se produce durante una etapa en la que el modelo transmite, siempre en el mismo orden: un mapa de las zonas de mayor contraste y una codificación progresiva de la superficie del estímulo. Estos dos componentes son decisivos en la reducción de errores en el reconocimiento y clasificación de estímulos y son precisamente estos dos componentes el resultado del procesamiento efectuado por la circuitería y las corrientes iónicas de las neuronas cuneotalámicas del modelo. Los aciertos aumentan cuando la red de aprendizaje toma en cuenta un código especificamente elaborado por el modelo del núcleo. El modelo pues, predice que el núcleo cuneatus, aparte de transmitir un mapa somatotópico de los estímulos, crea un canal de información, a partir de las oscilaciones en su respuesta global, que contiene durante un tiempo limitado, información para identificar estímulos, notable en algunos casos, según sus características geométricas o cualitativas (como su grado de desenfoque). Cada forma o textura en los estímulos tiende a generar una configuración de superficie, contornos y patrón de "llenado" distinta, por lo que tiene pleno sentido que el código elaborado por el modelo contribuya especificamente a la clasificación de los mismos.

Este trabajo no aborda el problema de como el tálamo o la corteza cerebral pueden utilizar la información creada en el núcleo cuneatus, pero parece razonable que son las neuronas de gran campo receptor situadas en la corteza somatosensorial primaria las destinatarias de tal código (referido a la variable *salida_global*). El modelo predice que el núcleo cuneatus colabora en una función hasta ahora encomendada a la corteza cerebral y es por tanto, pese a la sencillez de su circuitería, un lugar de procesamiento de la información y no sólo un lugar de relevo o filtrado.

## Referencias

1. Aguilar J, Soto C, Rivadulla C, Canedo A: The lemniscal-cuneate recurrent excitation is suppressed by strychnine and enhanced by $GABA_A$ antagonists in the anaesthetized cat. Eur J Neurosci. Vol. 16 (2002) 1697-1704.
2. Canedo A, Martinez L, Mariño J: Tonic and bursting activity in the cuneate nucleus of the chloralose anesthetized cat. Neuroscience. Vol. 84 **2** (1998) 603-617.
3. Canedo A, Aguilar J, Mariño J: Lemniscal recurrent and transcortical influences on cuneate neurons. Neuroscience. Vol. 97 **2** (2000) 317-334.
4. Navarro J, Sánchez E, Canedo, A: Coding Srategies in Early Stages of the Somatosensory System. Lect. Notes in Comp. Sci. Vol. 3561 (2005) 213-222.
5. Navarro J, Sánchez E, Canedo, A: Spatio-temporal information coding in the cuneate nucleus. Neurocomputing. En prensa. (2006).
6. Nuñez A, Buño W: In vitro electrophysiological properties of rat dorsal column nuclei neurons. Eur. J. Neurosci. Vol. 11 (1999) 1865-1876.
7. Reboreda A, Sánchz E, Romero M, Lamas A: Intrinsic spontaneous activity and subthreshold oscillations in neurones of the rat dorsal column nuclei in culture. J. Physiol. Vol. 551 **1** (2003) 191-205.
8. Sánchez E, Barro S, Canedo A: Edge Detection and Motion Discrimination in the Cuneate Nucleus. Lect. Notes in Comp. Sci. Vol. 2415 (2002) 198-203.
9. Sánchez E, Aguilar J, Rivadulla C, Canedo A: The Role of Glycinergic Interneurons in the Dorsal Column Nuclei. Neurocomputing. Vol. 58-60 (2004) 1049-1055.

# Reconocimiento de objetos de forma libre y estimación de su posicionamiento usando descriptores de Fourier

Elizabeth González [1], Vicente Feliú [1], Antonio Adán[2] y Luis Sánchez [1]

[1] E.T.S.I Industriales, Universidad de Castilla-La Mancha,
Ave. Camilo José Cela s/n, 13071, Ciudad Real, España
{vicente.feliu, luis.sanchez}@uclm.es
[2] Escuela Superior de Informática, Universidad de Castilla-La Mancha,
Ronda de Calatrava 5, 13071 Ciudad Real, España
antonio.adan@uclm.es

**Resumen.** Este trabajo presenta una estrategia para el reconocimiento de objetos de forma libre y estimación de su posicionamiento a partir de la representación de sus siluetas mediante un conjunto reducido de descriptores de Fourier. El método se divide en dos partes. Primeramente se lleva a cabo un proceso off-line donde se calculan y almacenan los descriptores de Fourier de siluetas correspondientes a objetos tomadas desde un amplio conjunto de puntos de vista. Usando tres descriptores de Fourier se agrupan las siluetas en clusters y se determina un prototipo representativo de cada cluster. La segunda parte es un proceso on-line que comienza por clasificar la silueta en el espacio de clusters obteniendo un sub-espacio de trabajo donde se resuelve el problema de reconocimiento/posicionamiento mediante un conjunto reducido de descriptores de Fourier de la silueta. Este método ha sido experimentado en un entorno real con un robot manipulador dotado de una microcámara en su efector final.

*Palabras clave*: Visión artificial, reconocimiento 3D, siluetas, clustering, descriptores de Fourier, identificación, estimación de posicionamiento

## 1 Introducción

El uso de sensores de visión acoplados a manipuladores robóticos proporciona un amplio conjunto de posibilidades y aplicaciones en entornos industriales. Sin embargo, lograr aplicaciones flexibles y robustas que trabajen en tiempo real constituye, en estos momentos, el reto de muchos investigadores.

El reconocimiento de objetos en entornos 3D es una tarea muy usual en aplicaciones robóticas tales como ensamblaje, seguimiento o agarre [13]. Para el reconocimiento en 3D basado en visión artificial se necesitan modelos del objeto que contengan una representación de las características geométricas del mismo. Existen dos tendencias básicas para obtener dicho modelo: las basadas en la apariencia de la imagen o las basadas en el modelo geométrico [1].

Las representaciones basadas en la apariencia tienen el inconveniente de que necesitan bases de datos muy grandes debido a que almacenan una gran cantidad de información referente a diversos aspectos (escalado, condiciones de luminosidad, etc.).

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 500-511, 2006.

Por el contrario, las representaciones basadas en el modelo geométrico son mucho más compactas [11]. Existe una extensa la bibliografía publicada en los últimos años sobre tipos de modelos geométricos [9],[10] entre los que destacan la representación axial, representación volumétrica [14] y la representaciones basadas en curvas y contornos [2].

El proceso de reconocimiento se compone usualmente de dos fases: identificación del objeto y estimación de su posicionamiento. Cuando los objetos no presentan texturas significativas la forma de la silueta desde distintos puntos de vista puede contener información relevante sobre su identidad y posicionamiento. En este artículo se presenta un método de reconocimiento de objetos 3D basado en las siluetas obtenidas desde diferentes puntos de vistas transformando así el problema de reconocimiento 3D en un problema 2D.

Los mayoría de los métodos para reconocimiento 2D a partir de siluetas se basan en obtener esquemas de representación de las mismas que sean invariantes a rotación, traslación y escalado. Los más populares son los momentos invariantes y los descriptores de Fourier [6],[15]. Los momentos invariantes tienen el inconveniente de que siluetas completamente diferentes pueden tener el mismo invariante de bajo orden lo cual puede conllevar a problemas de ambigüedad en el proceso de reconocimiento [5]. Los descriptores de Fourier por el contrario tienen la característica de que descriptores similares corresponden a siluetas con un alto grado de similitud [12]. Las siluetas pueden ser representadas por un conjunto mínimo de descriptores de Fourier que dan una aproximación razonable obteniéndose así una representación muy compacta en comparación con el número de puntos que conforman la silueta original.

Si se cuenta con bases de datos que contienen gran cantidad de objetos y los requerimientos del sistema exigen trabajar en tiempo real, como es el caso de aplicaciones robóticas, es necesario entonces aplicar técnicas de discriminación. Los trabajos presentados por Knungo [7], Cheung [3] muestran algoritmos de *clustering* muy eficientes que permiten hacer buenas discriminaciones. La mayoría de las formulaciones se basan en minimizar una función objetivo.

El algoritmo k-means clustering [13] es uno de los más estudiados en la literatura. Dado un conjunto de *n* puntos pertenecientes a un espacio real d-dimensional, $R^d$, y un entero k, el problema es calcular un conjunto de *k* puntos en $R^d$ llamados centros que minimizan la distancia media cuadrática entre cada punto y su centro mas cercano. El método propuesto en nuestro trabajo usa los centros de cada cluster como prototipo representativo de dicho cluster.

Teniendo las siluetas agrupadas en clusters es posible seleccionar los clusters cuya medida de similitud entre sus prototipos y la silueta de la escena sea mayor. Las siluetas contenidas en los clusters que cumplen dicha condición pasan a formar el subespacio de trabajo a partir del cual se resuelve el problema de reconocimiento/posicionamiento de un objeto.

En este artículo se describe una estrategia de reconocimiento de objetos 3D para una escena de trabajo de un robot que tiene en el extremo final del brazo una cámara de visión y una pinza. En futuros trabajos el método implementado será usado para desarrollar una estrategia de agarre de objetos.

El método propuesto aporta como novedad la aplicación de descriptores de Fourier al reconocimiento de objetos 3D a partir de siluetas tomadas desde múltiples puntos

de vista de los mismos. Además, introduce el uso de los descriptores de Fourier como vector característico en técnicas de agrupamiento de siluetas. Esta estrategia de *clustering* lleva consigo una mejora en los tiempos de ejecución del algoritmo de reconocimiento como consecuencia de una potente discriminación (>80%) sobre la base de datos.

El artículo está organizado de la siguiente manera. En la sección 2 se trata el problema de agrupamiento de siluetas y discriminación. En la sección 3 se expone el procedimiento de reconocimiento de una silueta y la determinación de su posición, orientación y escalado. En la sección 4 se aborda la fase experimental y los resultados obtenidos durante la misma. Finalmente se enuncian las conclusiones y trabajos futuros.

## 2 Discriminación y clustering de siluetas mediante descriptores de Fourier

En este método la silueta (silueta del objeto 3D a identificar) debe ser reconocida entre un conjunto de siluetas (en nuestro caso 80 o 320 por objeto) de una librería de objetos. El proceso consta de una fase *off-line* y otra *on-line*.

El proceso off-line se encarga de construir una base de datos de los descriptores de Fourier de las siluetas y agrupar las siluetas en *clusters*. Para cada objeto se ha obtenido previamente un modelo tridimensional de alta resolución y precisión a través de un scanner láser. A continuación, se extraen las siluetas 2D de las imágenes de cada modelo visto desde un conjunto homogéneo de puntos de vista. La implantación de estos puntos de vistas se realiza a través de los vértices de una esfera teselada semiregular con origen en el c.d.m del objeto. La figura 1 a) muestra el modelo dentro de una esfera de 80 vértices mientras que en la figura 1 b) se observa la apariencia del modelo desde determinado punto de vista. Para cada una de estas vistas se extrae la silueta correspondiente (Figura 1 c) y 1 d)) y se almacena en la base de datos sus descriptores de Fourier.

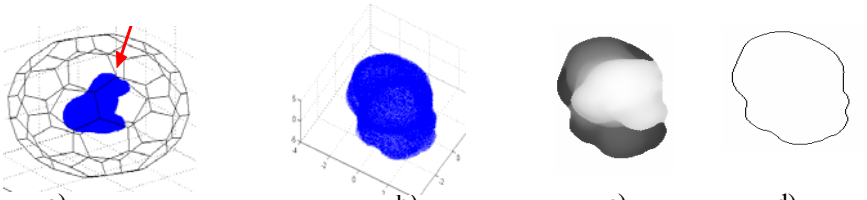



**Fig. 1.** a) Modelo del objeto dentro de una malla esférica. b) Vista del objeto desde un específico punto de vista. c) filtrado d) silueta 2D obtenida.

Obtenida la base de datos se procede a construir los clusters utilizando tres descriptores de Fourier como vector característico. La figura 2.a muestra el comportamiento del módulo de los descriptores de Fourier de varias siluetas después de haber sido centradas (primera componente igual a 0). Se observa que el mayor valor corresponde siempre a la última componente. En 2.b se puede apreciar que, después de normalizar usando la última componente, los valores más representativos se encuentran en las primeras y las últimas componentes (después de centrar y normalizar, la primera y la última componente se mantendrán constantes para todas las siluetas). Basándonos en esta característica se han escogido los módulos del segundo, tercer y penúltimo descriptor de Fourier para formar el vector característico o patrón del algoritmo de *clustering*. En la figura 2.c se puede apreciar que si usamos estos descriptores para representar una silueta se obtiene una aproximación primitiva de la misma.

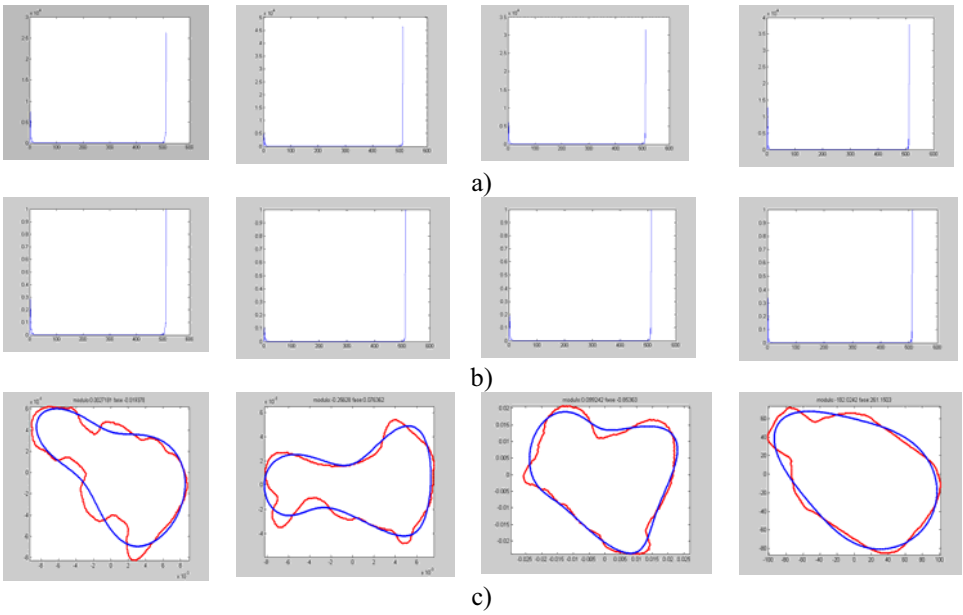



**Fig. 2.** a) Módulos de los descriptores de Fourier sin normalizar. b) Módulos de los descriptores de Fourier después de haber sido normalizados por la última componente. c) Representación de las siluetas usando solamente el segundo, tercer y penúltimo descriptor.

El algoritmo usado para formar los clusters fue el k-means [3]. Fijada una cantidad predeterminada de clusters, el algoritmo proporciona las siluetas y el prototipo representativo de cada cluster. En figura 3 se puede apreciar una representación espacial de los clusters calculados para nuestra base de datos mientras que la figura 4 muestra representaciones de siluetas y prototipos de algunos de los clusters obtenidos.

En el proceso *on-line* se soluciona el problema de reconocimiento/posicionamiento. El objeto es observado a través de una cámara y se necesita una rápida respuesta. Los pasos esenciales después de tomar una imagen del objeto son: extracción de la silueta, cálculo de los descriptores de Fourier, clasificación de la silueta y cálculo de posicionamiento.

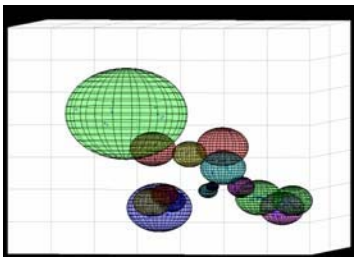


**Fig. 3.** Representación espacial de algunos clusters

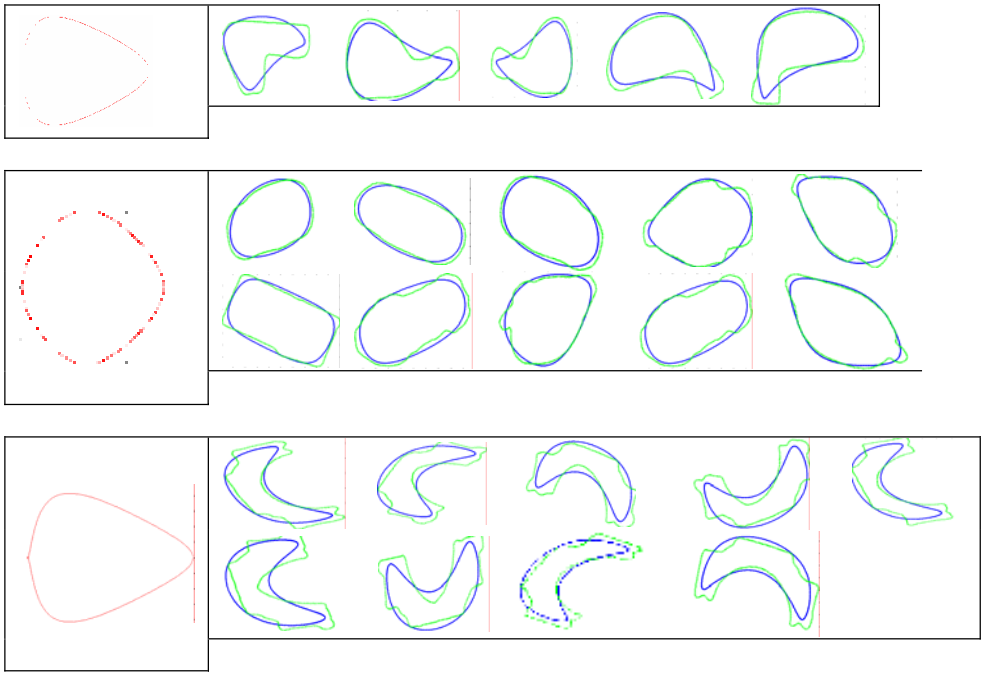


**Fig. 4.** Algunos clusters. A la izquierda el prototipo y a la derecha las siluetas que forman el cluster y sus representaciones primitivas usando solo tres descriptores de Fourier.

En la fase de clasificación, se establece un sub-espacio de búsqueda en el espacio de clusters. Para determinar este sub-espacio se determinan los clusters que mantienen un mínimo grado de similitud con la silueta escena. Formalmente el sub-espacio de búsqueda se obtiene de la siguiente manera:
Sea $\{O_1, O_2, ...O_N\}$ una librería de N objetos, $C_k$ el k-ésimo cluster, $k \in [1..K]$, donde $K$ es una cantidad predeterminada de clusters, $S_k^{mn}$ es la silueta $m$ del objeto $n$, $p_k$ el prototipo asociado al k-ésimo cluster, $D$ la distancia euclídea, $R_k$ el radio del k-ésimo cluster siendo $R_k = max\, D(p_k, S_k^{mn})$ y $z$ la silueta de la escena a identificar. El sub-espacio $S_{esp}$ estará formado por las siluetas de los clusters que cum-

plan los siguientes criterios:

***Criterio 1:*** Si $D(p_k, z) < R_k$ entonces $C_k \in S_{esp}$

***Criterio 2:*** Si $|D_i - min\, D_k| < \varepsilon$ entonces $C_i \in S_{esp}$, $i \in [1..K]$

El criterio 1 refleja los casos en que la silueta de la escena $z$ se encuentra contenida en un cluster y el criterio 2 se adapta a los casos en que $z$ se encuentra en un área donde hay mucha aglomeración de clusters o cuando no pertenece íntegramente a ningún cluster provocando incertidumbre.
Una vez concluido el proceso de discriminación se obtiene una base de datos reducida a partir de la cual usando el algoritmo que se explica en la sección 3.1 se extrae la mejor silueta candidata.

## 3 Algoritmo de identificación y estimación de posicionamiento libre

Los descriptores de Fourier tienen como característica ser invariantes a ciertas transformaciones geométricas y ser tolerantes ante ruido [4].

Una curva cerrada puede ser representada mediante series de Fourier con una parametrización adecuada. Consideremos un contorno *l(n)* constituido por N puntos en el plano XY:

$$l(n) = [x(n), y(n)], \quad n = 0..N-1 \tag{1}$$

cuyo origen es un punto arbitrario de la curva, n y n+1 son puntos consecutivos en alguna dirección (por ej., a favor de las manecillas del reloj) sobre la silueta. Asumiendo que todos los puntos sobre la curva han sido regularizados, y por tanto dos puntos consecutivos sobre la curva están a la misma distancia, podemos definir una secuencia compleja y su transformada discreta de Fourier *Z(k) = F(z(n))* como:

$$z(n) = x(n) + jy(n) \tag{2}$$

$$Z(k) = \sum_{n=0}^{N-1} z(n)\, exp(-j2\pi kn / N) \quad 0 \le k \le N-1 \tag{3}$$

Sea $(S_{esp})$ el sub-espacio de siluetas de trabajo con $R$ siluetas $s_r(n)$, $0 \le n \le N_r - 1$ con $1 \le r \le R$ y $N_r$ el número de puntos de una silueta, cuyas respectivas transformadas discretas de Fourier serán denotadas como $S_r(k)$.

El problema a resolver es encontrar la correspondencia entre la silueta de la escena *z(n)* y alguna silueta $s_{r^*}(n)$ de la base de datos bajo la hipótesis que *z(n)* puede estar escalada $(\lambda)$, trasladada $(c_x + jc_y)$ y rotada $(\theta)$ con respecto a la mejor silueta correspondiente de la base de datos. Además el punto de referencia en *z(n)* (n = 0) puede ser diferente del punto de referencia de la silueta encontrada en la base de datos (se denotará con $\delta$ dicho desplazamiento). En la próxima sección se describe como obtener *c,* $\delta$, $\theta$, $\lambda$.

### 3.1 Identificación de la silueta más próxima en la base de datos

Supongamos que *z* es la silueta de la imagen de un objeto 2D capturado por la cámara y que corresponde a la silueta $s_{r*}(n)$ perteneciente al sub-espacio de siluetas de trabajo ($S_{esp}$). En general *z* se corresponde con $s_{r*}(n)$ después de ser desplazada, rotada, escalada y trasladada mediante el cálculo de dichos parámetros. Para $s_r(n)$, en el dominio del espacio se tiene:

$$z(n) = \psi(s_r(n),\delta)\lambda exp(j\theta) + c \quad (4)$$

donde $\psi(s_r(n),\delta)$ está desplazada $\delta$ unidades del origen de la secuencias $s_r(n)$. Tomando la transformada de Fourier de la expresión anterior, se obtiene:

$$Z(k) = \lambda exp(j\theta) exp(j2\pi k\delta / N) S_r(k) + cN \quad (5)$$

A continuación se describe cómo caracterizar la silueta (identificarla y calcular su pose):

- Traslación: Si todas las siluetas tienen el origen de coordenadas en su centro de masa $S_r(0) = 0$, de la expresión (5) se tiene, $c = Z(0)/N$.
- Determinación de la silueta más próxima: Definiendo $\hat{Z}(k) = Z(k) - cN$, el módulo de la expresión (5) queda como:

$$\left|\hat{Z}(k)\right| = \lambda \left|S_r(k)\right| \quad (6)$$

El procedimiento de ajuste minimiza el error cuadrático medio entre los descriptores de Fourier de la silueta de la escena y la silueta de la base de datos. Dado un par de siluetas $(z(n), s_r(n))$, el índice de similaridad $J_r$ se define como:

$$J_r(\lambda) = (\left|\hat{Z}\right| - \lambda \left|S_r\right|)^t (\left|\hat{Z}\right| - \lambda \left|S_r\right|) \quad (7)$$

donde $\left|\hat{Z}\right| = (|\hat{Z}(0)|, \ldots |\hat{Z}(N-1)|)^t$, $|S_r| = (|S_r(0)| \ldots |S_r(N-1)|)^t$, $|\ .\ |$=valor absoluto

Minimizando $J_r(\lambda)$ con respecto al vector de escalado $\lambda$, se obtiene:

$$\lambda_r^\circ = \frac{\left|\hat{Z}\right|^t \left|S_r\right|}{\left|S_r\right|^t \left|S_r\right|} \quad , \; J_r^\circ = \left|\hat{Z}\right|^t \left|\hat{Z}\right| - \frac{(\left|\hat{Z}\right|^t \left|S_r\right|)^2}{\left|S_r\right|^t \left|S_r\right|} \quad (8)$$

Después de calcular $J_r^\circ$ para todas las siluetas de la base de datos se selecciona la silueta que verifica $J_r^\circ \le U$, $U$ siendo un umbral específico.

Cuando se trabaja con siluetas es usual que desde diferentes puntos de vista siluetas de diferentes objetos sean muy similares. Por lo tanto, pueden existir casos en los que el umbral *U* sea pasado por más de una silueta. En estos casos de incertidumbre se obtiene una nueva imagen desde otra posición y se repite el procedimiento anteriormente descrito hasta que sólo existe una silueta candidata. El problema para solucionar el mejor punto de vista desde el cual deberá tomarse la nueva imagen será abordado en trabajos posteriores. Después de determinar la mejor silueta candidata el

ajuste entre la silueta de la escena y la silueta candidata se explica a continuación.

### 3.2 Cálculo de los parámetros de posicionamiento

Si denominamos $\lambda\, exp(j\theta) = q$ , el costo de la función a minimizar estará dado por (ver (4)):

$$f(\lambda,\theta,\delta) = (\hat{Z} - q\,P(\delta))^{t^*}\,(\hat{Z} - q\,P(\delta)) \tag{9}$$

donde

$\hat{Z} = (\hat{Z}(0),\ldots,\hat{Z}(N-1))^t$, $P(\delta) = (s(0), s(1)exp(j2\pi\delta/N),\ldots,s(N-1)exp(j2\pi\delta(N-1)/N))^t$,

$*$ denota el conjugado, $t^*$ denota la transpuesta conjugada.
Optimizando (9) respecto al número complejo $q$ :

$$q^\circ(\delta) = \frac{Z^t\,P^*(\delta)}{S^{t^*}\,S} \quad , \quad f^\circ(\delta) = Z^{t^*}\,Z - \frac{\left|Z^{t^*}\,P(\delta)\right|^2}{S^{t^*}\,S} \tag{10}$$

(note que $P^{t^*}(\delta)\,P(\delta) = S^{t^*}\cdot S$ ).

Teniendo en cuenta que $0 \le \delta \le N-1$ es entero, la parte derecha de la expresión (10) es calculada para todos los posibles valores de $\delta$ y $f^\circ = \min_\delta f^\circ(\delta)$ es determinada.

Entonces $f^\circ$ es el índice de similaridad de la silueta en el proceso de ajuste, $\delta^\circ$ es el correspondiente desplazamiento y $q^\circ$ es obtenida de la parte izquierda de la ecuación (10) particularizada para $\delta^\circ$ . Rotación y escalado son estimados a partir de $q^\circ$ :

$$\lambda^\circ = \left|q^\circ\right|;\ \theta^\circ = \angle q^\circ \tag{14}$$

## 4 Experimentación

La validación de este método ha sido llevada a cabo en un entorno real en nuestro laboratorio. La plataforma experimental esta compuesta de un robot Staubli RX 90 con una micro cámara Jai-CVM1000. El sistema robótico controla la posición y el ángulo de visión de la cámara hacia el objeto (Figure 4.).

En el proceso fuera de línea, el modelo sintetizado del objeto (con 80000 polígonos por objeto) fue construido usando un scanner láser V1-910 Konica Minolta 3D. A partir de este modelo obtenido se construyó la base de datos con sus respectivos descriptores de Fourier. Durante los experimentos fueron usadas bases de datos de 80 y 320 siluetas/modelo.

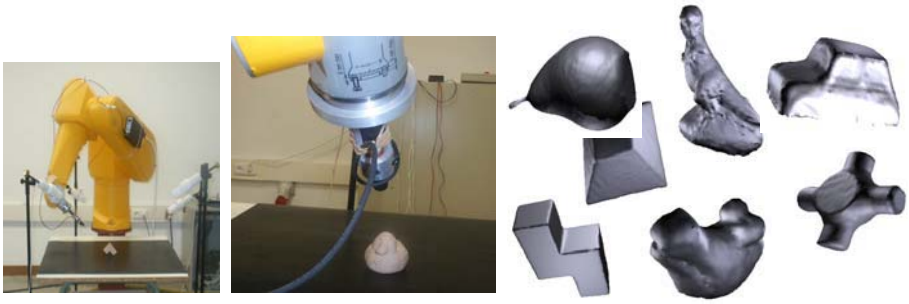
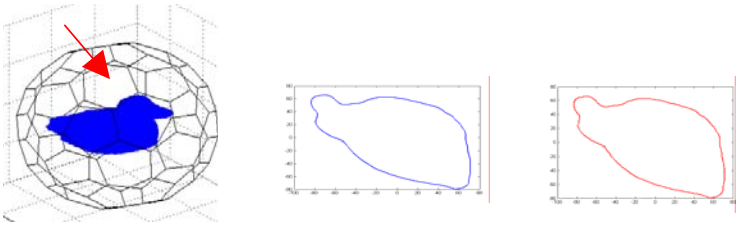


**Fig. 4.** a) Plataforma experimental. Imágenes de la escena tomada desde diferentes puntos de vista. b) Ejemplo de modelos sintéticos.

Con el objetivo de eliminar información redundante y optimizar los tiempos de ejecución de reconocimiento/posicionamiento se llevó a cabo un proceso de reducción de los descriptores de Fourier. Aprovechando la propiedad de que los primeros y los últimos descriptores son los que contienen la información más importante, el procedimiento de reducción consiste en buscar el mínimo índice *kr* de manera que el error de representación entre la silueta que se obtiene con los descriptores de Fourier en el intervalo [1,X(kr)], [X(N-kr),X(N)], kr=1,…N y la silueta original sea menor que un umbral $\chi$. La Tabla I muestra los resultados de la reducción de los descriptores de Fourier para diferentes valores de $\chi$ en figuras poliédricas y de forma libre.



b) c)

**Fig. 5.** b). Silueta correspondiente a 512 (original) descriptores c). Silueta reducida a 134 descriptores.

La experimentación fue llevada a cabo con 19 objetos previamente modelados con 80 y 320 vistas considerando reducciones para $\chi$ =0,05 y $\chi$ =0,5. Para el proceso de *clustering* se escogió que el número de clusters sería 50. El resultado durante los ensayos fue totalmente satisfactorio (100% de éxito).

Durante esta fase se trabajó con imágenes con una resolución de 640x480, las siluetas fueron almacenadas en la base de datos con una resolución de 512 puntos por silueta llegando a ocupar un espacio de memoria de 12 MB para el caso de 80 vistas y 42MB para 320 vistas.

La Tabla II muestra los resultados obtenidos durante el proceso de reconocimiento sin aplicar técnicas de discriminación (A) y aplicando estas (B). Se comparan los resultados en cuanto al número de siluetas del modelo, umbral de reducción de los descriptores ( $\chi$ ), error medio cuadrático ($\rho$) después de estimar los parámetros entre la silueta original y la obtenida por nuestro método de reconocimiento. La variable *t* es el tiempo de cálculo (segundos) en un Pentium II a 500Hhz.

Tabla I

| $\chi$ | Pol. | Free |
|---|---|---|
| 0.05 | 16,2% | 25,6% |
| 0.25 | 47,3% | 43,4% |
| 0.5 | 57,7% | 55,1% |

Tabla II

| **# sil.** | $\chi$ | $t_A$ | $t_B$ | $\rho$ |
|---|---|---|---|---|
| 80 | 0,05 | 2,652 | 2,475 | 2,382 |
| | 0,5 | 2,047 | 1,863 | 2,473 |
| 320 | 0,05 | 4,901 | 4,739 | 2,107 |
| | 0,5 | 3,336 | 3,096 | 2,261 |

Tabla III

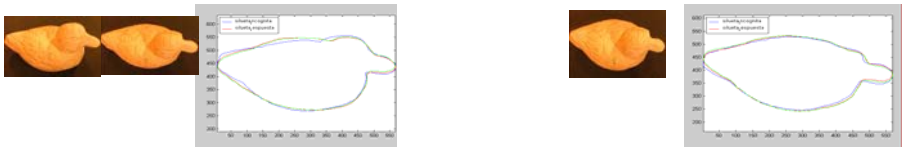
| **# sil.** | $\chi$ | | **Algoritmo** | **Tiempo (%)** |
|---|---|---|---|---|
| 80 | 0,5 | Sin *clustering* | Extracción de silueta | 92.5% |
| | | | Identificación | 6.4% |
| | | | Cálculo Posicionamiento | 1.1% |
| | | Con *clustering* | Extracción de silueta | 97.6% |
| | | | Construir Sub-Espacio | 0.5% |
| | | | Identificación | 1.2% |
| | | | Cálculo Posicionamiento | 0.7% |

Analizando la Tabla II y III se deducen los siguientes comentarios:

- Cuando se usan técnicas de discriminación durante el proceso de reconocimiento el tiempo es reduce en un 74%
- Trabajar con 320 siluetas/modelo hace que los tiempos de ejecución sean prácticamente el doble que si se usara 80 siluetas/modelo y $\rho$ decrece en un 0.3%

Las figuras 6,7 y 8 ilustran estos resultados. En la figura 6 se comparan los resultados entre usar 80 siluetas y 320. En el primer caso con bases de datos de 80 siluetas dos imágenes (posiciones de la cámara) son necesarias para hacer el reconocimiento mientras que con la base de datos de 320 siluetas/modelo solo fue necesaria una muestra.

La figura 7 presenta una comparación entre la reducción para $\chi$ =0,05 (izquierda) y $\chi$ =0,5 (derecha).



a) b)

**Fig. 6.** a) Comparación usando 80 siluetas (dos posiciones de la cámara) y b) 320 siluetas (una sola posición de la cámara).

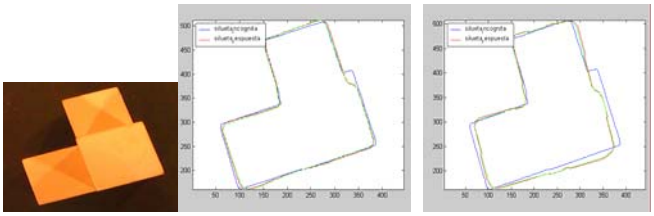


**Fig.7** Resultados para $\chi$ =0,05 (izquierda) y $\chi$ =0,5 (derecha).

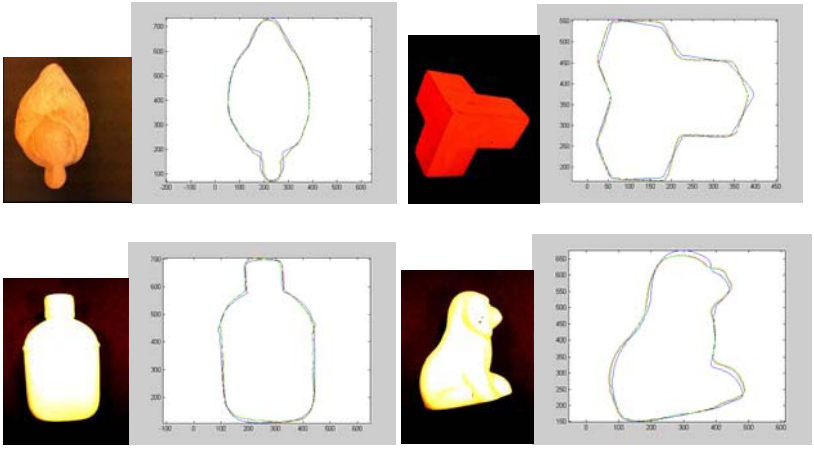


**Fig.8** Ejemplos del reconocimiento y el ajuste realizado a varios objetos

## 5 Conclusiones

En este artículo se presenta un nuevo método de reconocimiento y estimación de posicionamiento simultáneo en objetos de forma libre. El método esta basado en transformar un problema de reconocimiento 3D a un problema 2D de reconocimiento en siluetas. Los descriptores de Fourier han sido usados para llevar a cabo procesos de *clustering* de siluetas así como para realizar el proceso de identificación y calcular la estimación de los parámetros de posicionamiento.

En este método se diseña una base de datos de siluetas correspondientes a un gran número de puntos de vista de una librería de objetos. Para contrarrestar el problema de costo de memoria y procesamiento, se han desarrollado algoritmos eficientes para la reducción de los descriptores de Fourier y para la discriminación de siluetas que garantizan reducidos costos computacionales. El método ha sido experimentalmente probado en entornos reales, demostrando ser eficiente y de escaso costo computacional. Actualmente se trabaja en la optimización de clusters de siluetas y en estrategias de posiciones de la cámara para casos de ambigüedad en la identificación de siluetas.

## Referencias

[1] Adán, A., Cerrada, C., Feliú, V.: Global Shape Invariants A Solution for 3D Oject Dis crimination/Identification Problem. Pattern Recognition, vol. 34, pp. 1331-1348, 2001
[2] Antani, S., Kasturi, R. and Jain, R.: A survey of Free-Form Object Representation and Recognition Techniques. Comp. Vision Image Underst., vol. 81, pp. 166-210, 2001
[3] Cheung Y. M., k-Means: A new generalizedk-means clustering algorithm. Pattern Recognition Letters, 24, 2883-2893 (2003).
[4] Cyr, M.C., Benjamin, B.K.: 3D Object Recognition Using Shape Similiarity-Based Aspect Graph. In: Proceedings of ICCV. (2001).
[5] Dhillon, I. S., Guan, Y., & Kulis, B. Kernel k-means, Spectral Clustering and Normali-zed Cuts. Proceedings of the 2004 SIGKDD International Conference on Knowledge Discovery and DataMining, (pp. 551-556). (2004).
[6] Diaz de Leon, R. , Sucar, E.: Human Silhouette Recognition with Fourier Descriptors, Proc. of the International Conference on Pattern Recognition (ICPR'00),1051-4651/00
[7] Kanungo, T., Mount, D. M., Netanyahu, N., Piatko, C., Silverman, R. and Wu, A. Y.: An efficient k-means clustering algorithm: Analysis and implementation IEEE Trans. Pattern Analysis and Machine Intelligence, 24 881-892 (2002)
[8] Proakis, J.,Manolakis, D.: Tratamiento digital de señales. Prentice Hall pp 401-441 (2001)
[9] Rosenfeld, A.,:Axial representation of shape. Computer. Vision Graphics Image Processing, vol. 33, pp. 156-173 (1986).
[10] Sato, Y. Ohya, J. Ishii, K.: Smoothed local generalized cones: an axial representation of 3D shapes. Computer Vision and Pattern Recognition (1992)
[11] Selinger, A., Randal, C. N.: Minimally Supervised Acquisition of 3D Recognition Models from Cluttered Images. Comp.Vision and Pattern Recognition ( CVPR'01), vol 1, pp. 1960,
[12] Sethi, D. A., Renaudie, Kriegman, D., and Ponce, J.,: Curve and Surface Duals and the Recognition of Curved 3D Objects from their Silhouttes. International Journal of Computer Vision, 58(1), 73-86, 2004
[13] Smith, C., Papanikolopoulos, N.: Grasping of static and moving objects using vision-based control approach. Proceedings of the International Conference on Intelligent Robots and Systems (IROS'95), 0-8186-7108-4/03 (1995)
[14] Tubic, D., Hébert, P., Laurendeau, D.: A volumetric approach for interactive 3D model Ing. Computer Vision and Image Understanding, Volume 92,Issue 1, October 2003
[15] Zhang, D.S., Lu, G.: A Comparative Study of Fourier Descriptors for Shape Representation and Retrieval. In: Proceedings of ACCV. 646-651(2002)

# Verificación off-line de firmas manuscritas: Una propuesta basada en snakes y clasificadores fuzzy

José F. Vélez, Ángel Sánchez, Ana B. Moreno y José L. Esteban

Grupo de Algorítmica para la Visión Artificial y la Biometría (GAVAB),
Escuela Superior de Ciencias Experimentales y Tecnología,
Universidad Rey Juan Carlos, 28933, Madrid
jose.velez@urjc.es, angel.sanchez@urjc.es,
belen.moreno @urjc.es, joseluis.esteban@urjc.es

**Resumen.** En este artículo se presenta un método mejorado basado en la propuesta de *snakes* realizada por Kass, Witkin y Terzopoulos [11]. Dicho algoritmo permite la verificación, frente a falsificadores no entrenados, utilizando una firma en forma de imagen 2D bitonal obtenida *off-line*. Para la construcción del sistema se utiliza una única firma por individuo como muestra de aprendizaje. Además, se describe el caso real del problema de verificación de firmas en entorno bancario, y se adaptan los parámetros del algoritmo para resolver este problema de manera efectiva y eficiente.

## 1 Introducción

Las firmas manuscritas constituyen uno de los principales métodos de comprobación de autoría que manejamos [12]. Una de sus ventajas está en su fácil realización, pues apenas se necesita un papel y un bolígrafo. Otra ventaja está en la dificultad de realizar una falsificación que pueda engañar a una persona experta. Pero quizás su mayor ventaja esté en que es un mecanismo de autentificación personal ampliamente aceptado por la sociedad en general, y por organismos públicos y privados en particular.

Por estas razones, un sistema automático que realice la verificación de firmas de manera automática suscita un enorme interés [10]. El problema se ha abordado desde múltiples enfoques, habiéndose logrado buenos resultados bajo condiciones controladas. Sin embargo, aún no existe un sistema que, en condiciones reales, realice esta tarea con la misma efectividad que una persona con un entrenamiento mínimo.

El apartado 2 describe el problema de verificación. El apartado 3 realiza una breve introducción a los *snakes*. El apartado 4 introduce el algoritmo propuesto basado en una modificación al algoritmo original de *snakes*. Y los apartados 5 y 6 muestran nuestros experimentos, las conclusiones y futuras extensiones al trabajo realizado.

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 512-523, 2006.

## 2 La verificación de firmas

Supongamos que a un sistema que conoce las firmas de un conjunto de sujetos, se le presenta una firma y la supuesta identidad del firmante. El problema de la verificación consiste en determinar el grado de similitud entre la firma presentada y las que conoce del auténtico firmante, para establecer si es autentica o no. Como enuncia E. Justino [10] "en el problema de verificación de firmas se trata de maximizar las diferencias interpersonales y minimizar las diferencias intrapersonales".

Los falsificadores, que pueden tratar de engañar a un sistema de este tipo, pueden catalogarse en dos grupos: los falsificadores entrenados y los no entrenados [12]. Los primeros, que conocen la firma de la persona que quieren suplantar, se entrenan en reproducir la firma y consiguen unas falsificaciones de gran calidad. Los segundos no sólo no están entrenados, sino que no han visto nunca la firma, por lo que su reproducción no guarda ningún parecido con la auténtica. Curiosamente, y en contra de lo que pudiese pensarse, a este segundo grupo de falsificadores corresponde el 95% del fraude existente en entidades bancarias en el mundo [8].

El proceso de captura de firmas puede realizarse de dos formas diferentes: *on-line* y *off-line* [19]. En el modo *on-line* la captura se realiza utilizando un dispositivo especial (un lápiz electrónico o una tableta gráfica especial) que recoge información dinámica del escritor durante la firma. Esta información [19] incluye, además del grafismo, datos como presión, velocidad, puntos de inicio, direcciones de los trazos, inclinación, etc. Existen actualmente sistemas que realizan e forma eficiente reconocimiento y verificación utilizando esta información. Por otro lado, el método de captura *off-line* se basa en el escaneo de una firma, una vez realizada sobre un soporte ordinario (papel). En este caso, la información es mucho menor y la resolución espacial y radiométrica a la que se escanea influye en la verificación.

Además, en la formulación *off-line*, aparece el problema de localizar y segmentar la firma en el documento. A su vez, la segmentación dentro de un documento general presenta diferentes problemas: desconocimiento de la posición exacta y existencia de ruido blanco y con estructura. La Figura 1 ilustra estos problemas: sellos que se han superpuesto a la firma (c), tramas de logotipos sobre los que se ha firmado (b), texto adyacente a la zona de firmado y líneas o recuadros sobre los que se firma (a).


(a) (b) (c)

**Fig. 1.** Ejemplos de ruido blanco (a) y estructural (b), (c) y (d) que dificultan la segmentación de la firma.

### 2.1 Requisitos de un sistema offline real

Un primer requisito de un sistema offline de verificación, utilizable en condiciones reales, consiste en que no es viable solicitar muchas firmas a cada usuario para cons-

truir el sistema. En general sólo suele disponerse de una firma por persona. Sin embargo, sí es posible que la firma que se utilice para construir el sistema esté libre de ruido y en una posición conocida.

Otro requisito consiste en que, un sistema *off-line* real de verificación de firmas, deberá resultar escalable con respecto al número de individuos que es capaz de verificar, y no deberá ser exigente en cuanto a las condiciones de resolución tanto espacial como radiométrica.

Además, el usuario de un sistema como el indicado deseará percibir que el sistema sea fiable ante errores y resulte rápido en su funcionamiento. En algunos ámbitos, como el bancario, es preferible que el sistema acepte una firma falsa (Tasa de Error al Aceptar o FAR) a que rechace una firma verdadera (Tasa de Error al Rechazar o FRR) siempre que el coste económico de la supuesta falsificación no supere un umbral. Esto se debe a que previamente a la introducción de un sistema de verificación no se analiza la firma, o sólo se hace en casos de importes elevados, y por lo tanto el coste de trabajo es cero. Elevar este coste no suele ser aceptado por un cliente que "antes no tenía este problema".

### 2.2. Trabajos existentes

El tratamiento de firmas es un área de investigación muy activa desde mediados de 1970 [12]. Existen multitud de trabajos que abordan cada uno de los aspectos que se han presentado. Por ejemplo los hay que tratan el problema de la localización de la firma en un entorno ruidoso [18], los que tratan el problema de la imposibilidad de usar más de una firma por individuo para el entrenamiento [7], los que tratan el problema de la escalabilidad no asistida [4], o que tratan el problema de falsificadores entrenados [6].

En general, los enfoques existentes comparten el esquema clásico de extracción de características discriminantes y el posterior uso de un clasificador basado en tales características. Entre estas últimas las más utilizadas son: centros de gravedad (parciales o totales, línea base global (global *base line*), los límites superiores e inferiores de la firma, número de agujeros de la firma, esqueleto de la firma, tamaño de la envolvente convexa, caja que contiene la firma (*bounding box*), contorno o perímetro, ejes de mayor y menor inercia, relación entre área y perímetro, densidad de puntos en las regiones de la imagen, ángulo de inclinación (*slant*), puntos extremos superiores e inferiores, número de trazos, estructura de los trazos, puntos de cruce y de relleno de la firma a partir del eje de mínima inercia [19].

En cuanto a los métodos de clasificación existen propuestas variadas: los que usan HMMs [10][4], funciones de desplazamiento óptimo [7], ajuste elástico [6], lógica borrosa [9], redes neuronales [2][15] y algoritmos genéticos [14].

## 3 Base de datos de firmas

Debido a la inexistencia de una base de datos de referencia, hemos creado nuestra propia base de datos de firmas (http://gavab.escet.urjc.es). La base de datos consiste en 6 firmas tomadas a cada uno de 56 individuos. Las firmas se han tomado en instan-

tes diferentes y utilizando variados elementos de escritura. Finalmente, han sido escaneadas como imágenes bitonales a 300 puntos por pulgada, y almacenadas en formato BMP. La figura adjunta muestra algunas de las firmas de tal base de datos.

**Fig. 2.** Ejemplo de las firmas de dos individuos contenidas en la base de datos.

## 4 Los *snakes*

Los *snakes* [11][5] son un tipo de modelos de contornos activos [7], que se basan en el estudio del movimiento de un contorno abierto o cerrado sobre una imagen a la que trata de adaptarse. Asociado a este contorno se define una función de energía que tiene una componente interna y otra externa. La componente interna de la energía se debe a ciertas características de elasticidad y flexibilidad con que se dota al contorno. La componente externa se debe a la influencia de la imagen sobre la que se mueve el contorno. Utilizando estas componentes, un algoritmo mueve el contorno sobre la imagen buscando una posición de mínima energía. En este movimiento de búsqueda del mínimo se aprecia serpentear el contorno, hecho que da nombre al algoritmo.

Representando el parámetro $s$ la posición del *snake* sobre una imagen bidimensional como un conjunto infinito de puntos $v(s)= (x(s) , y(s))$ se puede escribir la función de energía:

$$E_{snake} = \int_0^1 E(v(s))ds = \int_0^1 \left[E_{\text{int}}(v(s)) + E_{imagen}(v(s)) + E_{cons}(v(s))\right]ds \quad \textbf{(1)}$$

En esta fórmula $E_{int}$ representa una energía debida a la relación entre los puntos del *snake*, $E_{imagen}$ representa una energía relativa a la posición que ocupan los puntos del *snake* dentro de la imagen, y finalmente $E_{cons}$ representa una energía asociada a otras condiciones externas.

### 4.1. Energía interna

Clásicamente [11] el término $E_{cons}$ no se considera, y para $E_{int}$ se propone la siguiente ecuación:

$$E_{\text{int}} = \frac{1}{2}\left(\alpha(s)\left[x_s^2(s) + y_s^2(s)\right] + \beta(s)\left[x_{ss}^2(s) + y_{ss}^2(s)\right]\right) \quad \textbf{(2)}$$

En la fórmula, $x_s$ e $y_s$ representan la derivada primera respecto al parámetro de posición $s$. Igualmente, $x_{ss}$ e $y_{ss}$ representan la derivada segunda. Por otro lado, los coeficientes $\alpha(s)$ y $\beta(s)$ corresponden respectivamente a la importancia de la elasticidad y a la flexibilidad del *snake*. Valores altos de $\alpha(s)$ hacen al *snake* encoger su forma. Mientras, valores altos de $\beta(s)$ fomentan que el *snake* enderecen su forma si no es

cerrado, o se vuelva convexo si es cerrado, haciéndolo en general más suave y menos anguloso.

Otras formulaciones [5] han sido propuestas para solventar diferentes tipos de problemas relacionados con el comportamiento del *snake* ante problemas particulares. En general, casi todas las formulaciones se basan en ecuaciones de tipo muelle.

### 4.2. Energía de la imagen

Asociado a cada punto de la imagen sobre la que se mueve el *snake* se define una función de energía. La función de energía en cada punto permite puntuar con valores menores aquellos puntos donde se desea que se sitúe el *snake*. Además se suele dotar a los puntos de una imagen de valores en gradiente que convergen hacia puntos de mínima energía, para fomentar el movimiento del *snake* en dicha dirección.

En su formulación original, el algoritmo de *snakes* propuesto por Kass y otros [11], contempla términos que atraen al contorno activo hacia líneas, bordes y terminaciones. Estos accidentes topológicos deben ser previamente segmentados en la imagen sobre la que se sitúa el *snake*. Así, la energía de la imagen se formula como:

$$E_{imagen} = E_{líneas} + E_{bordes} + E_{term} \quad (3)$$

Así, para atraer el *snake* hacia las líneas Kass propone utilizar la función de la imagen por si misma.

$$E_{líneas} = w\, I(x,y) \quad (4)$$

Para atraer el *snake* hacia los bordes se propone la función gradiente.

$$E_{bordes} = -\left|\nabla I(x,y)\right|^2 \quad (5)$$

Finalmente, para atraer el *snake* hacia los puntos de terminación se propone utilizar la curvatura de las líneas de nivel sobre la imagen suavizada *C* mediante un filtro de Gauss.

$$E_{term} = \frac{\partial \theta}{\partial n_{\perp}} \quad \text{donde } \theta = \text{arctg}(Cy/Cx) \text{ y } n_{\perp} = (-\text{sen}(\theta), \cos(\theta)) \quad (6)$$

### 4.3. Minimización de la energía del snake

En la búsqueda de un mínimo para el *snake* se hacen múltiples simplificaciones. En primer lugar se discretiza el contorno, de manera que pasa de ser una función continua a estar compuesto de un conjunto *n* de puntos de control $\{v_i\}_{i=1}^{n}$ que forman parte de un *spline* o incluso de un simple polígono. En segundo lugar se hacen constantes los coeficientes para las energías internas ($\alpha(s)$ y $\beta(s)$). Finalmente, se transforma la ecuación (1) a una formulación iterativa que permite el cálculo de las sucesivas posiciones de los puntos de control del *snake* en el tiempo $\{v_i(t)\}$, obteniendo finalmente un *problema variacional*.

Amini [1] propone el uso de programación dinámica para evitar la costosa evaluación de las múltiples posibilidades asociadas a cada movimiento del *snake*, siendo

éste el enfoque utilizado por nuestra implementación. En el método propuesto por Amini el cálculo de la energía de (1) puede descomponerse en etapas sucesivas según la siguiente formulación:

$$E(v_1, v_2, \ldots, v_n) = E_1(v_1, v_2) + E_2(v_2, v_3) + \ldots + E_{n-1}(v_{n-1}, v_n) \quad (7)$$

El cálculo de cada término de energía puede descomponerse utilizando programación dinámica discreta en una secuencia de funciones de una variable $s_i$ donde el conjunto $\{s_i\}_{i=1}^{n-1}$ se obtiene según el siguiente sistema de ecuaciones de recurrencia:

$$s_1(v_2) = \min_{v_i} E_1(v_1, v_2) \quad s_2(v_3) = \min_{v_2}\{s_1(v_2) + E_2(v_2, v_3)\} \quad s_3(v_4) = \min_{v_3} s\{_2(v_3) + E_3(v_3, v_4)\}$$

$$\cdots \qquad \min_{v_1,\ldots,v_n} E(v_1,\ldots,v_n) = \min_{v_{n-1}}\{s_{n-2}(v_{n-1}) + E_{n-1}(v_{n-2}, v_{n-1})\} \quad (8)$$

Para poder evaluar no sólo el primer sumando sino toda la ecuación (2), hay que tener en cuenta que la derivada segunda involucra a un tercer vértice, por lo que la descomposición de energías debe hacerse según:

$$E(v_1, v_2, \ldots, v_n) = E_1(v_1, v_2, v_3) + E_2(v_2, v_3, v_4) + \ldots + E_{n-1}(v_{n-2}, v_{n-1}, v_n) \quad (9)$$

donde $E_{i-1}(v_{i-1}, v_i, v_{i+1}) = E_{img}(v_i) + E_{int}(v_{i-1}, v_i, v_{i+1})$

Por ello, la secuencia de funciones $\{s_i\}_{i=1}^{n-1}$ será de dos variables, y para el caso de la función (2) tendrá la siguiente forma:

$$s_i(v_{i+1}, v_i) = \min\left\{s_{i-1}(v_i, v_{i-1}) + \alpha\left|v_i - v_{i-1}\right|^2 + \beta\left|v_{i+1} - 2v_i + v_{i-1}\right|^2 + E_{ext}(v_i)\right\} \quad (10)$$

## 5 Método propuesto de verificación de firmas basado en *snakes*

El problema de verificar una firma consiste en minimizar la energía de ajuste del *snake* asociado a la firma a comprobar con respecto a la firma original. El algoritmo de verificación propuesto en este trabajo puede resumirse en las siguientes etapas:

1. Crear de una línea poligonal P, con vértices equiespaciados, sobre el trazo de la imagen de una firma. Tal firma se utiliza como modelo.
2. Disponer P de manera aproximada sobre la imagen de una firma a verificar.
3. Utilizar el algoritmo de *snakes* para aproximar P a la imagen de la firma.
4. Utilizando una función que refleje el incremento de energía o de ajuste elástico respecto a la deformación que se produce en el *snake* tras la adaptación, obtener una medida de similitud entre la firma que dio origen al *snake* y la firma sobre la que se ha iterado. Este valor permitirá considerar la firma como auténtica o como falsa.

### 5.1. Propuesta inicial de ajuste.

En una primera aproximación se utilizó la definición original de Kass [11] tanto para las energías internas, como para la energía correspondiente a la imagen. Como las imágenes de firmas eran bitonales, la localización de bordes y de extremos no tenía

sentido para ajustar el *snake*. Tan sólo el uso de la energía asociada a cada píxel (9) era necesario para atraer el *snake* hacia la imagen de la firma.

Para imágenes sencillas el resultado el *snake*, en unas pocas iteraciones, se ajustaba con bastante precisión a la curva propuesta. Sin embargo el modelo tenía dos inconvenientes: por un lado la localidad y por otro la perdida de forma.

Si el *snake* no se encontraba muy cerca de la zona de cambio de valor de los píxeles de la imagen, el algoritmo fracasaba. Además, si no encontraba pronto la curva para ajustarse a ella, el *snake* perdía rápidamente su forma original. Para subsanar ambos problemas se decidió ampliar la zona de influencia de los bordes de la imagen mediante un suavizado de Gauss. En la Figura 3 se puede observar el resultado de este primer enfoque sobre un trazo sencillo. El proceso seguido consistió en la creación manual de una línea poligonal *P* sobre el trazo de la imagen de una firma, considerada como modelo. Además, se calcula el centro de masas *C* de los píxeles negros de la imagen de entrenamiento, y se localiza dentro del *snake*. Luego, se sitúa *P* sobre la imagen de una firma a verificar, haciendo coincidir el centro de masas de los píxeles negros de la imagen a verificar con el punto *C* del *snake*. Finalmente, se reescala *P* proporcionalmente a lo alto y a lo ancho para que coincida el ancho del *snake* con el ancho de la imagen.

**Fig. 3.** Ajuste del *snake* de parámetros ($\alpha,\beta,\omega$)=(0.1,1.0,-10.0). Donde $\alpha,\beta,\omega$ aparecen en las ecuaciones (2) y (4).

Tras diversas pruebas se pudo comprobar que este primer enfoque sólo ayuda en la zona afectada por filtro de Gauss, lo cual resultó insuficiente para imágenes de firmas reales.

### 5.2. Propuesta mejorada de ajuste

Para resolver estas dificultades se ha modificado la definición de energía $E'_{snake}$ del algoritmo original de *snakes* en dos sentidos. Para solventar el problema de la localidad se utilizan *mapas de potencial* para la energía asociada a la imagen $E'_{imagen}$. Para evitar el problema de la pérdida de forma se proponen unas fuerzas internas $E'_{forma}$ que mantienen la forma del *snake*. Así, la energía asociada al *snake* de una firma puede representarse como:

$$E'_{snake} = E'_{imagen} + E'_{forma} \quad (11)$$

**Mapas de potencial.** Cohen y Cohen [16] propusieron el uso de una fuerza externa utilizando un *mapa de potencial* basado en la distancia euclídea como solución al problema de no localidad. El enfoque resulta aplicable a este caso pues los puntos

pertenecientes a la firma están claramente determinados. El valor de distancia para cada punto del mapa $m_{imagen}(x,y)$, es igual al mínimo de los valores calculados para sus vecinos incrementado en una unidad. Este algoritmo calcula los valores de distancia partiendo de los puntos adyacentes a la firma y avanzando en capas hacia los límites de la imagen. En la Figura 4 se aprecia el resultado del cálculo del *mapa de potencial* para una firma de ejemplo.

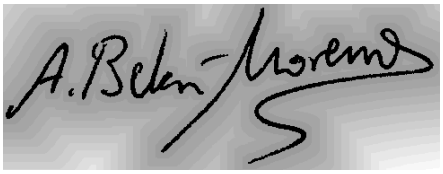


**Fig. 4.** Ejemplo de *mapa de potencial* sobre una firma. Se ha discretizado a 16 valores y se ha aumentado el contraste para que se aprecie mejor la ilustración.

El nivel de intensidad asociado a cada punto de la imagen representa la distancia al punto de la firma más cercano por lo que la diferencia entre dos píxeles adyacentes siempre es la unidad. Utilizando este *mapa de potencial* la energía de la imagen asociada a un punto de control $v_i$ del *snake* se define como:

$$E'_{imagen} = m_{imagen}(v_{i-1}) + m_{imagen}(v_i) + m_{imagen}(v_{i+1}) \quad \text{donde } i \text{ varía entre 1 y } n. \quad (12)$$

Esta formulación tiene en cuenta las energías de los puntos adyacentes a cada punto de control del *snake* evitando la inmovilidad de los extremos del *snake*.

**Mantenimiento de forma.** Con objeto de que el *snake* no pierda sensiblemente su forma original durante las sucesivas iteraciones, que se producen cuando su posición inicial no está cerca de su ajuste definitivo, se han propuesto dos términos para la energía interna:

$$E'_{forma} = E'_{angulo} + E'_{prop} \quad (13)$$

La primera función se encarga de mantener dentro de un rango el ángulo entre cada par de segmentos adyacentes de la línea poligonal $P$ con respecto al ángulo que inicialmente tenían esos dos segmentos. Para ello, cuando el ángulo en una iteración se separa de ese ángulo inicial se penaliza en un valor proporcional a la desviación. Además, si el ángulo supera un umbral $U_a$, a partir del cual se cambia en exceso la forma, se prohíbe dicho cambio haciendo infinita la energía.

Esta función se puede expresar como:

$$E'_{angulo}(v_{i-1}, v_i, v_{i+1}, t) = \begin{cases} \infty & \text{si } \delta > U_a \\ k_a \cdot \delta & \text{si } \delta \le U_a \end{cases} \quad (14)$$

donde $\delta = \left| ang(\overrightarrow{v_{i-1}(0)v_i(0)}, \overrightarrow{v_i(0)v_{i+1}(0)}) \right| - ang(\overrightarrow{v_{i-1}(t)v_i(t)}, \overrightarrow{v_i(t)v_{i+1}(t)})$ y *ang* representa el ángulo formado por dos vectores y $k_a$ es una constante que pondera la influencia del cambio del ángulo.

La segunda función preserva las proporciones entre segmentos adyacentes de la línea poligonal $P$. Para ello se calcula el ratio $\phi$ dado por el cociente de las proporcio-

nes iniciales de un par de segmentos adyacentes y el cociente de las proporciones del mismo par de segmentos en una iteración. El cambio respecto a 1 del cociente $\phi$ se penaliza hasta alcanzar un umbral $U_p$ a partir del cual se prohíbe haciéndolo infinito.

$$E'_{prop}(v_{i-1}, v_i, v_{i+1}, t) = \begin{cases} \infty & \text{si } 1 - U_p \text{ o si } \phi > 1 + U_p \\ k_p \cdot \phi & \text{si } 1 - U_p \le \phi \le 1 + U_p \end{cases} \quad \textbf{(15)}$$

donde $$\phi = \frac{\left\|\overrightarrow{v_i(t)v_{i+1}(t)}\right\| / \left\|\overrightarrow{v_{i-1}(t)v_i(t)}\right\|}{\left\|\overrightarrow{v_i(0)v_{i+1}(0)}\right\| / \left\|\overrightarrow{v_{i-1}(0)v_i(0)}\right\|}$$

y $k_p$ es una constante que pondera la influencia del cambio del tamaño.

Se aprecia que ambas fórmulas son compatibles con el enfoque de minimización aportado por (4).

### 5.3. Medida de la similitud.

Tras un número variable de iteraciones el *snake* converge a una solución. Para estudiar la similitud entre el *snake* y la figura a la que se ha ajustado se experimenta con diferentes algoritmos de *ajuste elástico* [17][3] pero se comprueba que éstos no se pueden utilizar, ya que las variaciones que se producen en el *snake* a lo largo de las iteraciones son demasiado pequeñas como para que estos algoritmos den una medida fiable para la posterior clasificación.

Por ellos se toma la decisión de definir las siguientes características dicriminantes: Factor de coincidencia y factor de distancia.

**Factor de coincidencia.** Este factor tiene su origen en la función de energía. Al igual que aquélla, utiliza una función de *mapa de potencial* $m_{imagen}(x,y)$ para asignar un valor a cada punto del *snake*, pero en esta ocasión se computan todos los $N$ puntos de los segmentos que componen el *snake*, y no sólo los puntos de control.

Para obtener un valor normalizado entre 0 y 1 de la coincidencia entre el *snake* y la firma a la que se ajusta se utiliza la siguiente expresión:

$$fc = 1 - \frac{1}{N}\sum_{i=0}^{N} c(p(i)) \qquad \text{siendo} \quad c(p(i)) = \frac{m_{\text{imagen}}(p(i))}{k_{fc} / g} \quad \textbf{(16)}$$

y donde $N$ el número de puntos considerados en el *snake*, $p(i)$ son los puntos interiores a los segmentos del *snake*, $g$ el grosor medio en píxeles del trazo de la firma y $k_{fc}$ es un factor de escala.

**Factor de distancia.** Este otro factor utiliza un *mapa de potencial* $m_{snake}(x,y)$, que tiene su origen en el *snake* una vez iterado, y qué permite calcular a que distancia se encuentran los píxeles de la firma respecto a la posición del *snake*. La formulación de la medida de la distancia es:

$$fd = \sum_{i=0}^{M} d(r(i)) \qquad \text{siendo} \quad d(p(i)) = \begin{cases} 1 & \text{si } m_{snake}(r(i)) < g \\ 0 & \text{si } m_{snake}(r(i)) \ge g \end{cases} \quad \textbf{(17)}$$

y donde $r(i)$ son los píxeles con valor negro de la imagen a verificar, $M$ es el número de tales píxeles y g es el grosor medio en píxeles del trazo de la misma firma.

Esta segunda medida tiene el problema de ser sensible al ruido. Mientras que el ruido aleatorio no debe presentar mayor problema, pues se puede eliminar con un sencillo filtro morfológico, el ruido con estructura puede ser un inconveniente en caso de no eliminarse para la verificación.

## 6 Modelo fuzzy

Sea un patrón $X$ cuyas características discriminantes $x_1$ y $x_2$ corresponden con el factor de coincidencia y el factor distancia. Para la clase de las firmas genuinas los valores de estos factores definen sendos conjuntos *fuzzy* $A_1$ y $A_2$. Utilizando un modelo de Takagi-Sugeno (TS) como mecanismo de clasificación obtendremos dos reglas con la forma:

$$\text{Regla}_k : \text{SI } x_k \text{ es } A_k \text{ ENTONCES } y_k=c_k+d_k x_k \quad k=1,2$$

Para el conjunto *fuzzy* $A_k$ se utiliza la siguiente función de pertenencia:

$$\mu_k(x_k) = e^{-\frac{(x_k-a_k)^2}{b_k}} \qquad k=1,2 \tag{18}$$

Y la salida se corresponde con:

$$O = \sum_{i=1}^{2} \mu_i y_i \tag{19}$$

Para calcular el valor de los parámetros $a_k$, $b_k$, $c_k$ y $d_k$ se ha utilizado un procedimiento iterativo basado en descenso del gradiente [20]. La muestra usada para el aprendizaje ha utilizado 28 de los 56 individuos de la base de datos, reservándose los otros 28 individuos para el test. Para cada individuo $I$ se ha utilizado 1 firma para construir el *snake* de $I$ y las otras 5 firmas de $I$ para crear la muestra de firmas genuinas del individuo. También se utilizan las firmas de los individuos distintos a $I$ para construir el conjunto de firmas a rechazar por el *snake* de $I$. Al realizar este proceso para los 28 individuos se obtendría un conjunto de entrenamiento con 28x5 pruebas de firmas a aceptar y 28x27x6 firmas a rechazar. Sin embargo, debido a que el proceso de aprendizaje resulta sesgado debido a la desproporción entre las dos clases a entrenar se eliminan aleatoriamente 27 de cada 28 firmas del conjunto de firmas a rechazar.

### 6.1 Resultados

Utilizando los 28 individuos restantes y el mismo procedimiento descrito en el punto 6 se crea un conjunto de test compuesto de 28x5 firmas a aceptar y 28x27x6 firmas a rechazar. Obsérvese que para utilizar el sistema con nuevos individuos no es necesario cambiar los parámetros del modelo TS.

Tras iterar cada firma con el *snake*, calcular sus características y comprarlas utilizando el modelo TS se obtiene un valor numérico entre 0 y 1. Este valor indica la pertenencia de la firma a la clase de las genuinas o de los falsificadores. Solo queda elegir el valor umbral que defina si el patrón presentado se considera genuino o una

falsificación. La curva de la figura 7 presenta los valores obtenidos de falso rechazo (FFR) y falsa aceptación (FAR) para diferentes valores de este umbral.

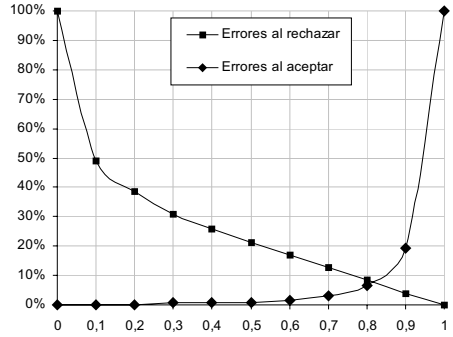



**Fig. 5.** Tasa de error al aceptar y al rechazar en función del umbral asignado a la salida.

En el gráfico se observa que se obtiene un punto de igual valor para los errores de un 7'5%. Además se aprecia que se puede obtener un 0% de error al rechazar con un 38% de error al aceptar.

## 7 Conclusiones y futuros trabajos

Se ha propuesto un algoritmo que ajusta un *snake* a una firma manuscrita. También se ha propuesto un algoritmo de verificación *off-line* de firmas en condiciones reales utilizando una única firma para el aprendizaje.

Actualmente, estamos emprendiendo trabajos en "sentido horizontal", abarcando los procesos que faltan para disponer de un sistema realmente operativo, y en "sentido vertical", mejorando los algoritmos descritos en este artículo.

En particular, los elementos que faltan para disponer de un sistema completo de verificación de firmas son dos: a) Un método de creación automática del *snake* a partir de la imagen de aprendizaje, para evitar su trazado manual. b) Un algoritmo que permita la localización y segmentación de la imagen de una firma dentro de una imagen de un documento genérico.

Por otro lado, las mejoras sobre el trabajo ya realizado se concentran en dos aspectos: a) La ampliación de la base de datos de test. b) La mejora del método de comparación entre el *snake* iterado y la imagen a verificar. c) La optimización de los algoritmos y parámetros para funcionar en los casos en que exista ruido.

## Referencias

1. Amir A. Amini y otros, "Using Dynamic Programing for Solving Variational Problems in Vision", IEEE Transactions on Patt. Analysys and Machine Intell., Vol 12, nº 9, Sept 1990.
2. R. Bajaj y S. Chaudhury, "Sign.Verification using Multiple Neural Classifiers", Patt. Recognition, Vol. 30, no. 1, pp. 1-7, January 1997.

3. A. Blake y M. Isard, “Active Contours”, Capítulo 3, Springer 2000.
4. J. L. Camino y otros, “Sign.Classification by HMM”, IEEE 33 Intern. Carnahan Conference, Madrid 1999.
5. N.E. Davison, “Snakes simplified”, Patt. Recog., Vol 33 pags 1651-1664, 2000.
6. B. Fang y otros, “Off-line Sign.verification with generated training samples”, IEE Proc.-Vis. Image and Signal Processing, Vol. 149, no. 2, pp. 85-90, 2002.
7. Z. Hou, C. Han, “Force field analysis snake: an imporved parametric active contour model”, Patt. Recog. Letters 26, 513-526, 2005.
8. Investigación y Programas, S.A, informe interno.
9. M. A. Ismail, Samia Gad, “Off-line arabic Sign.recognition and verification”, Patt. Recog., vol 33, pp 1727-1740, 2000.
10. E.J. Justino y otros, “The Interpersonal and Intrapersonal Variability Influences of Off-Line Sign.Verification using HMM”, Proc. XV Brazilian Symposium on Comp. Graphics and Image Proces. (SIBGRAPI 2002), Fortaleza (Brazil), October 2002.
11. M. Kass y otros, “Snakes: Active Contour Models”, International Journal of Computer Vision, 321-331, 1988.
12. F. Leclerc y R. Plamondon, “Automatic Sign.Verification: The State of the Art”, Intl. J. Patt. Recog and Artificial Intelligence, Vol. 8, no. 3, pp. 643-660, March 1994.
13. Y. Muzukami, y otros, “And off-line Sign.verification system using an extracted displacement function”, Patt. Recog. Letters, Vol. 23, pp. 1569-1577, 2002.
14. V.E. Ramesh, M. Narasimha, “Off-line Sign.verification using genetically optimized weighted features”, Patt. Recog, vol 32, pp 217-233, 1999.
15. J. Vélez y otros, “Robust off-line Sign. verification using compression NN and positional cuttings”, Proc. of the IEEE Conf. on NN for Signal Procesing, NNSP 2003,Tolouse 2003.
16. L. D. Cohen y I. Cohen, “Finite element method for active contour models and ballons for 2-D and 3-D images”, IEEE Trans. PAMI 15 (11),1131-1147, 1993.
17. K. Yoshida y H. Sakoe, “Online handwriteng character recognition for a personal computer system”, IEEE Trans. Cosum. Electron., 28(3), 202-9, CE 1982.
18. M. Ammar y otros, “Off-line preprocessing and verification of signatures”,Int. Journal of Patt. Recog. and Arti. Intel., 1987.
19. R. Plamondon, “Progress In Automatic Sign.Verfication”, Series in Machine Perception a Artificial Inteligence. Vol 13, 1994.
20. M. Hanmandlu, y otros. “Off-line Sign.verification and forgery detection using fuzzy modeling”, Patt. Recog Vol 38, 341-356, 2005.

# Boosting con reutilización de clasificadores débiles

Juan J. Rodríguez y Jesús Maudes

Lenguajes y Sistemas Informáticos, Universidad de Burgos
{jjrodriguez,jmaudes}@ubu.es

**Resumen** Boosting es un conjunto de métodos de combinación de clasificadores que se caracterizan por que son capaces de obtener clasificadores precisos (denominados *fuertes*) a partir de clasificadores *débiles*. Así, es habitual utilizar este método con árboles de decisión con sólo una decisión (denominados "decision stumps").
En este trabajo se propone una modificación del método basada en combinar los clasificadores débiles entre si, obteniendo árboles con varias decisiones, de modo que no sean tan débiles. El estudio experimental desarrollado demuestra que esta modificación es claramente beneficiosa, con un coste computacional muy similar.

*Palabras clave:* boosting, combinación de clasificadores, aprendizaje automático.

## 1. Introducción

La agrupación de varios clasificadores [13], es una manera natural de incrementar la precisión, con respecto a la que se consigue con la utilización aislada de dichos clasificadores. Hay métodos que combinan clasificadores generados con distintos métodos, como es el caso de Stacking [22]. Sin embargo, también es posible combinar clasificadores obtenidos utilizando un único método. Dado que la combinación de clasificadores idénticos no tiene ninguna utilidad, es necesario plantear alguna estrategia que permita generar distintos clasificadores a partir de un mismo método.

En el método denominado Bagging [2], cada clasificador se construye usando un conjunto de datos diferente, que es una muestra con reemplazamiento del conjunto de datos original. Normalmente el tamaño de la muestra coincide con el del conjunto de datos de partida, pero los conjuntos de datos difieren ya que en una muestra una instancia puede seleccionarse varias veces mientras que otra puede no seleccionarse. En el caso del método de los Subespacios Aleatorios [11] cada clasificador se construye utilizando todas las instancias de entrenamiento, pero con un subconjunto aleatorio de los atributos.

Hay métodos de combinación de clasificadores diseñados específicamente para combinar un tipo determinado de clasificadores, típicamente árboles de decisión. El método de los Bosques Aleatorios [3] es una variante de Bagging, en la que los clasificadores a combinar son Árboles Aleatorios. En este tipo de árboles,

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 524-535, 2006.

al seleccionar una decisión para un nodo, no se consideran todos los posibles atributos sino un subconjunto de los mismos. Otro manera de construir árboles aleatorios es la propuesta en [6].

Otros métodos de combinación de clasificadores diseñados específicamente para trabajar con árboles de decisión son [14,16,17]. En el caso de los árboles de decisión, una alternativa a crear una agrupación de árboles consiste en extender la idea de árboles de decisión, de modo que una sola estructura sea equivalente a varios árboles de decisión. Dentro de este tipo de métodos se encuentran los Árboles de Opciones [4,12] y los Multiárboles [8].

Uno de los métodos más populares para crear estas agrupaciones de clasificadores es el Boosting [19,20]. Este término engloba toda una familia de métodos, de la que AdaBoost es la variante más conocida.

Estos métodos trabajan asignando un peso a cada ejemplo. Inicialmente, todos los ejemplos tienen el mismo peso. En cada iteración, se construye un clasificador, denominado *base* o *débil*, utilizando algún método de aprendizaje, y teniendo en cuenta la distribución de pesos. A continuación, el peso de cada ejemplo se reajusta, en función de si el clasificador base le asigna la clase correcta o no. El resultado final se obtiene mediante voto ponderado de los clasificadores base.

Lo que diferencia a los métodos que se engloban dentro del nombre Boosting, frente a otro tipo de métodos de combinación de clasificadores, es el hecho de que se puede obtener un clasificador *fuerte* a partir de un método *débil*. Se dice que un método de construcción de clasificadores es débil si lo único que garantiza, para un conjunto de datos binario, es que el clasificador generado tenga un error sobre los datos de entrenamiento menor del 50 %. En conjuntos de datos con dos clases, el cumplir este requisito no debería suponer ninguna dificultad.

Un tipo de clasificador particularmente simple que se suele usar con Boosting [21,10] son los denominados "decision stumps", árboles de decisión con una sola decisión. En el presente trabajo se presenta una propuesta para mejorar los resultados obtenidos al utilizar Boosting sobre este tipo de clasificadores. Dicha propuesta se basa en combinar varios de estos decision stumps para formar árboles más complejos.

El artículo se organiza como sigue. La variante propuesta en el presente trabajo se describe en la sección 2. La sección 3 está dedicada a la validación experimental. Finalmente, las conclusiones se presentan en la sección 4.

## 2. Reutilización de clasificadores débiles

### 2.1. AdaBoost

Por conveniencia para el lector, la figura 1 muestra el método AdaBoost. Cada ejemplo $x_i$ pertenece a un dominio $\mathcal{X}$ y tiene asociada una etiqueta binaria $y_i$, que en este método se codifican como $+1$ y $-1$.

AdaBoost asocia a cada ejemplo $x_i$ un peso, que en la iteración $t$ se denomina $D_t(x_i)$. El método base genera un clasificador base $h_t$, teniendo en cuenta los

pesos. Se selecciona un valor real $\alpha_t$, el peso del clasificador base, que depende del error cometido por ese clasificador. Entonces, se reajustan los pesos.

Dados $(x_1, y_1), \ldots, (x_m, y_m)$ donde $x_i \in \mathcal{X}, y_i \in \{-1, +1\}$
Inicializar $D_1(i) = 1/(m)$
Para $t = 1, \ldots, T$:

- Entrenar clasificador base utilizando la distribución $D_t$
- Obtener clasificador base $h_t : \mathcal{X} \rightarrow \mathbb{R}$
- Seleccionar $\alpha_t \in \mathbb{R}$ (en función del error cometido por $h_t$ sobre $D_t$)
- Actualizar

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

donde $Z_t$ es un factor de normalización (elegido de modo que $D_{t+1}$ sea una distribución).

El clasificador final es :

$$H(x) = \text{signo}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

**Figura 1.** AdaBoost [19,20].

### 2.2. Ejemplo de clasificador

En la figura 2 se muestra un ejemplo de clasificador obtenido con AdaBoost, utilizando como clasificadores base decision stumps. En concreto, ese clasificador se ha obtenido para el conjunto de datos *Sonar* [1], en el que se trata de discriminar entre dos clases: "rock" y "mine". En ese conjunto de datos hay 60 atributos numéricos.

El número de clasificadores que forman la combinación es 5. Esos 5 clasificadores son los que se obtienen si se fija el número de iteraciones a 5, pero dado que AdaBoost es un método iterativo también son los 5 primeros clasificadores de un clasificador obtenido con más iteraciones.

Para clasificar utilizando esa combinación de clasificadores, la instancia es clasificada por los 5 árboles. Cada uno de los clasificadores tiene un voto, que en valor absoluto se corresponde con el peso del árbol. El signo del voto depende de si el clasificador predice una u otra clase. Se suman los votos de los 5 árboles y en función del signo de la suma se predice una u otra clase.

### 2.3. Método propuesto

Para intentar mejorar los resultados obtenidos con AdaBoost y un método de clasificación débil, caben dos aproximaciones directas:

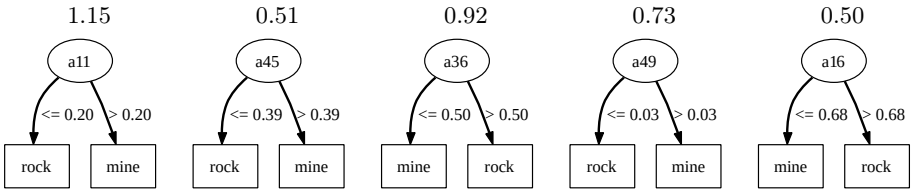



**Figura 2.** Ejemplo de combinación de clasificadores obtenida con AdaBoost. Encima de cada clasificador se indica su peso.

- Añadir más clasificadores débiles.
- Utilizar clasificadores base más complejos.

Ambas aproximaciones tienen los inconvenientes de que es necesario más tiempo para construir los clasificadores, para utilizar los clasificadores y que sus necesidades de almacenamiento aumentan.

Por tanto, sería interesante el poder obtener clasificadores más precisos sin aumentar significativamente el coste computacional o el espacio de almacenamiento.

La propuesta que se presenta en el presente trabajo consiste en combinar varios clasificadores débiles para obtener clasificadores menos débiles. Por ejemplo, si se combinan los 3 primeros clasificadores de la figura 2 se obtendría el árbol de la figura 3. Para determinar el valor asociado a cada una de las hojas del árbol habría que determinar a qué hoja va a parar cada uno de los ejemplos de entrenamiento, y asociar a cada hoja la clase más frecuente de sus ejemplos.

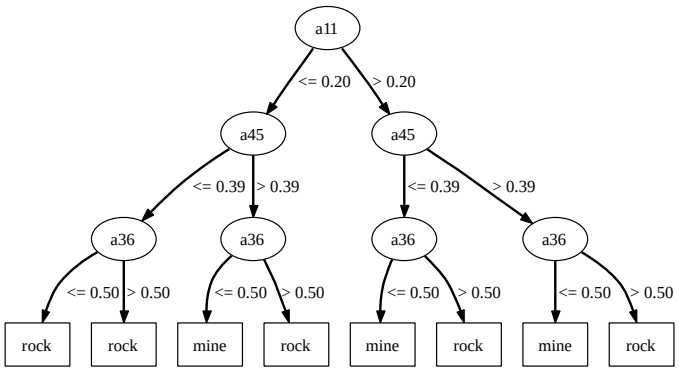



**Figura 3.** Árbol obtenido mediante combinación de clasificadores débiles.

El proceso consiste, por tanto, en que cada vez que se selecciona un clasificador débil, construir un clasificador algo más fuerte utilizando los clasificadores seleccionados en las iteraciones anteriores. Un parámetro del método, que se de-

nominará nivel de reutilización, $r$, será cuántos clasificadores de las iteraciones anteriores se van a utilizar. Por ejemplo, si el nivel de reutilización es 2 y se hacen 5 iteraciones de boosting, el clasificador final estaría formado por 5 árboles. El primero con sólo una decisión, el segundo con 2 (y cuatro hojas) y los 3 restantes con 3 decisiones (y 8 hojas).

La figura 4 muestra la variante de AdaBoost con reutilización. La única diferencia es que en vez de utilizar el clasificador $h_t$ directamente, éste se combina con los $r$ anteriores para obtener un clasificador con reutilización $h_t^r$.

Dados $(x_1, y_1), \ldots, (x_m, y_m)$ donde $x_i \in \mathcal{X}, y_i \in \{-1, +1\}$
Inicializar $D_1(i) = 1/(m)$
Para $t = 1, \ldots, T$:

- Entrenar clasificador base utilizando la distribución $D_t$
- Obtener clasificador base $h_t : \mathcal{X} \to \mathbb{R}$
- Construir clasificador $h_t^r$ combinando $h_t, h_{t-1}, \ldots h_{\text{máx}(h-r,0)}$
- Seleccionar $\alpha_t \in \mathbb{R}$ (en función del error cometido por $h_t^r$ sobre $D_t$)
- Actualizar

$$D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t y_i h_t^r(x_i))}{Z_t}$$

  donde $Z_t$ es un factor de normalización (elegido de modo que $D_{t+1}$ sea una distribución).

El clasificador final es :

$$H(x) = \text{signo}\left(\sum_{t=1}^{T} \alpha_t h_t^r(x)\right)$$

**Figura 4.** AdaBoost con Reutilización.

### 2.4. Algunos detalles de implementación

Es necesario asociar para cada árbol cuál es la clase que se predice en cada una de las hojas. Dado que se consideran problemas binarios, sólo hace falta un bit por hoja. Si el nivel de reutilización es $r$, el número de hojas de cada árbol será $2^{r+1}$. Por ejemplo, si $r = 2$ basta con un byte por árbol. Para el árbol de la figura 3 el valor de ese byte sería 00101010 (ó 11010101 si la correspondencia entre clases y dígitos fuera la contraria).

La clasificación de una instancia tiene un coste similar a la versión sin reutilización, ya que ésta es clasificada por los mismos clasificadores débiles. Para determinar a qué hoja del clasificador compuesto está asociada la instancia, basta con una variable de tipo entero con $r + 1$ bits. Dicho número inicialmente estará a 0. Cada vez que se evalúa un clasificador débil, la salida del mismo se introduce por la izquierda, con un desplazamiento, perdiéndose el bit más significativo. Por ejemplo, si el nivel de reutilización es 2, la variable tiene el valor

110 y el decision stump actual devuelve un 1, dicha variable pasaría a valer 101. Eso quiere decir que en el árbol actual la hoja correspondiente a la instancia es la 3 (teniendo en cuenta que se numeran de 0 a 7).

A la hora de construir el clasificador con la versión de boosting con reutilización, es necesario clasificar cada una de las instancias de entrenamiento con clasificador compuesto. Esto se debe a que para actualizar los pesos de cada instancia se necesita saber si el clasificador actual la clasifica bien o mal. Para realizar esta clasificación haciendo el mismo número de clasificaciones de instancias por decision stump, basta con utilizar el mismo método que se planteaba en el parrafo anterior para la clasificación. Cada instancia de entrenamiento tendrá asociado un entero de $r + 1$ bits que se corresponde con la hoja asignada por el clasificador compuesto actual.

Para determinar cuál es la clase asociada a cada hoja de un clasificador compuesto, el punto de partida serán las instancias de entrenamiento, cada una de las cuales tiene asociada una clase y un entero de $r+1$ bits que indica cuál es su hoja. Basta con calcular una matriz de frecuencias, de dimensiones $2^{r+1} \times 2$. En la posición $(i, j)$ de la matriz se almacenará cual es la suma de los pesos de las instancias de la hoja $i$ pertenecientes a la clase $j$. Esta matriz se puede ir actualizando según se clasifica cada instancia por el decision stump. Una vez que esta matriz está calculada, la clase asociada a la hoja $i$ sera el valor de $j$ que maximice el valor de la matriz en $(i, j)$.

## 3. Validación experimental

Los conjuntos de datos utilizados son los que aparecen en el cuadro 1. Todos ellos son del repositorio de la UCI [1]. Para cada conjunto de datos y método se realizaron 10 validaciones cruzadas estratificadas de 10 grupos.

Se consideraron 3 variantes de AdaBoost: sin reutilización, reutilizando un clasificador (denominada AdaBoost-R1) y reutilizando 2 clasificadores (denominada AdaBoost-R2). En cuanto al tamaño de la combinación de clasificadores, o lo que es lo mismo, el número de iteraciones de boosting, se consideraron 3 valores: 10, 25 y 100.

No se consideraron otros métodos, como por ejemplo AdaBoost con árboles de mayor profundidad, ya que se deseaba comparar métodos cuya complejidad fuese similar. En la variante con reutilización de AdaBoost se construyen tantos clasificadores, y de la misma complejidad, como en la variante clásica (sin reutilización). Para un tamaño de 100, el número de decisiones que tiene el clasificador es 100 para las 3 variantes consideradas. Si se hiciese boosting con árboles de mayor profundidad, el número de decisiones sería mayor. Por ejemplo, para árboles con 3 decisiones y 4 hojas, el número de decisiones de una combinación de 100 árboles sería 300. El número de decisiones es una medida relevante de la complejidad de un clasificador puesto que a la hora de construir el clasificador

**Cuadro 1.** Conjuntos de Datos.

| | | | Atributos | |
|---|---|---|---|---|
| Conjunto | Clases | Ejemplos | Discretos | Continuos |
| anneal | 6 | 898 | 32 | 6 |
| audiology | 24 | 226 | 69 | 0 |
| autos | 7 | 205 | 10 | 16 |
| balance-scale | 3 | 625 | 0 | 4 |
| breast-cancer | 2 | 286 | 10 | 0 |
| cleveland-14-heart | 2 | 303 | 7 | 6 |
| credit-rating | 2 | 690 | 9 | 6 |
| german-credit | 2 | 1000 | 13 | 7 |
| glass | 7 | 214 | 0 | 9 |
| heart-statlog | 2 | 270 | 0 | 13 |
| hepatitis | 2 | 155 | 13 | 6 |
| horse-colic | 2 | 368 | 16 | 7 |
| hungarian-14-heart | 2 | 294 | 7 | 6 |
| hypothyroid | 4 | 3772 | 22 | 7 |
| ionosphere | 2 | 351 | 0 | 34 |
| iris | 3 | 150 | 0 | 4 |
| labor | 2 | 57 | 8 | 8 |
| letter | 26 | 20000 | 0 | 16 |
| lymphography | 4 | 148 | 15 | 3 |
| mushroom | 2 | 8124 | 22 | 0 |
| pendigits | 10 | 10992 | 0 | 16 |
| pima-diabetes | 2 | 768 | 0 | 8 |
| primary-tumor | 22 | 239 | 17 | 0 |
| segment | 7 | 2310 | 0 | 19 |
| sonar | 2 | 208 | 0 | 60 |
| soybean | 19 | 683 | 35 | 0 |
| splice | 3 | 3190 | 60 | 0 |
| vehicle | 4 | 846 | 0 | 18 |
| vote | 2 | 435 | 16 | 0 |
| vowel-context | 11 | 990 | 2 | 10 |
| vowel-nocontext | 11 | 990 | 0 | 10 |
| waveform | 3 | 5000 | 0 | 40 |
| wisconsin-breast-cancer | 2 | 699 | 0 | 9 |
| zoo | 7 | 101 | 16 | 2 |

hay que seleccionar todas esas decisiones y a la hora de utilizar ese clasificador se deben evaluar todas las decisiones del mismo.[1]

Para trabajar con problemas con varias clases, dado que el método propuesto está diseñado para problemas binarios, se utilizó la estrategia denominada "uno contra todos" o "uno contra el resto", en la que se construyen tantos clasificadores binarios como clases. Así, si el problema tiene más de 2 clases, el número de decision stumps construidos será el producto del número de clases por el número de iteraciones de boosting.

El cuadro 2 muestra las tasas de acierto obtenidas por los distintos métodos para los distintos conjuntos de datos. En el cuadro 3 se indica cuántas veces un método obtiene mejor resultado que otro. Puede comprobarse que, con independencia del número de clasificadores considerado, AdaBoost-R2 es claramente superior a AdaBoost y a AdaBoost-R1.

La figura 5 muestra la relación de los resultados obtenidos por AdaBoost-R2 y AdaBoost. Se consideran dos relaciones: la diferencia entre tasas de acierto y el ratio entre las tasas de acierto. En estas gráficas se ordenan ascendentemente la relación (diferencia o ratio) para los distintos conjuntos de datos. Se puede comprobar, además de que se mejora en más casos de los que se empeora, que los beneficios obtenidos cuando se mejora son mayores que los perjuicios cuando se empeora.

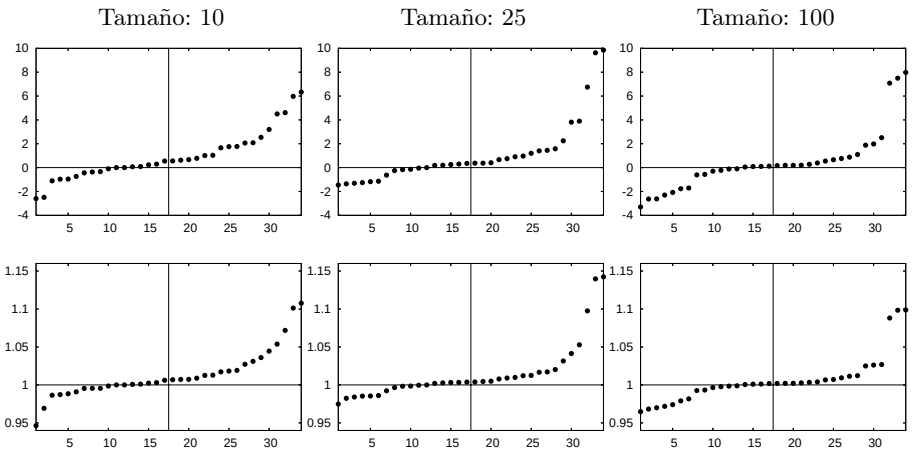



**Figura 5.** Relación, para los distintos conjuntos de datos, entre las tasas de acierto de AdaBoost-R2 y AdaBoost. Fila superior: diferencia entre las tasas de acierto. Fila inferior: ratio entre las tasas de acierto.

[1] Una misma decisión puede seleccionarse varias veces, en cuyo caso a la hora de clasificar sería suficiente con que se evaluara sólo una vez. Pero esta situación también aparece en las variantes con reutilización de AdaBoost.

**Cuadro 2.** Tasas de acierto de los distintos métodos para los distintos conjuntos de datos. Se muestran los datos correspondientes a 3 tamaños de la combinación: 10, 25 y 100 clasificadores. Para cada conjunto de datos y tamaño de la combinación considerado se resalta el mejor método.

| | AdaBoost | AdaBoost-R1 | AdaBoost-R2 | AdaBoost | AdaBoost-R1 | AdaBoost-R2 | AdaBoost | AdaBoost-R1 | AdaBoost-R2 |
|---|---|---|---|---|---|---|---|---|---|
| anneal | 96.49 | 97.10 | **98.25** | 98.25 | 98.90 | **99.22** | 99.12 | 99.39 | **99.51** |
| audiology | **75.05** | 73.74 | 74.09 | 77.08 | 77.33 | **77.43** | **80.15** | 78.87 | 78.07 |
| autos | 70.50 | 69.25 | **73.04** | 73.80 | 75.27 | **77.70** | 81.08 | 81.69 | **81.84** |
| balance-scale | 86.19 | **86.54** | 86.28 | 91.35 | 91.43 | **91.65** | 92.14 | 92.69 | **92.80** |
| breast-cancer | 71.93 | 71.44 | **75.13** | 72.57 | 71.87 | **73.48** | 72.25 | 71.70 | **73.13** |
| cleveland-14 | **83.28** | 82.68 | 82.31 | **82.85** | 81.82 | 81.39 | **82.35** | 81.02 | 79.73 |
| credit-rating | 84.80 | 84.80 | **85.42** | 85.71 | 85.58 | **86.38** | **86.14** | 85.97 | 85.84 |
| german-credit | **71.27** | 71.25 | **71.27** | 72.60 | **72.69** | 72.35 | **74.63** | 74.30 | 74.51 |
| glass | **69.60** | 69.00 | 69.50 | 71.63 | 72.95 | **73.08** | 73.70 | 74.40 | **75.68** |
| heart-statlog | **81.59** | 81.52 | 81.22 | **81.81** | 80.70 | 80.67 | **81.56** | 80.56 | 79.26 |
| hepatitis | 80.10 | **81.66** | 81.13 | 81.63 | 81.33 | **83.03** | **82.49** | 82.48 | 81.88 |
| horse-colic | 81.92 | **81.93** | 80.81 | **82.12** | 81.39 | 80.81 | 81.98 | 81.55 | **82.26** |
| hungarian-14 | 80.71 | **81.06** | 79.97 | 81.10 | **81.36** | 80.47 | **81.44** | 80.71 | 79.73 |
| hypothyroid | 99.10 | 99.08 | **99.33** | 99.20 | 99.44 | **99.57** | 99.58 | 99.62 | **99.63** |
| ionosphere | 90.89 | 91.05 | **91.45** | **92.34** | 92.34 | 92.34 | 92.63 | 92.37 | **92.82** |
| iris | **94.67** | 94.00 | **94.67** | **94.67** | 94.33 | 94.53 | 94.00 | 93.80 | **94.20** |
| labor | **84.30** | 83.33 | 81.70 | **86.43** | 83.17 | 85.17 | **87.43** | 83.70 | 84.80 |
| letter | 62.65 | 66.12 | **67.15** | 69.21 | 72.86 | **75.96** | 76.22 | 79.86 | **83.72** |
| lymphography | 80.42 | 80.09 | **81.43** | 82.62 | 82.37 | **83.37** | 83.37 | 83.33 | **83.92** |
| mushroom | 96.29 | 97.87 | **97.95** | 98.73 | 99.79 | **99.93** | 99.90 | **100.00** | **100.00** |
| pendigits | 85.82 | 88.46 | **90.43** | 92.05 | 94.24 | **95.86** | 95.89 | 97.49 | **98.40** |
| pima-diabetes | **74.93** | 74.37 | 74.59 | 75.37 | 75.22 | **75.57** | 75.45 | 75.27 | **75.54** |
| primary-tumor | **46.38** | 46.10 | 43.89 | **46.70** | 46.00 | 45.52 | 45.52 | **45.96** | 45.64 |
| segment | 93.50 | 93.87 | **94.17** | 94.66 | 95.66 | **96.24** | 96.52 | 97.31 | **97.62** |
| sonar | 75.65 | 77.31 | **77.71** | 81.20 | 81.15 | **81.46** | **84.86** | 84.42 | 84.29 |
| soybean | 88.22 | 89.87 | **89.00** | 92.19 | **92.49** | 90.83 | **93.75** | 92.86 | 90.45 |
| splice | 92.26 | 93.56 | **94.03** | 94.97 | 94.91 | **95.16** | 95.66 | 95.63 | **95.83** |
| vehicle | 66.93 | 68.50 | **69.01** | 71.24 | 72.53 | **73.49** | 75.19 | 76.06 | **77.07** |
| vote | **95.68** | 95.30 | 95.24 | **95.65** | 95.51 | 95.61 | **95.65** | 95.60 | 95.42 |
| vowel-context | 58.78 | 62.40 | **65.11** | 69.15 | 74.55 | **79.00** | 80.83 | 85.03 | **88.81** |
| vowel-nocontext | 58.98 | 62.35 | **64.95** | 68.95 | 74.42 | **78.58** | 80.36 | 84.24 | **87.44** |
| waveform | 80.27 | 80.69 | **80.82** | 83.59 | 83.69 | **84.00** | **84.92** | 84.89 | 84.82 |
| wisconsin-breast | 95.08 | **95.19** | 95.15 | 95.22 | 95.40 | **95.59** | 95.72 | 95.85 | **95.91** |
| zoo | 96.34 | **97.63** | 96.63 | 96.42 | **96.73** | 96.25 | **96.22** | 95.14 | 94.45 |
| | Tamaño: 10 | | | Tamaño: 25 | | | Tamaño: 100 | | |

**Cuadro 3.** Comparación de tasas de aciertos. Número de victorias, empates y derrotas del método de la columna respecto del de la fila.

| | AdaBoost-R1 | AdaBoost-R2 | AdaBoost-R1 | AdaBoost-R2 | AdaBoost-R1 | AdaBoost-R2 |
|---|---|---|---|---|---|---|
| AdaBoost | 19/1/14 | 22/1/11 | 19/0/15 | 22/0/12 | 14/0/20 | 21/0/13 |
| AdaBoost-R1 | - | 22/0/12 | - | 26/0/8 | - | 21/1/12 |
| | Tamaño: 10 | | Tamaño: 25 | | Tamaño: 100 | |

El cuadro 4 también muestra una comparación de los distintos métodos. Pero en esta tabla no se comparan directamente los valores medios de las tasas de acierto, sino que se usa el *estadístico test-t remuestreado corregido* [15] considerando los 100 valores (10 repeticiones de validación cruzada con 10 grupos) obtenidos para cada método y conjunto de datos. La supremacía de las versiones con reutilización sobre la versión sin reutilización son evidentes.

**Cuadro 4.** Comparación de tasas de aciertos. Número de victorias, empates y derrotas, de acuerdo al test estadístico utilizado, del método de la columna respecto del de la fila.

| | AdaBoost-R1 | AdaBoost-R2 | AdaBoost-R1 | AdaBoost-R2 | AdaBoost-R1 | AdaBoost-R2 |
|---|---|---|---|---|---|---|
| AdaBoost | 6/28/0 | 7/27/0 | 7/27/0 | 8/26/0 | 6/28/0 | 6/27/1 |
| AdaBoost-R1 | - | 4/30/0 | - | 5/29/0 | - | 4/29/1 |
| | Tamaño: 10 | | Tamaño: 25 | | Tamaño: 100 | |

Otra aproximación para comparar el resultado de varios métodos es el calcular un ranking promedio [5]. Para cada conjunto de datos, se calcula un ranking. Al mejor método se le asigna la posición 1, al segundo la 2, y así sucesivamente. Si varios métodos tienen el mismo resultado se les asigna una posición promedio (e.g., si 2 métodos tienen el mejor valor, a cada uno de ellos se le asigna la posición 1.5). Una vez que se ha calculado el ranking de cada conjunto de datos, se puede calcular para cada método el valor medio de su posición en el ranking. El cuadro 5 muestra estos rankings promedios.

**Cuadro 5.** Ranking promedio. Entre paréntesis se muestra la desviación estándar.

| | Tamaño: 10 | | Tamaño: 25 | | Tamaño: 100 | |
|---|---|---|---|---|---|---|
| AdaBoost | 2.25 | *(0.85)* | 2.21 | *(0.81)* | 2.03 | *(0.90)* |
| AdaBoost-R1 | 2.07 | *(0.70)* | 2.21 | *(0.64)* | 2.22 | *(0.51)* |
| AdaBoost-R2 | 1.68 | *(0.80)* | 1.59 | *(0.86)* | 1.75 | *(0.92)* |

## 4. Conclusiones y trabajos futuros

Si bien es posible utilizar boosting para combinar clasificadores complejos (árboles de decision, redes de neuronas...), lo que da el nombre al método es la capacidad de obtener un clasificador fuerte a partir de clasificadores débiles. Así, a diferencia de otros métodos de combinación de clasificadores, como Bagging, con Boosting se pueden obtener clasificadores precisos a partir de clasificadores muy simples.

En el presente trabajo se ha presentado un método que permite mejorar los resultados de AdaBoost cuando se utiliza con clasificadores base muy simples, en concreto árboles de decisión con sólo una decisión y dos hojas. Dicho método consiste en combinar varios árboles con una sola decisión para formar árboles más complejos.

La sobrecarga computacional introducida por la variante es mínima. Por un lado, al construir el clasificador, se construyen tantos árboles de una decisión como en la versión sin reutilización. Aunque en la versión con reutilización una misma decisión aparecerá en varios árboles, sólo es necesaria su evaluación para cada ejemplo de entrenamiento una vez.

En cuanto a los requisitos de almacenamiento del clasificador, son los mismos que para la versión sin reutilización, con la única diferencia de que se necesitan $2^{r+1}$ bits por clasificador, donde $r$ es el grado de reutilización. En el presente trabajo el valor máximo de $r$ considerado es 2, por lo que un byte es suficiente por cada clasificador. La clasificación de una instancia es prácticamente igual de rápida que en la versión sin reutilización, porque aunque cada decisión aparecerá en varios árboles, sólo se evalúan una vez.

La validación experimental, realizada sobre 34 conjuntos de datos del repositorio de la UCI, demuestra la validez del método propuesto.

En el presente trabajo sólo se consideran árboles con una sola decisión. El método es también aplicable a otro tipo de clasificadores, siempre y cuando sean relativamente simples. Por tanto, en trabajos futuros se considerará el uso del método sobre otros clasificadores base.

El método propuesto asume, tal como hacía el método original de AdaBoost, que la tarea de clasificación es binaria, por lo que en el presente trabajo para abordar problemas multiclase se construyen tantos clasificadores como clases. Sin embargo, resultaría interesante la capacidad de reutilización a variantes de AdaBoost que sí que permiten trabajar directamente con varias clases, como AdaBoost.M2 [9], AdaBoost.MH [21], AdaBoost.OC [18] o BoostMA [7].

## Agradecimientos

Este trabajo ha sido parcialmente financiado por los proyectos DPI2005-08498 del MEC y VA088A05 de la Junta de Castilla y León.

## Referencias


1. C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 1998. `http://www.ics.uci.edu/~mlearn/MLRepository.html`.
2. L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.
3. L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
4. W. Buntine. Learning classification trees. In *Artificial Intelligence frontiers in statistics*, pages 182–201. Chapman & Hall,London, 1993.
5. J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.
6. T. G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–158, 2000.
7. G. Eibl and K. P. Pfeiffer. Multiclass-boosting for weak classifiers. *Journal of Machine Learning Research*, 6:189–210, 2005.
8. V. Estruch, C. Ferri, J. Hernández-Orallo, and M.J. Ramírez-Quintana. Beam search extraction and forgetting strategies on shared ensembles. In *Multiple Classifier Systems, 4th International Workshop, MCS 2003*, 2003.
9. Y. Freund and R. E. Scapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, August 1997.
10. J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. *The Annals of Statistics*, 38(2):337–374, 2000.
11. T. K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.
12. R. Kohavi and C. Kunz. Option decision trees with majority votes. In *Machine Learning: Proceedings of the Fourteenth International Conference*, 1997.
13. L. I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, 2004.
14. J. Li and H. Liu. Ensembles of cascading trees. In *3rd IEEE International Conference on Data Mining (ICDM'03)*, pages 585–588, 2003.
15. C. Nadeau and Y. Bengio. Inference for the generalization error. *Machine Learning*, 52(239–281), 2003.
16. J. J. Rodríguez, L. I. Kuncheva, and C. J. Alonso. Rotation forest: A new classifier ensemble method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2006. Accepted.
17. J. J. Rodríguez and J. Maudes. Ensembles of grafted trees. In *ECAI-06, 17th European Conference on Artificial Intelligence*, 2006. Accepted.
18. R. E. Schapire. Using output codes to boost multiclass learning problems. In *14th International Conference on Machine Learning (ICML-97)*, pages 313–321, 1997.
19. R. E. Schapire. A brief introduction to boosting. In Thomas Dean, editor, *16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 1401–1406. Morgan Kaufmann, 1999.
20. R. E. Schapire. The boosting approach to machine learning: An overview. In *MSRI Workshop on Nonlinear Estimation and Classification*, 2002. `http://www.cs.princeton.edu/~schapire/papers/msri.ps.gz`.
21. R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. In *11th Annual Conference on Computational Learning Theory (COLT 1998)*, pages 80–91. ACM, 1998.
22. D.H. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–260, 1992.

# Análisis de escenas 3D: Segmentación y grafos de situación

Antonio Adán[1], Pilar Merchán[2] y Santiago Salamanca[3]

[1] Escuela Superior de Informática, Universidad de Castilla-La Mancha,
Ronda de Calatrava 5, 13071 Ciudad Real, España
Antonio.adan@uclm.es
[2, 3] Escuela de Ingenierías Industriales, Universidad de Extremadura,
Avda de Elvas s/n. 06071 Badajoz, España
{pmerchan@unex.es,ssalaman@unex.es}

**Resumen.** En este trabajo se presenta un método de extracción de información sobre la disposición relativa de los objetos en una escena compleja de la que se dispone de una única imagen de rango. En las escenas no existen restricciones de forma y posicionamiento de los objetos ni restricciones de oclusión. El proceso de análisis se basa en una estrategia de segmentación 3D distribuida y en una clasificación de siluetas de los segmentos en términos de oclusión. Finalmente se obtiene un *grafo de oclusión* que sintetiza la disposición de los objetos en la escena. Este trabajo forma parte de una etapa en un proyecto de interacción inteligente de robots en escenas y ha sido experimentado en un entorno real con resultados satisfactorios.

## 1 Introducción

Supongamos que tenemos una única vista de una escena compleja y un robot tiene que manipular los objetos que están en ella. En la figura 1.a) se muestra un entorno real con los elementos que estamos utilizando en nuestro trabajo: la escena, el sensor de visión inmóvil y el robot. La complejidad de la escena viene determinada por la aparición de sombras, oclusiones, contactos, superficies vistas desde ángulos oblicuos, objetos sin textura y sin restricciones de forma.

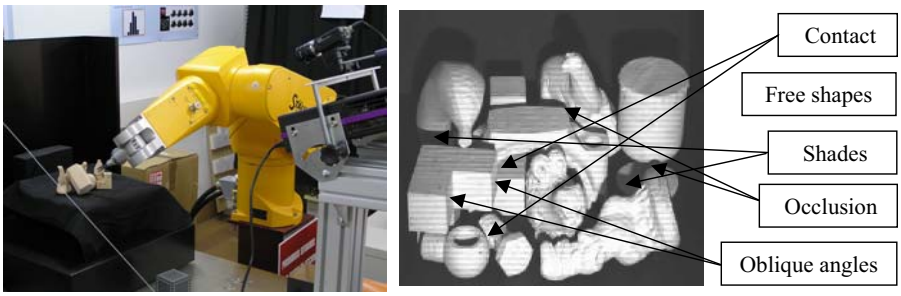
La interacción (agarre, empuje, etc.) de un robot en este tipo de escenas resulta ser una tarea de alto grado de dificultad que exige un conocimiento previo de los objetos y de su disposición. En este artículo tratamos el problema de la segmentación de los objetos que forman una escena y de la posición relativa de los objetos en la misma. La complejidad del análisis depende del grado de complejidad de la escena. Por ejemplo, en el caso de imágenes no texturadas, ninguna técnica de procesamiento de imagen garantiza buenos resultados en la segmentación. Por este motivo, nosotros hemos utilizado técnicas de segmentación de imágenes de rango en lugar de técnicas de procesamiento de imágenes.

Las técnicas de segmentación de imágenes de rango pueden ser clasificas en dos grandes grupos: técnicas basadas en la detección de bordes y técnicas basadas en el crecimiento de regiones. La referencia [1] ofrece una evaluación y una comparativa

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 536-547, 2006.

experimental entre ellas. En las aproximaciones basadas en la detección de bordes se identifican primero los puntos situados en los contornos, y a continuación se unen esos puntos para definir contornos y superficies. En este campo se pueden encontrar una gran variedad de algoritmos ([2], [3], [4]). En las aproximaciones basadas en el crecimiento de regiones se eligen de un cierto número de semillas y se impone un criterio de crecimiento. En [5], [6] y [7] se proponen algunos métodos de este tipo.

Por otra parte, la posición relativa de los objetos en la escena, a la que hemos llamado "disposición de la escena", es un problema que puede resolverse con técnicas basadas en siluetas. En un entorno controlado, la silueta de un objeto resulta bastante útil para determinar cuál es dicho objeto y su posición. A continuación se mencionan algunos trabajos basados en siluetas: en [8] Super et al. utilizan un método de recuperación de la forma basada en partes como generador de hipótesis para el sistema. Así evitan la comparación con todos los modelos de los objetos de la base de datos. En la referencia [9] Bilodeau et al. proponen una técnica en la que los objetos 3D son segmentados en partes 2D definidas como regiones delimitadas por grupos de arcos circulares y segmentos rectos. Serratosa et al. [10] definen los grafos descriptores de función (FDG), que se aplican al reconocimiento de rostros humanos. En [11] se presenta una aproximación de gráfico de aspecto que mide la similitud entre dos vistas a partir de una métrica de siluetas bidimensionales. En [12] Sebastian et al. proponen un entorno de reconocimiento basadas en perfiles de siluetas 2D.




*a)* *b)*

**Fig. 1** a) Interacción de un robot en la escena b) Vista de una escena, altamente compleja, utilizada en nuestro laboratorio.

Todos los trabajos mencionados imponen algún tipo de restricción: posiciones de los objetos limitadas [11], varias vistas de la escena [11,12], sólo para escenas simples [8, 9, 10] y no se permiten oclusiones. Por lo tanto, en todos los casos referidos se dispone de las siluetas completas de los objetos. La restricción de oclusión es una de las restricciones más exigentes ya que el tratamiento de objetos ocluidos implica un aumento de complejidad en el análisis de la escena.

En este trabajo se solucionan dos problemas esenciales en el análisis: segmentación de los objetos de la escena y conocimiento de la disposición relativa de los mismos en términos de oclusión construyéndose un grafo de oclusión. Por lo tanto se determinará qué partes de cada objeto ocluyen o son ocluidas por otros objetos. Este

proceso de alto nivel es un elemento fundamental para realizar una interacción inteligente de un robot. En este caso, a través del grafo de oclusión se puede guiar al robot para realizar una extracción secuenciada de los objetos evitando problemas de colisiones.

El artículo presentado está organizado de la siguiente manera: la sección 2 describe las etapas principales que constituyen el proceso de segmentación de la escena. La sección 3 se dedica a explicar la estrategia adoptada para analizar la organización de la escena generando el grafo de oclusión. La sección 4 muestra la experimentación realizada para validar el método. Por último, en la sección 5 se exponen las conclusiones.

## 2 Segmentación 3D de escenas complejas

Como ya se ha dicho, partimos de una vista única de la escena para separar los objetos que la componen. El método consiste en definir un conjunto de direcciones de exploración sobre una imagen de rango de la escena para llevar a cabo una técnica de segmentación distribuida. Es importante señalar que el sentido de la expresión *segmentación distribuida* se utiliza para expresar que los datos se distribuyen a lo largo del proceso a medida que aparecen niveles.

### 2.1 El núcleo del proceso distribuido

El esquema de segmentación propuesto es un proceso distribuido en el que un núcleo, al que llamamos de forma genérica $A^n_{i...jk}$, se ejecuta repetidamente en el nivel n siguiendo la secuencia i,j,…k a través de 2,…n-1 niveles. Al comienzo del proceso, el núcleo $A^1$ acepta los datos de la escena completa $D^0$ (la nube de puntos) proporcionados por el sensor. Como resultado de nuestra técnica de segmentación, cada parte segmentada de $D^0$ es una nueva nube de puntos que podría ser segmentada de nuevo, de forma que los datos de entrada para $A^n_{i...jk}$ es una parte de los datos segmentados en un núcleo previo $A^{n-1}_{i...j}$ .(Figura 2a)). El núcleo está formado por dos etapas :

I. Exploración de la escena, donde se selecciona una dirección de exploración
II. Segmentación 3D de la escena, donde, siguiendo la dirección de exploración, se realiza una proyección ortogonal de los datos en una imagen bidimensional sobre la que posteriormente se llevará a cabo la segmentación. Después se efectúa una transformación inversa para reasignar cada segmento bidimensional proyectado a los puntos tridimensionales correspondientes en la escena.

Cuando se ejecuta un núcleo existen dos posibilidades (Figura 2 b)):

- No hay segmentación: significa que el punto de vista utilizado no es apropiado para segmentar los datos de entrada. Si ocurre esto, se vuelve a la etapa de exploración y se busca un nuevo punto de vista (Figura 3 casos 3 y 4).

- Se produce segmentación: la información 3D es separada en varias partes (Figura 3 casos 1 y 2). Cada una de ellas puede ser un único objeto o un grupo de objetos de la escena. Cada segmento se toma de nuevo como entrada de la etapa 1 para volver a ser segmentados hasta que no sea posible más segmentación. Esta idea implica una distribución de la información original que se implementa en el árbol de datos Λ, donde la raíz de Λ es la imagen de rango original de la escena. Así, a lo largo del proceso de segmentación va surgiendo un conjunto de niveles de distribución.

## 2.2 Exploración de la imagen de rango

Para limitar el número de puntos de vista candidatos para la exploración utilizamos las direcciones que definen los nodos de una esfera teselada que envuelve a la imagen de rango de la escena (ver figura 3). Cada nodo *N* define un punto de vista *ON*, donde *O* es el centro de la esfera y el origen del sistema de referencia del mundo. A continuación se mapea sobre cada nodo N la probabilidad de éxito en segmentación teniéndose una esfera de probabilidades *E.* El nodo que se corresponde con el mayor valor de *E* define el punto de vista más adecuado para conseguir segmentación de la nube de puntos.

Por tanto, de acuerdo con nuestra estrategia de distribución, se obtiene un árbol de esferas de probabilidades $\Gamma$ dual al árbol de datos Λ. En consecuencia, para segmentar el segmento $D^L$, en el nivel $L$, la exploración se realiza de acuerdo con la esfera de probabilidades $E^L$. Además, $E^L$ se modifica a partir de los resultados de la segmentación (éxito o fracaso) una vez que se ha probado el mejor punto de vista teórico. Cuando ocurre segmentación se genera un nuevo nivel del árbol y la esfera de probabilidades se actualiza en $E^{L+1}$, que es una esfera-hijo de $E^L$. De esta forma, $E^{L+1}$ será la esfera-padre para el siguiente nivel. La figura 2 b) muestra la sinopsis de este procedimiento para el primer nivel. Los detalles de este subproceso, incluidas las reglas de actualización de las esferas de probabilidades, se encuentran en la referencia [13].

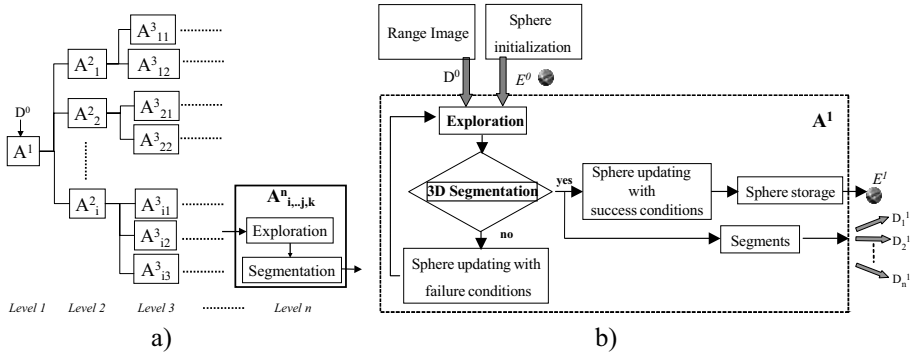



**Fig 2**. a) Representación del método distribuido b) Detalles de la entrada/salida y algoritmo en el núcleo $A^1$.

Hay dos razones fundamentales para utilizar esferas de probabilidades: por un lado, constituyen una herramienta útil que proporciona una búsqueda estructurada del punto de vista, lo que agiliza el proceso. Por otro, mejoran la eficacia del proceso de búsqueda porque nos permiten utilizar una heurística, esto es, podemos tomar decisiones durante el proceso basadas en el último estado de la esfera.

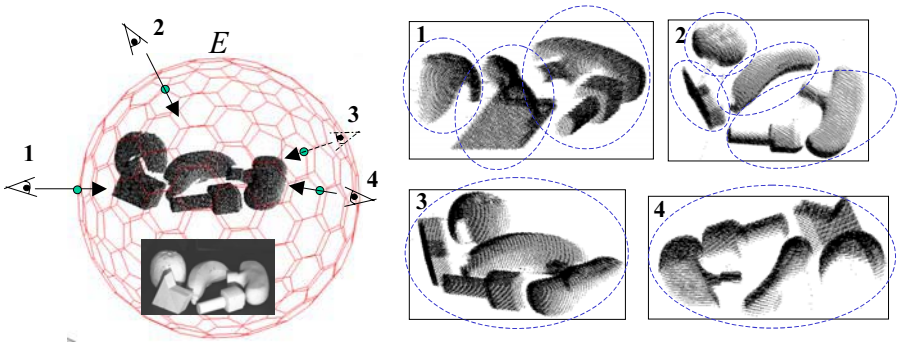



**Fig. 3.** Escena y esfera teselada utilizada para explorar la escena e imágenes de los datos de rango proyectados desde varios puntos de vista. Desde los puntos de vista 1 y 2 se lograría segmentación mientras que desde 3 y 4 no se extrae ningún segmento.

### 2.3 Segmentación de la imagen de rango

Para obtener los objetos que constituyen la escena hemos desarrollado una estrategia [datos 3D]→ [datos proyectados 2D]→[segmentación 2D]→[datos segmentados 3D]. La figura 4 a) muestra un ejemplo del procedimiento completo.

El proceso 3D → 2D supone una proyección ortogonal de la imagen de rango utilizando el punto de vista *ON* seleccionado en la etapa anterior. Esta transformación implica un cambio del sistema de referencia original S={OXYZ} a otro S'={OX'Y'Z'}donde la componente Z' es el valor de profundidad. La imagen proyectada $I_p$ se obtiene eligiendo las coordenadas X'e Y' e imponiendo un factor de paso de milímetros a pixels.

Para establecer la relación entre cada punto tridimensional y su píxel 2D asociado en $I_p$ , se define una estructura tridimensional llamada *Matriz Multipixel*, $M_{pix}$ (*i,j,k*). $M_{pix}$ se define de manera que todos los puntos que comparten la misma proyección están situados en la misma fila *i* y columna *j* y sus índices respectivos se almacenan a lo largo de la dimensión *k*. Esta estructura es creada por dos motivos: para realizar una segmentación eficiente y para realizar la transformación inversa 2D a 3D. Nótese que $M_{pix}$ es una estructura de datos que almacena los índices de los puntos originales proporcionados por el sensor.

La imagen $I_p$ es segmentada en regiones 8-conectadas a través de un algoritmo de semilla donde el tamaño de cada región es evaluado a través de la componente k de $M_{pix}$ de los puntos del segmento. Esto significa que regiones que en principio pueden parecer pequeñas en $I_p$ se correspondan en realidad con superficies proyectadas con

un ángulo oblicuo y se asocian con un número de puntos reales suficientemente grande como para poder ser considerado un objeto y no como simple ruido.

Finalmente se realiza la transformación inversa 2D-3D de cada segmento a través de $M_{pix}$ obteniendo los segmentos tridimensionales. La figura 4 muestra un diagrama que sintetiza este proceso y presenta un ejemplo de segmentación de una escena.

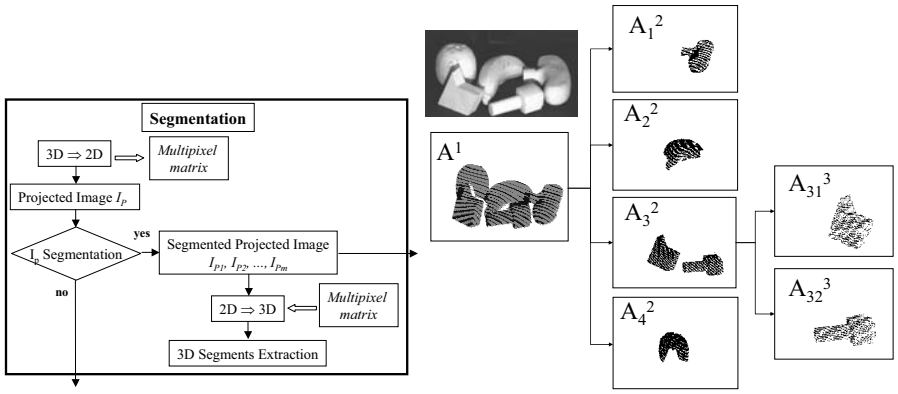



**Fig. 4**. a) Esquema de segmentación 3D b) Ejemplo de segmentación. *Arbol de datos* Λ.

## 3 Situación de los objetos en la escena

La disposición relativa de los objetos en la escena es conseguida a través del tratamiento de las siluetas de los segmentos obtenidos en la fase de segmentación. Sea $\{H_1, H_2, \ldots H_n\}$el conjunto de siluetas correspondientes a n objetos de la escena. Para una silueta genérica $H_j$ el problema que se plantea es saber cuáles de sus puntos pertenecen a la silueta real (o no ocluida) del objeto y cuáles no. El resultado será una partición de $H_j$, $\{H_j^1, H_j^2, \ldots H_j^q\}$, donde se alternen los partes reales y falsas de silueta. El algoritmo se realiza en dos pasos:

I. Se define $H_j$ como una secuencia de *segmentos aislados (I)* o *conectados (C)*. $H_j$ puede expresarse entonces como:

$$H_j = \{H_j\}_I \cup \{H_j\}_C \qquad (1)$$

Un segmento es *aislado* si todos sus puntos están suficientemente lejos de cualquier otra silueta de la escena. Formalmente,

$$H_j^m \in \{H_j\}_I \Leftrightarrow \forall p \in H_j^m / d(p,q) > l, \quad \forall q \in H_i \; i \neq j$$

donde $d$ es la distancia euclídea en la imagen y $l$ es un límite impuesto (en nuestro caso $l<10$ píxeles). Es obvio que un *segmento conectado* coincidirá o estará muy próximo a otro segmento que pertenezca a cualquier otra silueta. Formalmente:

$$H_j^m \in \{H_j\}_C \Leftrightarrow \forall p \in H_j^m, \exists q \in H_i, i \neq j / d(p,q) < l$$

En resumen, en esta etapa del proceso se trata de encontrar las partes de siluetas que están en contacto con otros objetos y las que no lo están. La información de las parejas de *segmentos conectados* se almacena para un proceso posterior.

II. Se clasifica $H_j$ en *silueta real (R)* o *falsa (F)*. El objetivo es descubrir qué partes de las siluetas ocluyen a otras *(R)* y cuáles son partes ocluidas *(F)*. Para las primeras:

$$\{H_j\}_R = \{\{H_j\}_I\}_R \cup \{\{H_j\}_C\}_R \qquad (2)$$

Es claro que si el segmento es aislado, entonces se corresponde con una parte de silueta real, es decir $\{\{H_j\}_I\}_R = \{H_j\}_I$, pero lo contrario no siempre es cierto. Por tanto, sólo el conjunto de los segmentos conectados $\{H_j\}_C$ debe ser clasificado como reales o falsos. Para hacer esto se utiliza la componente Z' (profundidad del píxel) y se compara la profundidad media de cada segmento conectado de $H_j$ con la profundidad media de su segmento conectado asociado (que pertenecerá a otra silueta). Comparando todas las parejas de segmentos conectados se define $\{\{H_j\}_C\}_R$. De esta forma puede evaluarse completamente la expresión (2). La figura 5 b) muestra el resultado de esta evaluación, donde cada silueta de cada objeto ha sido segmentada en partes reales y falsas (marcadas en diferente color). Notar que el objeto 3 no es ocluido y toda su silueta es real.

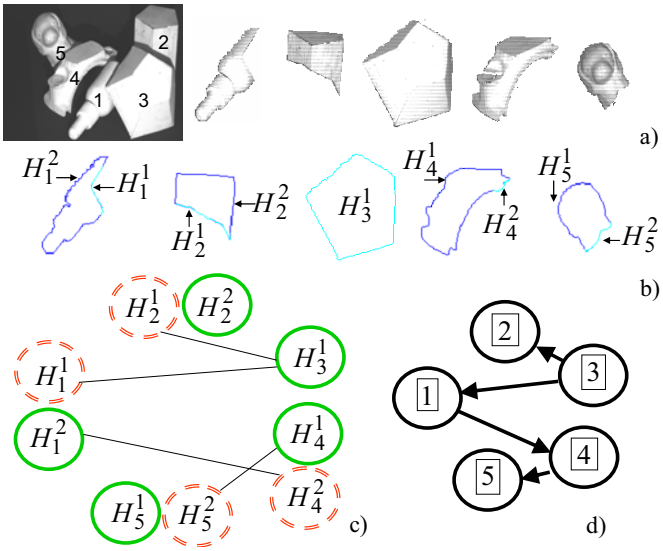



**Fig 5**. a) Segmentos obtenidos en la imagen de intensidad. b) Definición de siluetas *reales* y *falsas*. c) Grafo de conectividad entre siluetas d) Grafo de oclusión *G*.

Al final del proceso tenemos información suficiente para poder construir un grafo direccional que muestra la posición relativa de cada objeto de la escena en términos de oclusión. La figura 5 muestra el gráfico obtenido para la escena ejemplo. En la parte c) se enlazan las partes de siluetas que son *conectadas* y se representan partes

reales con trazo continuo y falsas con trazo cortado. En la parte d), se representa el grafo de oclusión *G* donde los objetos ocluidos son aquellos a los que llegan flechas que parten de los objetos oclusores.

Este tipo de información proporciona un punto de partida para etapas posteriores en el campo del análisis y la interpretación de escenas tridimensionales. En concreto, será muy útil para tareas de interacción de robots en la escena como veremos en la próxima sección.

## 4 Resultados de análisis en escenas complejas

Los procesos de segmentación y obtención de grafos de situación han sido realizados sobre un conjunto de 15 escenas utilizando un sensor de rango *Gray Range Finder*. Este sensor proporciona las coordenadas (x,y,z) de los puntos de la escena con un error inferior a 1 mm. Las escenas sobre las que se han probado los algoritmos están compuestas por varios objetos (desde 4 a12 objetos) colocados de manera desordenada. En ellas, los objetos están situados sin ninguna restricción de posicionamiento, presentan partes ocluidas por ellos mismos o por otros objetos de la escena y están todos pintados del mismo color. Por lo tanto se trata de imágenes no texturadas.

Como aplicación del análisis mediante grafos de situación, se ha establecido un algoritmo que ayuda a un robot manipulador a establecer un orden de acción sobre los objetos de la escena. La acción consistirá en la extracción de cada objeto de la escena en la dirección de visión de modo que se eviten colisiones. Utilizando el grafo de situación se han establecido un algoritmo de búsqueda basado en una heurística de prioridad expropiatoria. Cuando un objeto es extraído el grafo G es actualizado y se busca el próximo objeto de la lista. El orden de prioridades es el siguiente:

Criterio 1: mínimo(Número de partes ocluidas en el objeto)
Criterio 2: mínimo(Distancia a la cámara)
Criterio 3: máximo(Número de oclusiones que produce)

Las figuras 6 y 7 ilustran los resultados del análisis de la escena y del algoritmo de extracción para las escenas de mayor grado de dificultad. El esquema de segmentación distribuida obtenido una vez completada la etapa de segmentación aparece en la parte a). En ella se ve que las *hojas* finales del árbol son los objetos de los que consta la escena. Estos objetos se corresponden con los segmentos separados de la imagen de intensidad que se muestra en b). Se presentan etiquetados las siluetas extraídas y la clasificación de las siluetas en *reales* y *falsas*. Las *siluetas falsas* aparecen marcadas en la figura 6 y con diferente color en la figura 7. En la parte c) se ofrece el grafo de oclusión de la escena, en el que los objetos que conforman la escena están relacionados según el criterio de oclusión. Los objetos ocluyentes dirigen enlaces a los objetos ocluidos.

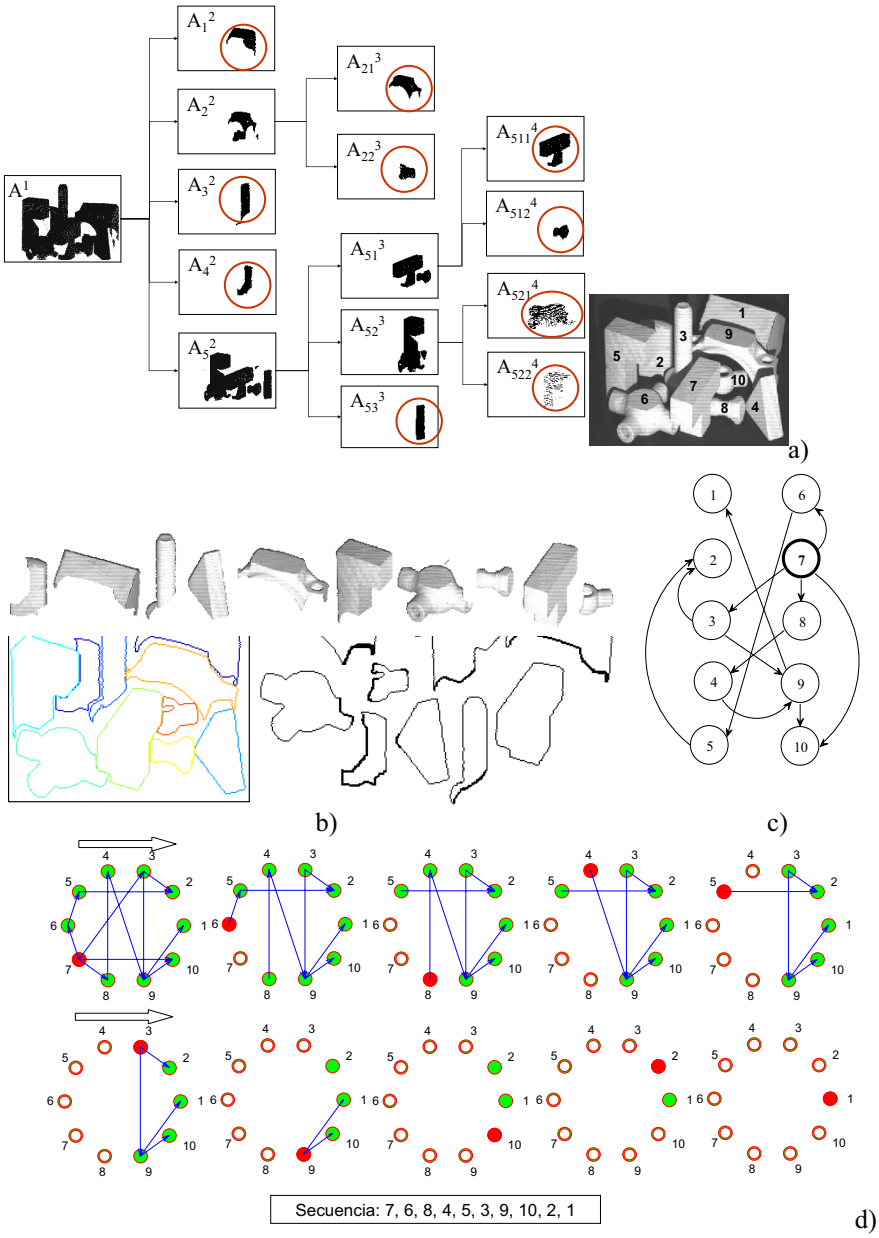
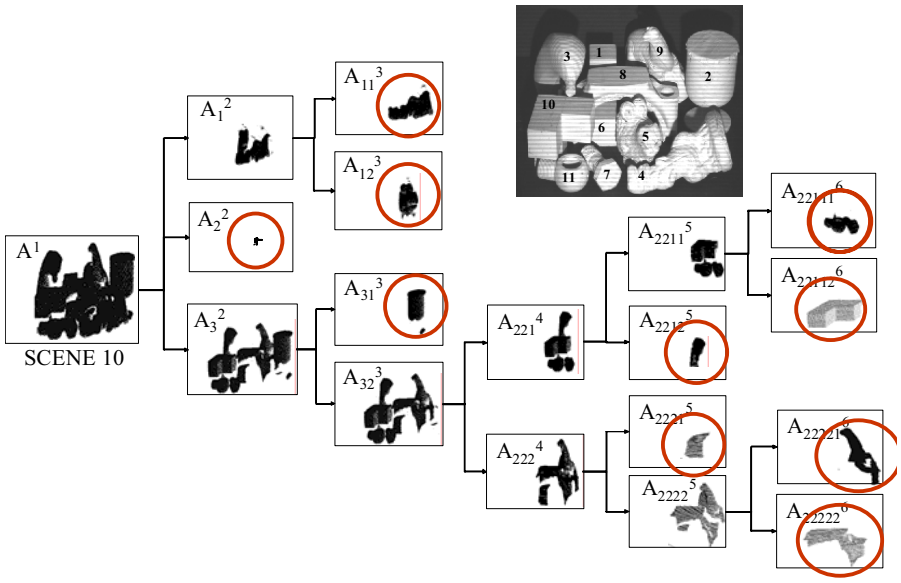



**Fig.6:** a) Arbol de segmentación de la escena nº 9. b) Segmentos en la imagen de intensidad, siluetas extraídas de los objetos y clasificación de siluetas. Las siluetas *falsas* están resaltadas c) Gráfico de oclusión de la escena, d) Secuencia de extracción de objetos y actualización del grafo de oclusión.

Pueden aparecer casos extremos en los que la clasificación de la silueta real sea errónea. Este caso se da en la figura 7 donde el objeto número 3 es etiquetado como objeto no ocluido. La parte realmente ocluida de la silueta de este objeto es etiquetada como silueta real ya que la distancia a la silueta conectada en el objeto 10 supera el umbral *l* introducido en la sección 3. Efectivamente, en ese par de siluetas conectadas, existe una distancia apreciable debido a que el objeto 10 proyecta una sombra sobre el objeto 3. Esta sombra es tanto mayor cuanto mayor es el gradiente en profundidad de ambos objetos. Por lo tanto el motivo final es que existe una separación apreciable en la dirección de la cámara entre ambos objetos. En la figura 7 b) se muestra un detalle en el que se aprecia esta separación. Este tipo de errores es difícilmente evitable en estas escenas altamente complejas y constituye una línea de investigación actual.

En la parte c) se muestra la secuencia de extracción de objetos mostrando la actualización de los grafos de oclusión a medida que los objetos son extraídos. En este sentido el error cometido con el objeto 3 de la figura 7 hace que este objeto sea extraído en tercer lugar de la secuencia (Figura 7 d)).




a)

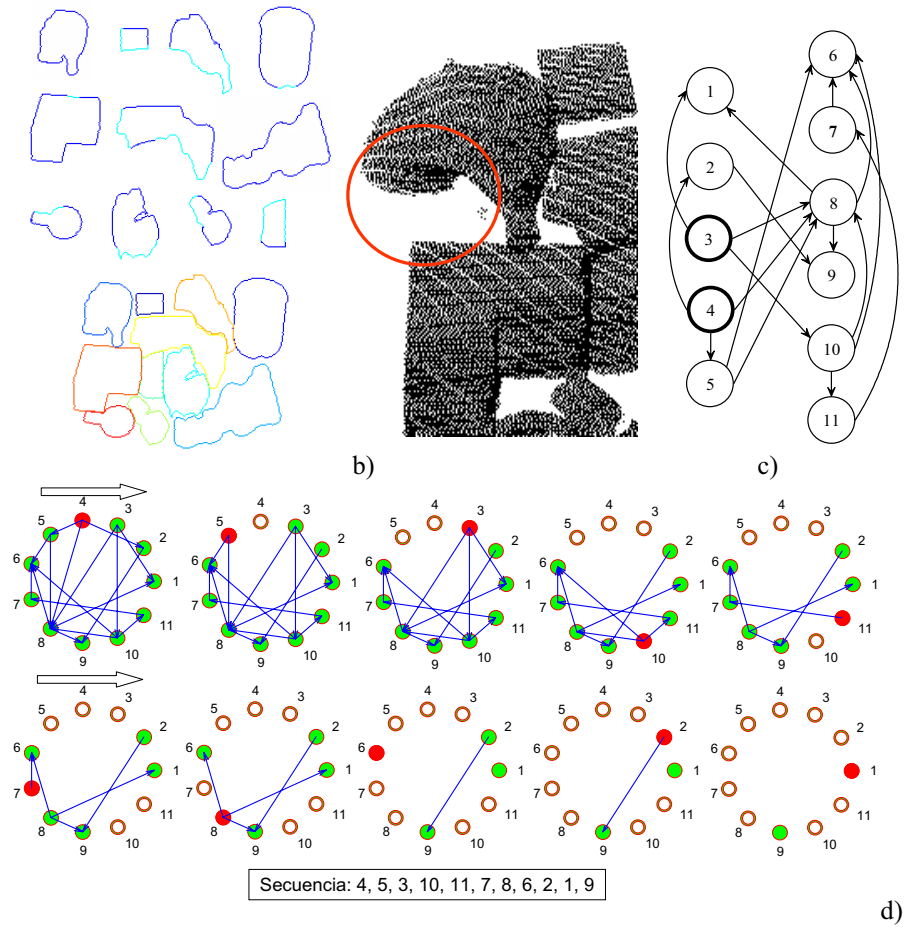



**Fig.7.** a) Arbol de segmentación de la escena nº 10. b) Siluetas extraídas de los objetos, clasificación de siluetas (las siluetas reales y falsas aparecen en distinto color) y detalle de separación entre los objetos 3 y 10. c) Gráfico de oclusión de la escena, d) Secuencia de extracción de objetos y actualización del grafo de oclusión.

## 5 Conclusiones

Este trabajo presenta una estrategia nueva para análisis de escenas complejas en entornos tridimensionales. En primer lugar, proponemos una solución general para la segmentación de la escena tridimensional que evita la mayoría de las restricciones impuestas en otros métodos, algunas de las más comunes podrían ser: necesidad de disponer de varias vistas de la escena, no se permiten oclusiones, los objetos no pueden estar en contacto, los objetos deben tener color o textura, o existen restricciones de forma. Nuestra técnica no requiere ninguna de estas restricciones.

El método se fundamenta en la aplicación de una estrategia nueva basada en la selección de puntos de vista que nos permite explorar y segmentar eficientemente los datos 3D proporcionados por el sensor. En segundo lugar, se ha establecido un procedimiento que permite, a través del estudio de siluetas de los segmentos extraídos, conocer las relaciones de oclusión entre los objetos de la escena. Como resultado, se logra obtener un grafo de oclusión o situación de la escena.
Se ha realizado una experimentación real para un conjunto de escenas complejas en nuestro laboratorio y se ha desarrollado una aplicación para ayuda de toma de decisiones de interacción de robots en la escena. Los resultados obtenidos permiten justificar la validez de nuestra estrategia para entornos robot-visión.

## Agradecimientos

Esta investigación ha sido financiada con el proyecto estatal PDI2002-03999-C02 financiado por el MEC y por el proyecto PBI05-028 financiado por la JCCLM.

## Referencias

[1] Hoover, A., Baptiste, G. J., Jiang, X., Flynn, P. J., Bunke, H., Goldgof, D. B., Bowyer, K, Eggert, D. W., Fitzgibbon, A., and Fisher, R. B., "An Experimental Comparison of Range Images Segmentation Algorithms", *IEEE Transactions on PAMI*, Vol. 18, no.7, 673-689, 1996.

[2] Burgiss, S.G., Whitaker, R.T., and Abidi, M.A., "Range Image Segmentation through Pattern Analysis of the Multiscales Wavelet Transfom", *Digital Signal Processing*, vol. 8, 267-276, 1998.

[3] Huang, J. and Menq, C.H., "·Automatic Data Segmentation for Geometric Feature Extraction From Unorganized 3-D Coordinate Points", IEEE Trans. on Robotics and Automation, Vol. 17, no. 3, 268-279, 2001.

[4] In Su Chang, Rae-Hong Park, "Segmentation based on fusion of range and intensity images using robust trimmed methods", *Pattern Recognition*, vol. 34, 1951-1962, 2001.

[5] Jiang, X., Bunke, H. and Meier, U., "High Level Feature Based Range Image Segmentation", *Image and Vision Comp.* 18, 817 - 822, 2000.

[6] Guoyu Wang, Zweitze Houkes, Guangrong Ji, Bing Zheng, Xin Li, "An estimation-based approach for range image segmentation: on the reliability of primitive extraction", *Pattern Recognition,* 36, 157-169, 2003.

[7] Lee, K. M., Mee, P. and Park, R.H., "Robust Adaptative Segmentation of Range Images", *IEEE Transactions on PAMI*, Vol. 20, no.2, 200-205, 1998.

[8] Super, B. J., Lu, H., "Evaluation of a hypothesizer for silhouette-based 3-D object recognition", *Pattern Recognition*, 36, 69-78, 2003.

[9] Bilodeau, G.-A., Berbegin, R., "Part segmentation of objects in real images", *Pattern Recognition* 35, 2913-2926, 2002.

[10] Serratosa, F., Alquézar, R., Sanfeliu, A., "Funtion-described graphs for modelling objects represented by set of attributed graphs", *Pattern Recognition* 36, 781-798, 2003.

[11] Cyr, C. M., Kimia, B. B., "3D Object Recognition Using Shape Similarity-Based Aspect Graph", *ICCV 2001*, 254-261.

[12] Sebastian, T. B., Klein, P. N., Kimia, B.B., "Recognition of shapes by Editing Shock Graphs", *ICCV 2001*, 755-762.

[13] Merchán, P., Adán, A., Salamanca, S., Cerrada, C., "3D Complex Scenes Segmentation from a Single Range Image using Virtual Exploration", *Lecture Notes in Artificial Intelligence* 2527, 923-932, 2002.

# Clasificación de cobertura vegetal usando wavelets

Omar Mayta[1], Rene Reynaga[2] y Luis Alonso Romero[2]

[1] Universidad Mayor de San Andrés, av. Villazón 1995, La Paz - Bolivia,
omarmc2005@yahoo.com
[2] Universidad de Salamanca, Plaza de la Merced s/n, Salamanca - España,
rene@tejo.fis.usal.es
lalonso@usal.es

**Resumen** Se muestra una técnica de clasificación de cobertura vegetal a partir de las texturas presentes en las imágenes digitales del terreno. La generación de características es realizada utilizando paquetes de onditas (wavelet packets). El procedimiento es descomponer la textura en sub-bandas de frecuencias altas y bajas hasta un nivel de resolución deseada. Las características seleccionadas se pasan a un sistema clasificador que usa una red neuronal tipo perceptrón multicapa.

## 1. Introducción

El reconocimiento de patrones es la disciplina científica cuyo objetivo es la clasificación de objetos en un cierto número de categorías o clases. Dependiendo de la aplicación esos objetos pueden ser imágenes, formas de ondas de señales o cualquier tipo de medidas que necesitan ser clasificadas (Theodoridis & Koutroumbas,1998)[12].

En el presente trabajo, estos objetos son muestras de texturas obtenidas de fotografías aéreas tomadas de determinadas áreas, y que pertenecen a ciertas categorías (bosque, matorral, agua, etc.) que están presentes en la imagen de la fotografía y necesitan ser clasificadas. Nos referiremos a esos objetos o muestras de textura de forma genérica utilizando el término patrones.

El reconocimiento de patrones se emplea en una amplia variedad de campos: reconocimiento de la voz, identificación de huellas digitales, reconocimiento de caracteres ópticos, identificación de la secuencia DNA, segmentación y clasificación de regiones de imágenes, etc. Está claro que una buena precisión en el reconocimiento de patrones será de gran utilidad. Esta disciplina es de gran importancia para los usuarios de GIS (Geographic Information Systems). A través de la determinación de patrones y objetos, las distintas superficies pueden ser delimitadas y clasificadas. Las imágenes pueden ser adquiridas a través de varias plataformas que incluyen fotografías aéreas, imágenes satelitales y de radar (Thurston J.,2002).

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 548-557, 2006.

Los métodos de análisis de texturas ha sido clasificados en cinco grandes categorías (Tuceryan M. & Jain A., 1998) [1]. Los métodos estadísticos describen la textura mediante un punto en un espacio multidimensional de características. Los métodos estructurales se concentran en la relaciones de los pixeles que conforman la textura. El presente trabajo puede estar enmarcado dentro de los métodos de procesamiento de señales, de esta manera, la textura es descompuesta en bandas frecuenciales (Randem & Husoy, 1999) [11]. Wickerhauser, Coifman y Meyer [17] proponen una forma de descomposición que proporciona una mejor localización en frecuencia y que denominan *wavelet packet*.

En el análisis de texturas, la principal dificultad es la identificación de características que potencien la diferencia entre clases. Este hecho ha estimulado la búsqueda de nuevas herramientas que permitan extraer características que varíen poco dentro de las regiones que contienen la misma textura. En este sentido, vale la pena mencionar trabajos de análisis de texturas basados en las estadísticas de la matriz de coocurrencias (Haralick et al, 1973; Davis et al, 1979) [9] [7]. Luego fueron planteados los campos aleatorios de Markov (Gaussian Markov Random Fields) (Cross & Jain, 1983; Chellappa & Chatterjee, 1985) [6] [4] y campos aleatorios de Gibbs (Derin & Elliot, 1987) [8]. El análisis de texturas mediante fractales, introducido por (Pentland, 1984) [14], destaca la correlación entre la dimensión fractal de una textura y su aspecto "tosco". Los métodos anteriores trabajan sobre el dominio espacial. Sin embargo, puede resultar más conveniente representar la imagen en otro dominio, de forma que se representen mejor las texturas. Tal es el caso del análisis multiresolución (Arivazhagan & Ganesan, 2002; Unser, M. & Eden, M., 1989; Bovik, A., Clark, M., Geisler, W.S., 1990; Chang, T. & Jay Kuo, 1993) [10] [15] [3] [5]que se basa sobre la transformada wavelets como una herramienta matemática desarrollada en los ochenta del siglo pasado. El intento del proyecto es demostrar la efectividad de esta herramienta para clasificar las propiedades de cobertura de la superficie terrestre.

## 2. Wavelet packets

La teoría de la Transformada de Wavelets esta bien documentada en diversos trabajos, por ejemplo en Wickerhauser [17]. La Figura 1 permite visualizar la extensión conceptual de los wavelet packets (Andréasson T., 2003) [2].

En ella se muestra que la transformada wavelet requiere de los espacios $V_m$ y $W_m$ [16]. Por lo tanto, se asume que se usa una base ortonormal. Cuando se estudia la transformada wavelet, se descompone el espacio $V_m$ en dos subespacios ortogonales $V_{m-1}$ y $W_{m-1}$. La notación $\Omega_m^k$ nos permite trabajar con estos subespacios.

Primero definimos $\Omega_0^0 := V_0$. La descomposición se denota como

$$\Omega_0^0 = \Omega_{-1}^0 \oplus \Omega_{-1}^1, \tag{1}$$

donde $\Omega_{-1}^0 = V_{-1}$ y $\Omega_{-1}^1 = W_{-1}$. De forma más general,

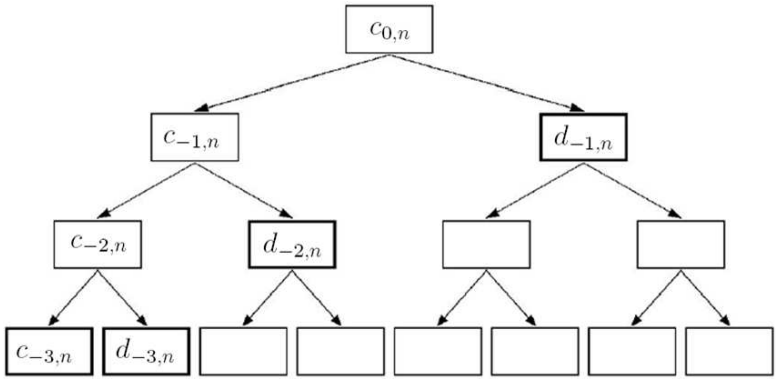



**Figura 1.** La transformada wavelet extendida a wavelet packet

$$\Omega_m^k = \Omega_{m-1}^{2k} \oplus \Omega_{m-1}^{2k+1}, \tag{2}$$

donde $\Omega_m^0 = V_m$ y $\Omega_m^1 = W_m$. La descomposición puede ser ilustrada en un árbol de wavelet packet, como se muestra en la Figura 2. Para cualquier árbol admisible se muestra fácilmente que los subespacios se generan a partir del espacio $\Omega_0^0 = V_0$.

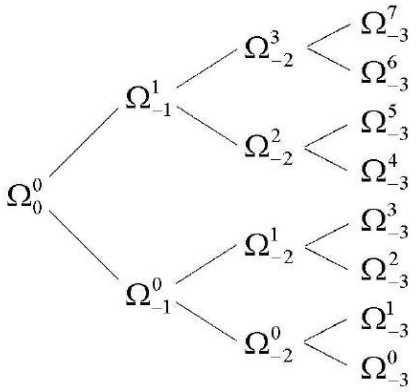



**Figura 2.** El árbol de wavelet packet para los espacios.

Se sabe que el espacio $V_m$ es generado por las funciones base $\{\phi_{m,n}\}$ y los espacios $W_m$ son generados por $\{\psi_{m,n}\}$. Los espacios $\Omega_m^k$ también son generados por las funciones base. Se usará la notación $\theta_{m,n}^k$ para estas funciones de base. Como se ha mencionado arriba, se puede identificar $\theta_{m,n}^0 = \phi_{m,n}$ y

$\theta^1_{m,n} = \psi_{m,n}$. La conexión entre funciones base en dos niveles consecutivos de la descomposición se da por una generalización de la función escala.

$$\theta^{2k}_{m,n} = \sqrt{2} \sum_i h_i \theta^k_{m+1,i+2n} \tag{3}$$

y por una generalización de la ecuación wavelet:

$$\psi(t) = \sum_n g_n \phi(t) = \sqrt{2} \sum_n g_n \phi(2t - n). \tag{4}$$

De esta manera,

$$\theta^{2k+1}_{m,n} = \sqrt{2} \sum_i g_i \theta^k_{m+1,i+2n}. \tag{5}$$

Cada wavelet packet puede ser expandida en las funciones base que generan los espacios en las salidas. Como fue el caso para la transformada wavelet, los coeficientes pueden ser usados en tal expansión, en lugar de las funciones base, para identificar la expansión. Se usará la notación $c^k_{m,n}$ para los coeficientes pertenecientes a la función base $\theta^k_{m,n}$.

## 3. Transformada wavelet discreta 2-D

El espacio $V_m$ es generado por la familia ortonormal

$$\phi^a_{m;n_1,n_2}(x,y) = \phi_{m,n_1}(x)\phi_{m,n_2}(y) \quad n1, n2 \in Z; \tag{6}$$

y las otras tres descomposiciones, son generadas por,

$$\begin{aligned} \psi^h_{m;n1,n2}(x,y) &= \phi_{m,n1}(x)\psi_{m,n2}(y) \\ \psi^v_{m;n1,n2}(x,y) &= \psi_{m,n1}(x)\phi_{m,n2}(y) \\ \psi^d_{m;n1,n2}(x,y) &= \psi_{m,n1}(x)\psi_{m,n2}(y). \end{aligned} \tag{7}$$

La familia de wavelets 2-D tiene una función escala (o wavelet padre) y tres funciones de wavelets madre. Como en wavelets 1-D, la función sirve para representar las aproximaciones y las wavelets madre sirven para representar los detalles, es decir, $\phi^a$ captura la parte uniforme, $\psi^v$ captura el detalle vertical, $\psi^h$ captura el detalle horizontal, y $\psi^d$ el detalle diagonal. El esquema del algoritmo para la descomposición en bandas de frecuencias aplicando filtros Pasa-Bajo $H^*$ y Pasa-Alto $G^*$ tanto a las filas como a las columnas adecuadamente, se muestra en la Figura 3.

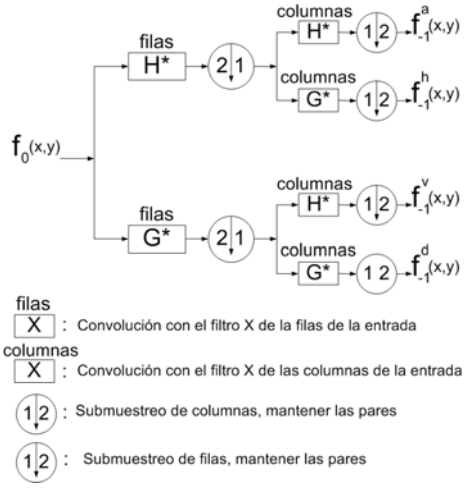



**Figura 3.** Esquema del algoritmo de filtrado por filas y columnas para la descomposición de una imagen

## 4. Aplicación de wavelet packets a las texturas de cobertura vegetativa

El análisis de texturas con los wavelet packets se procesó sobre la fotografía aérea del Área de Manejo Integrado de Cota-Pata en el Departamento de La Paz, Bolivia, la cual se muestra en la Figura 4.

En esta imagen se observa que la vegetación esta compuesta principalmente de texturas de bosque, matorrales y estepa altiplánica. También, se pueden apreciar otras coberturas como nieve y sombra. El procedimiento inicial que se ha aplicado para clasificar estas coberturas, es extraer porciones de texturas (patrones u objetos) de 12 por 12 píxeles de la imagen 4, como se observa en la Figura 5.

Para diseñar un sistema clasificador según (Theodoridis & Koutroumbas) [12], se siguen las fases que se muestran en la figura 6. Los coeficientes wavelet -de los niveles de resolución donde se concentra la energía- son entonces seleccionados como elementos del vector de características $x_i, i = 1, 2, ..., l$ donde

$$x = [x_1, x_2, ..., x_l] .$$

Utilizando los vectores base de Haar como filtros, la Figura 7 muestra la descomposición mediante el filtrado. La Figura 7 a) muestra la imagen descompuesta mediante trasformada wavelet discreta tradicional y luego la Figura 7 b) muestra la descomposición utilizando los wavelet packets.

En la Figura 7 b) se observan las pequeñas áreas que corresponden a los nodos terminales de la descomposición. Para conformar el vector de características se escogieron tres nodos (áreas), por prueba y error según la

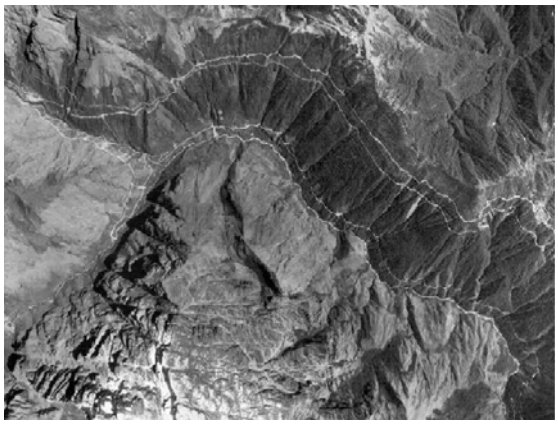


**Figura 4.** Fotografía aérea de Cota Pata - La Paz, Bolivia

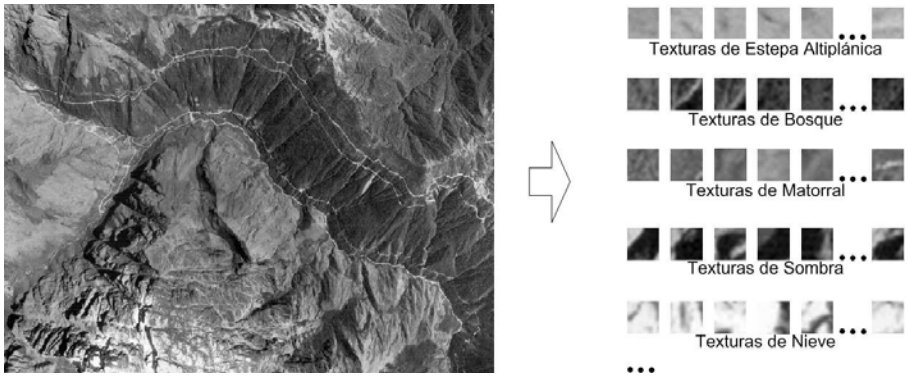



**Figura 5.** Diferentes patrones o texturas de terreno extraídas de la fotografía aérea para el diseño del sistema clasificador.

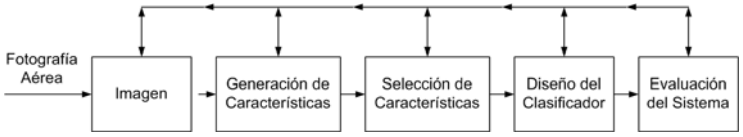



**Figura 6.** Fases en el diseño de un Sistema Clasificador

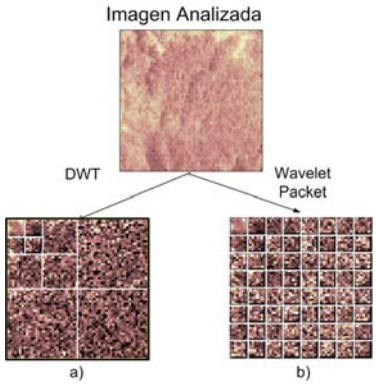



**Figura 7.** Textura de cobertura vegetal analizada, a) muestra la descomposición de la imagen usando la transformada discreta wavelet; b) muestra la descomposición mediante wavelet packet

calidad del clasificador. Los coeficientes se calculan mediante el uso de algoritmos implementados en las herramientas del $MATLAB$. Esta parte generalmente corresponde a la fase de selección de características que se muestra en la Figura 6. Los nodos seleccionados son los descriptores o rasgos de textura que mejor discriminan entre las clases de coberturas vegetales. Los tres nodos terminales seleccionados se observan en la Figura 8, identificadas mediante las letras a,b y c.

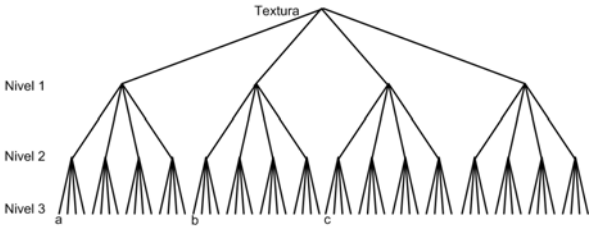



**Figura 8.** Nodos a,b y c seleccionados para formar el vector de características

Una vez que se tiene el conjunto de vectores de características, se pasa al la siguiente etapa, que es la fase del diseño del clasificador.

De todos los métodos para propósitos de clasificación, el que más éxito tuvo en nuestro caso -por su capacidad de generalización, y su robustez en la clasificación de datos de entrada con ruido- fueron las redes neuronales. Más precisamente, una red perceptrón de 3 capas, con 40 neuronas en la primera capa oculta, 20 neuronas en la segunda capa oculta y 16 neuronas en la capa de salida. El aprendizaje se realizó con el algoritmo de retropropagación. El presente trabajo

es una continuación natural de [13] con la selección de características modificada pero usando el mismo sistema clasificador basado sobre MLP. Otros clasificadores basados sobre RBF pueden entrenar más rápido pero generalizan peor.

### 4.1. Evaluación del sistema clasificador

De la imagen 4 se han obtenido 50,51,51,25 y 26 muestras de texturas para las categorías de bosque, matorral, estepa altiplánica, nieve y sombra respectivamente y se calculado la tasa de error. Los resultados se muestran en la tabla 1.

| **Categoría** | **Número de muestras** | **Procentaje (aciertos)** |
|---|---|---|
| Bosque | 50 | 96 % |
| Matorral | 51 | 100 % |
| Estepa altiplánica | 51 | 98 % |
| Nieve | 25 | 92 % |
| Sombra | 26 | 85 % |

**Cuadro 1.** Tasa de error para diferentes categorías

En áreas de vegetación, la clasificación de textura provee resultados precisos. Pero no así en áreas que presentan coberturas de sombra y nieve. Algunos ejemplos se muestran en la Figura 9.

## 5. Conclusiones

En este trabajo se ha mostrado como el uso de la herramienta wavelets packets permite extraer características que combinadas a la flexibilidad y capacidad de aprendizaje de las redes neuronales integran un sistema informático clasificador capaz de discriminar entre las diferentes texturas con un porcentaje de aciertos elevado.

Sobre estas bases, se ha construido un prototipo en $MATLAB$ con técnicas de GUI que entrega un entorno amigable, intuitivo y sencillo de utilizar. Sin embargo, al ser un prototipo interpretado, los tiempos de entrenamiento pueden tomar horas para imágenes del orden de Megapixeles. En futuras versiones, se pretende utilizar otras técnicas de extracción de características y de entrenamiento y con esto disminuir la complejidad computacional.

## Referencias

1. Tuceryan M.* & Jain A.**. *Texture Analysis.* World Scientific Publishing Co., *Department of Computer and Information Science, Indiana University - Purdue University at Indianapolis,**Computer Science Department, Michigan State University East Lansing, 1998.

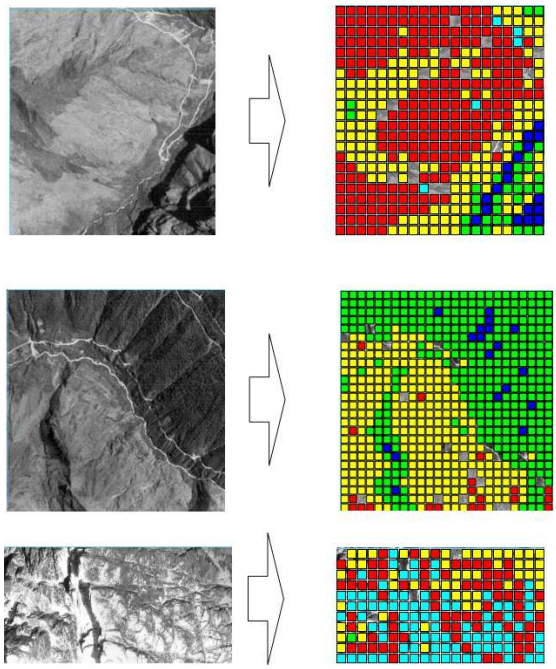


**Figura 9.** Clasificación de coberturas

2. Thomas Andréasson. *Signal Processing Using Wavelets in a Ground Penetrating Radar System*. Linköping University, 2003.
3. Clark M. Geisler W.S. Bovik, A. Multichannel texture analysis using localized spatial filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, 12:55–73, 1990.
4. R. Chellappa & S. Catterjee. Classification of textures using gaussian markov random fields. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 33:959–963, 1985.
5. T. & Jay Kuo Chang. Texture analysis and classification with tree-structured wavelet transform. *IEEE Transactions Image Processing.*, 2 (4):429–440, 1993.
6. Jain A.K. Cross, G.R. Markov random field texture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, PAMI-5 (1):25–39, 1983.
7. Johns S.A. Aggarwal J.K. Davis, L.S. Texture analysis using generalized co-occurrence matrices. *IEEE Transactions on Pattern Analysis and Machine Intelligence. PAMI-1*, pages 251–259, 1979.
8. Elliot H. Derin, H. Modeling and segmentation of noisy and textured images using gibbs random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, PAMI-9:39–59, 1987.
9. R. M. Haralick et al. Textural features for image classification. *IEEE Trans. on Systems Man Cybernat.*, 8(6):610–621, 1973.
10. S. Arivazhagan & L. Ganesan. Texture classification using wavelet transform. *Pattern Recognition Letters*, 24:1513–1521, 1984.
11. T. Randem & J. H. Husoy. Filtering for texture classification: A comparative study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(4), 1999.

12. Theodoridis S. & Koutroumbas K. *Pattern Recognition.* Academic Press, Departments of Informatics, University of Athens, 1998.
13. R. Aguilar, R. Cuarite, R. Reynaga y H. Bañados. *Reconocimiento de patrones a partir de imágenes aéreas.* En Reconocimiento de patrones con Redes neuronales. CYTED 2001, ISBN: 84-95721-07-4.
14. A. P. Pentland. Fractal-based description of natural scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, 6:661–672, 1984.
15. M. Unser, M. & Eden. Multiresolution feature extraction and selection for texture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, 2:717–728, 1989.
16. Wickerhauser M. V. *Adapted Wavelet Analysis from theory to software.* IEEE press, Piscataway, NJ, 1994.
17. Mladen Victor Wickerhauser. *Acoustic Signal Compression with Wave Packets.* Department of Mathematics, Yale University, 1989.

# Regresión logística con construcción de características mediante Boosting

Jesús Maudes y Juan J. Rodríguez

Lenguajes y Sistemas Informáticos, Universidad de Burgos
{jmaudes, jjrodriguez}@ubu.es

**Resumen.** El presente trabajo presenta una combinación de clasificadores o ensemble, constituido por uno de mayor precisión llamado fuerte y un conjunto de otros más simples llamados débiles, de manera que el conjunto de clasificadores débiles es obtenido mediante Boosting. La salida de los clasificadores débiles construye una serie de características que le son complementarias al fuerte, pues además de tomar estos nuevos atributos no renuncia a tomar los atributos originales del conjunto de datos. El resultado es un clasificador que en la mayoría de los casos es más preciso que el clasificador fuerte por si solo, pero que es capaz de mejorar el tratamiento que éste da a aquellas regiones del espacio de entrada que tienden a ser peor clasificadas, debido a la influencia de las nuevas características obtenidas mediante Boosting. En ese sentido, se presentan resultados utilizando árboles de una sola decisión (Decision Stumps) como clasificadores débiles y una regresión logística como fuertes.

*Palabras clave:* construcción de características, Boosting, combinación de clasificadores, aprendizaje automático

## 1 Introducción

Una combinación de clasificadores o ensemble es un conjunto de clasificadores que colaboran para la obtención de un método de clasificación final [10]. Las combinaciones de clasificadores más conocidas incluyen Bagging y Boosting. En ambos métodos la combinación se forma por varios clasificadores del mismo tipo, que llamaremos clasificadores base, que a través de una votación forman un clasificador final de mayor precisión [1, 4].

Bagging genera cada uno de los clasificadores base tomando aleatoriamente distintos subconjuntos del conjunto de entrenamiento, con lo que el resultado final no está excesivamente orientado a explorar los casos difíciles, pero sin embargo la obtención de los clasificadores base se puede paralelizar. Sin embargo, Boosting genera secuencialmente cada uno de los clasificadores base, de manera que el clasificador siguiente da más importancia a clasificar correctamente aquellas instancias que el clasificador anterior no ha podido clasificar bien. De esta manera, Boosting consigue explorar el espacio de entrada buscando aquellos casos más difíciles de clasificar obteniendo buenos resultados aun cuando los clasificadores base sean poco precisos.

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 558-563, 2006.

En Boosting la votación final de cada clasificador se ve ponderada en función de la estimación del error producido por cada uno de ellos. Una posible intuición sobre cómo mejorar Boosting podría venir de sustituir la votación por un método más sofisticado de combinar las salidas de los clasificadores base, de manera que la salida de unos clasificadores sirvan de entradas a otros, en lo que se conoce como topología en cascada [6]. Este trabajo presenta una instancia particular de dicha topología con el fin de mejorar los resultados de la regresión logística.

La *construcción de características* es el proceso mediante el cual se descubre información no presente sobre las relaciones de los atributos, aumentando el espacio de características al deducir o crear otras nuevas [7]. En este trabajo se utiliza esta aproximación, pues para ayudar a que el clasificador fuerte no pierda precisión respecto a cuando se utiliza por si solo, no solamente se le dan como atributos los que se obtienen de la salida de los débiles, sino que se mantienen los atributos del conjunto original. El resultado en buena parte de las ocasiones mejorará la precisión del clasificador fuerte en solitario, en tanto se ve beneficiado del potencial de Boosting para explorar aquellas regiones del espacio de entrada que tienen un tratamiento más difícil. Las características construidas se corresponderán con el vector de probabilidades que devuelven cada uno de los clasificadores débiles. En definitiva, lo que se propone es utilizar un método de Boosting para construir nuevas características que serán útiles a un clasificador ya de por si fuerte.

Nótese que la construcción de características no presentes en el conjunto de datos es distinta a la (i) extracción de características (que persigue encontrar un conjunto mínimo de nuevas características a través de alguna transformación y según un criterio de optimización), (ii) la selección de características (con objeto de eliminar aquellas que resulten redundantes), y de (iii) la combinación de características (con objeto de obtener grupos de las mismas que faciliten la tarea de clasificación).

Existen aproximaciones de selección de características con Boosting [15, 3], y de combinación de características con Boosting [17]. También existen aproximaciones de extracción de características con Boosting. Así, [11] utiliza Boosting para obtener un conjunto optimizado de proyecciones que reducen la dimensionalidad del problema y minimizan el error de clasificación al aplicarlas como entrada de otro clasificador.

Este artículo se organiza como sigue. En la sección 2 se describe pormenorizadamente el algoritmo para la obtención de una combinación de clasificadores genérica que utilice la construcción de características mediante Boosting. La sección 3 está dedicada a la validación experimental. La sección 4 presenta las conclusiones.

## 2 Construcción de características mediante Boosting

El método que se propone consiste en tomar un clasificador fuerte y proveerle tanto los atributos correspondientes al conjunto de datos como además, los atributos tomados de la salida de varios clasificadores débiles generados mediante Boosting. De esta manera, se añade a la potencia que de por si tiene el método fuerte, la capacidad que tiene Boosting de explorar aquellas regiones del espacio pobladas por instancias que son difíciles de clasificar. Utilizando clasificadores computacionalmente simples

como clasificadores débiles, el incremento del coste computacional sobre la utilización del método final por separado no debe ser muy grande.

Así pues, el resultado final es una combinación de clasificadores formado por varios clasificadores débiles obtenidos mediante Boosting y que toma como entradas las propias del conjunto de datos en cuestión, más un clasificador fuerte que toma como entradas las del conjunto de datos más la salida de los clasificadores débiles.

Visto de otra forma, se ha sustituido la decisión mediante voto ponderado que hace Boosting, por una decisión mediante un método entrenado, en este caso una regresión logística, que además tiene en cuenta los valores de los atributos.

$$\textit{FASE}\ 1$$

$\textit{Dados}\ (x_1, y_1), \ldots, (x_m, y_m) / x_i \in X,\ y_i \in Y$

$\textit{Inicializar}\ D_1(i) = 1/(m)$

$\textit{Para}\ \ t = 1, \ldots, T:$

$\textit{Entrenar clasificador base utilizando la distribución}\ D_t$

$* \ \textit{Obtener clasificador base}\ h_t: X \to Y$

$* \ \textit{Seleccionar}\ \alpha_t \in \mathbb{R}$

$* \ \textit{Actualizar}$

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

$\textit{donde}\ Z_t\ \textit{es un factor de normalización}$

$(\textit{elegido de modo que}\ D_{t+1}\ \textit{sea una distribución}).$

$$\textit{FASE}\ 2$$

$\textit{Entrenar el clasificador final}\ H(.)\ \textit{con}$

$(x_1, h_1(x_1), \ldots h_T(x_m), y_1), \ldots, (x_m, h_1(x_m), \ldots, h_T(x_m), y_m)$

**Fig 1.** Entrenamiento de un clasificador final utilizando construcción de características mediante Boosting

El algoritmo de entrenamiento por tanto tiene dos pasos fundamentales:

1. Obtener los *T* clasificadores débiles mediante Boosting. Este paso puede tomase tal cual de una implementación de Boosting.
2. Entrenar el clasificador fuerte a partir del conjunto de entrenamiento, pero concatenando a cada tupla de entrada las salidas de los clasificadores débiles.

La figura 1 muestra el algoritmo. En la primera fase las T iteraciones van obteniendo cada uno de los clasificadores débiles, los cuales son distintos en cuanto a que cada uno da más importancia a las instancias mal clasificadas por los anteriores. Para ello, las instancias llevan asociadas un peso que se recalcula cada vez que se genera un nuevo clasificador débil, de manera que el peso va aumentando en las instancias mal clasificadas. El algoritmo de entrenamiento del clasificador débil ha de ser sensible a los pesos de las instancias, de manera que intente no errar en aquellas con mayor peso. Esta idea ya está implementada en cualquier variante del algoritmo de Boosting [12, 13], y el mecanismo de cálculo de los pesos se ha tomado tal cual lo hace Adaboost M1 [14], la variante típica de Boosting para problemas multiclase.

Una vez obtenidos los clasificadores débiles, la obtención del clasificador fuerte es directa, basta entrenarlo con las instancias de conjunto de datos concatenadas con las salidas de los clasificadores débiles. En la implementación que se ha realizado se ha tomado como salida de los clasificadores débiles un vector de distribución que tiene tantas componentes, como clases tiene el problema, de manera que cada componente $c_i$ expresa la probabilidad de que la instancia evaluada pertenezca a la clase i-ésima.

## 3 Validación experimental

Para validar la construcción de características mediante Boosting se han utilizado 32 conjuntos de datos correspondientes al repositorio UCI [2]. Para cada conjunto de datos y método se realizaron 10 validaciones cruzadas estratificadas de 10 grupos.

Para la implementación del algoritmo se ha utilizado una regresión logística como clasificador final, y 10 clasificadores base obtenidos mediante Boosting. Como clasificador base se tomaron Decision Stumps, pero al tratarse de clasificadores binarios y al ser la mayoría de los conjuntos de datos multiclase, no se han utilizado directamente, sino que para cada clasificador base se construyen tantos Decision Stumps como clases, utilizando la estrategia denominada "uno contra todos".

Se han comparado los resultados con:

1. AdaBoost M1 con clasificadores base Decision Stumps utilizando también la estrategia uno contra todos, por ser M1 un método de Boosting multiclase, y por utilizar el mismo tipo de clasificadores base que el constructor de características propuesto. Se muestran resultados para 10 y 100 clasificadores base.
2. Regresión Logística. Dado que la estrategia uno contra todos discretiza las entradas, es más justo hacer la comparación con entradas discretizadas. Por ello, también se proveen resultados de este clasificador pasándole como entrada los conjuntos de datos previamente discretizados.

Para la validación se ha utilizado WEKA [16], siendo las implementaciones de AdaBoost M1, Decision Stump y Regresión Logística las que provee este entorno (regresión implementada mediante el método *Simple Logistic* [9]). La discretización para la regresión se efectúa según [5].

El cuadro 1 indica cuántas veces uno de los métodos contra los que se valida, gana, empata o pierde frente a la regresión con constructor de características. En la primera columna se muestran los resultados absolutos, mientras que en la segunda columna se tiene en cuenta el *test-t estadístico remuestreado corregido* [8] considerando los 100 valores (10 repeticiones de validación cruzada con 10 grupos) obtenidos por cada método y conjunto de datos. Se observa que el método con construcción de características es mejor que el resto de los considerados.

**Cuadro 1.** Comparación de tasas de aciertos. Número de victorias, empates y derrotas, de la Regresión Logística con constructor de características, sobre el resto de métodos. Columna 1: resultados en valor absoluto, Columna 2: resultados de acuerdo al test estadístico utilizado

| | | |
|---|---|---|
| 17/1/14 | 4/27/1 | Regresión logística discretizada |
| 18/3/11 | 6/26/0 | Regresión logística |
| 26/0/6 | 15/17/0 | Adaboost M1 de 100 decision stumps uno contra todos |
| 30/0/2 | 16/16/0 | Adaboost M1 de 10 decision stumps uno contra todos |

El cuadro 2 muestra el ranking entre los cinco métodos calculado a partir de la diferencia entre el número de victorias y derrotas significativas según el test al emparejar todos los métodos. Nuevamente el método con constructor es superior.

**Cuadro 2.** Ranking de los métodos de acuerdo con la diferencia entre el número de victorias y derrotas significativas

| Método | Victorias menos Derrotas | Victorias | Derrotas |
|---|---|---|---|
| Regresión logística con Constructor de Características | 40 | 41 | 1 |
| Regresión logística discretizada | 24 | 32 | 8 |
| Regresión logística | 18 | 30 | 12 |
| Adaboost M1 100 iteraciones | -35 | 7 | 42 |
| Adaboost M1 10 iteraciones | -47 | 1 | 48 |

## 4 Conclusiones

En el presente trabajo se presenta una combinación de clasificadores en topología de cascada en la que los clasificadores débiles son generados a través de Boosting. Los clasificadores débiles actúan como constructores de características, pues el clasificador fuerte o final además de los nuevos atributos sigue tomando como entradas todos los atributos del conjunto de datos de partida. Con ello, se pretende que la capacidad exploratoria de Boosting de aquellas regiones del espacio de entrada que son difíciles de clasificar, aporte una mejora a la utilización del algoritmo final por si sólo.

Se ha probado experimentalmente este algoritmo utilizando 10 clasificadores base, cada uno de ellos formado por tantos Decision Stumps como clases, y una regresión logística como clasificador final obteniendo buenos resultados al compararlo con la regresión logística en solitario y con AdaBoost M1 aun utilizando este último hasta 100 Decision Stumps por clase.

Debido a que los Decision Stumps trabajan con problemas binarios, en la implementación se ha recurrido a generar un Decision Stump por cada clase (estrategia uno contra todos). Por ello, podría pensarse que los resultados de la regresión no eran comparables, en tanto la mencionada estrategia tiene el efecto de discretizar las entra-

das; por lo que, se ha completado satisfactoriamente la validación con pruebas sobre regresión logística discretizada.

## Agradecimientos

Este trabajo ha sido financiado parcialmente por los proyectos DPI2005-08498 del MEC y VA088A05 de la Junta de Castilla y León.

## Referencias

1. E. Bauer, R. Kohavi. An Empirical Comparision of Voting Classification Algorithms: Bagging, Boosting and Variants. Machine Learning, 36, pp. 105-142, 1999.
2. C. L. Blake and C. J. Merz. UCI repository of machine learning databases, 1998. http://www.ics.uci.edu/mlearn/MLRepository.html.
3. S. Das, Filters, Wrappers and a Boosting-Based Hybrid for Feature Selection, Proceedings of the Eighteenth International Conference on Machine Learning, p.74-81, 2001.
4. T. G. Dietterich, Ensemble Methods in Machine Learning, in Lecture Notes in Computer Science, vol 1857, pp. 1-15, 2000.
5. U. M. Fayyad and K.B. Irani, On the Handling of Continuous-Valued Attributes in Decision Tree Generation. Machine Learning, 8(87-102), 1992.
6. J. Gamma and P. Brazdil. Cascade Generalization. Machine Learning, 41 (315-343), 2000.
7. H. Liu, L. Yu and H. Motoda. Feature Extraction, Selection and Construction. In The Handbook of Data Mining. Nong Ye Eds., pp 409-422, 2003.
8. C. Nadeau and Y. Bengio. Inference for the Generalization Error. Machine Learning, 52(239-281), 2003.
9. N. Landwehr, M. Hall, and E. Frank. Logistic model trees. In ECML, pp 241–252, 2003.
10. L. I. Kuncheva. Combinig Pattern Classifiers, Methods and Algorithms. Wiley Interscience. 2004.
11. D. Masip, J. Vitrià. Boosted discriminant projections for nearest neighbor classification. Pattern Recognition 39 (2006) 164-170.
12. R. E. Schapire. Using output codes to boost multiclass learning problems. In 14th International Conference on Machine Learning (ICLM-97), pp 313-321, 1997.
13. R. E. Schapire. A brief introduction to Boosting. In Thomas Dean, editor, 16th International Joint Conference on Artificial Intelligence (IJCAI-99), pp 1401-1406. Morgan Kaufmann, 1999.
14. R. E. Schapire, The Boosting approach to machine learning: An overview, in MSRI Workshop on Nonlinear Estimation and Classification, (2002). http://www.cs.princeton.edu/schapire/papers/msri.ps.gz.
15. P. Viola, M. Jones. Robust real-time object detection. In IEEE Workshop on Statistical and Computational Theories of Vision
16. H. Witten and E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, Morgan Kaufmann, 2nd edn., 2005. http://www.cs.waikato.ac.nz/ml/weka.
17. X-C Yin, C-P Liu, Z Han, Feature combination using Boosting. Pattern Recognition Letters. 26(2005) 2195-2205.

# Extracción de líneas melódicas a partir de imágenes de partituras musicales

Ángel Sánchez, Juan J. Pantrigo y José Ignacio Pérez

Universidad Rey Juan Carlos, Madrid, Spain
{angel.sanchez, juanjose.pantrigo, joseignacio.perez}@urjc.es

**Resumen** El objetivo de este trabajo ha sido el desarrollo de una herramienta automática para la segmentación de partituras musicales no manuscritas, que permite extraer pentagramas, compases y duración y tono de las notas incluidas en ella. Una vez procesada la imagen, un segundo objetivo es obtener líneas melódicas y, en ellas, ciertos patrones musicales proporcionados por el usuario. El método elegido para el análisis se ha derivado de diferentes propuestas encontrados en el estado del arte de los sistemas OMR (*Optical Music Recognition*) combinadas con aportaciones propias de los autores. Los experimentos se han realizado usando diversas piezas de música de contrapunto (invenciones) de J. S. Bach.

## 1. Introducción

La música computacional (*computer music*) es un área interdisciplinar de creciente interés. Su ámbito de aplicación engloba tareas tan diversas como reconocimiento de estilos musicales [1], clasificación de piezas por estilos musicales [2], reconocimiento de notas en partituras musicales [3], búsqueda de similitudes [4], estudios relacionados con el ritmo [5], generación automática de piezas musicales [6] o reconocimiento de elementos en partituras musicales (OMR) [7]. La mayor parte de los sistemas que utilizan una representación simbólica de la imagen escaneada de una partitura musical coinciden en una serie de pasos comunes. Estos comprenden: detección de las líneas del pentagrama, segmentación de elementos musicales y análisis de la imagen segmentada. Estas son las etapas típicas de un sistema de tratamiento digital de imágenes. La mayoría de los autores considera la detección de las líneas de los pentagramas como el paso preliminar. En primer lugar, se analizan los ángulos de inclinación de los pentagramas para rotar las partituras en caso de que sea necesario. Un gran número de métodos se opta por identificar las líneas, empleando la transformada de Hough en la mayoría de los casos (con el consiguiente consumo de recursos debido a la complejidad de dicho algoritmo función), para obtener las ecuaciones de las rectas de la imagen, y posteriormente eliminarlas. Otro método consiste en el análisis de la proyección horizontal de la imagen. El histograma resultante

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 564-569, 2006.

presenta picos debidos a la concentración de píxeles que conforman las líneas del pentagrama.

Una vez eliminadas las líneas de los pentagramas, en la partitura permanecen símbolos aislados que han de ser identificados. Estos símbolos pueden ser notas musicales, caracteres (generalmente numéricos y para cuyo reconocimiento se puede hacer uso de un motor OCR, *Optical Character Recognition*), texto o elementos ornamentales de la partitura. Puesto que, generalmente, sólo interesa la información musical contenida en el pentagrama, se procura eliminar los caracteres y otros elementos. Es en esta etapa donde más diversidad de métodos existe. La mayor parte de éstos optan por realizar una búsqueda de las componentes conexas para segmentar y buscar patrones en partituras musicales 15, usando principalmente grafos de adyacencias de líneas (LAG). Una vez elaborado el grafo se procede al estudio de cada una de las componentes conexas para caracterizarlas. Otra alternativa es el uso de plantillas, que es el método propuesto en este trabajo, debido a su mayor simplicidad y los buenos resultados que proporciona. En algunos casos, se emplean conjuntamente ambos métodos. El formato de las partituras se tiene en cuenta para la identificación de la información. Hay algunas características comunes en toda partitura, como por ejemplo, la ubicación de las claves, que siempre aparecen en el lado izquierdo del primer compás de un pentagrama. El uso de bases del conocimiento musical simplifica el proceso de segmentación. Finalmente, una vez identificados los elementos que componen la partitura, la siguiente etapa consiste en asignar una notación determinada que permita establecer la relación entre la partitura original y la representación simbólica de la misma.

## 2. Metodología propuesta

En esta sección se describe la metodología desarrollada para la segmentación y análisis de partituras musicales. En ella se pueden distinguir cuatro etapas fundamentales: preproceso de la imagen, identificación de los compases, búsqueda de notas y determinación de su tono y duración.

### 2.1. Preproceso de la imagen

En esta etapa se acometen dos tareas fundamentalmente: binarización de la imagen de la partitura y eliminación de información innecesaria para su posterior análisis. Para convertir una imagen en niveles de gris (256 niveles de gris) en una imagen binaria, se calcula el umbral a partir del cuál un valor de gris será considerado como blanco o negro en la representación binaria. El cálculo de dicho umbral se realiza empleando el método de Otsu [8], que elige un umbral para reducir al mínimo la variación de los rangos en los que un pixel será considerado como negro o blanco en la imagen binaria resultante. Una vez calculado dicho umbral y transformada la imagen original a su representación binaria, se elimina la información innecesaria que aparece en la partitura. Se considera información irrelevante el número de página y el título de la pieza

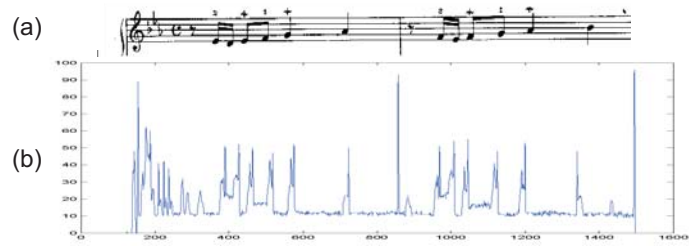



**Figura 1.** Etapa de identificación de los compases: (a) imagen original del pentagrama y (b) histograma de proyección vertical.

musical. Para localizar estas regiones de la imagen se ha optado por caracterizar la morfología del histograma de la proyección horizontal de toda la partitura.

### 2.2. Identificación de los compases

Esto se consigue mediante sucesivos refinamientos en el análisis de la imagen binaria resultante del preproceso. En primer lugar, se detectan los sistemas es decir, grupos de pares de pentagramas, puesto que se analizan piezas escritas para piano. Para ello, nos basamos nuevamente en la estructura del histograma de la proyección horizontal de la partitura. Una vez obtenidas las coordenadas de los sistemas, se prosigue en la tarea de detección de los pentagramas, utilizando nuevamente el histograma horizontal de la partitura. Existe una pequeña zona de separación entre un pentagrama y el siguiente de cada par considerado. Basándonos en dicha separación, pueden establecerse las dimensiones de la ventana que delimita a un determinado pentagrama.

Por último, se procede a la extracción de los compases de cada pentagrama. En una partitura, los compases vienen separados unos de otros mediante unas líneas verticales, que producen un pico en un histograma vertical del pentagrama. En la Figura 1, es posible identificar las barras de separación de los compases de un pentagrama a través de los picos que producen en el histograma.

### 2.3. Búsqueda de notas

Esta fase es la más compleja del proceso, debido a la exhaustividad de las exploraciones que se realizan sobre cada uno de los compases en la búsqueda de las notas musicales. El procedimiento desarrollado traza, en primer lugar, una serie de "pistas virtuales" en el compás, que recorren horizontalmente las líneas y los espacios del pentagrama. Sobre dichas pistas, se desplaza posteriormente una máscara de la cabeza de una nota musical. El uso de dichas pistas se justifica por los siguientes motivos:

1. Eficiencia: en lugar de recorrer una a una las posiciones de la matriz que contiene la imagen correspondiente al compás, sólo se buscarán las notas

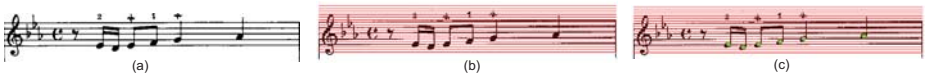



**Figura 2.** Etapa de identificación de las notas: (a) imagen original del pentagrama, (b) Posición de las pistas virtuales, y (c) resultado de la detección de notas (se marca coloreada la cabeza de las notas).

en unas determinadas filas en las que la probabilidad de hallar una nota es mayor, puesto que la disposición de las notas en un compás está limitada.
2. Efectividad: el uso de las pistas virtuales permite asignar el tono de una nota de modo sencillo.

Para elegir estas pistas, se utiliza el histograma de la proyección horizontal del compás de la Figura 2(a). A continuación, se trazan las pistas virtuales. Además se estiman unas líneas intermedias, que recorren los espacios. El resultado se muestra en la Figura 2(b). La siguiente tarea consiste en recorrer cada una de las pistas con la máscara, cuyo tamaño se determina experimentalmente. Durante este desplazamiento, se compara la región de la imagen de tamaño igual a la máscara elegida que rodea al punto en el que se desea saber si hay una nota, y la máscara de la cabeza de una nota musical. Se estudia el índice de solapamiento entre ambas regiones, y si supera un cierto umbral, se considerará candidata a nota musical.

### 2.4. Determinación del tono y duración de las notas

La siguiente tarea consiste en asignar un valor que identifique el tono y la duración de la nota. Para asignar estos valores se ha elegido la nomenclatura propuesta en [9]. En esta referencia, se proponen diferentes notaciones, de las cuales se han adoptado las notaciones implícita (*splitted implicit notation*) y diferencial (*splitted diferential notation*). En la notación implícita se representa el valor de la escala de tonos mediante un valor numérico. La diferencia entre un tono y el siguiente se traduce en un incremento numérico de dos unidades, a excepción de los pasos de $Mi$ a $Fa$ y de $Si$ a $Do$, entre los que sólo existe un semitono de diferencia, lo cual implica un incremento de una unidad.

La notación diferencial se basa en la notación implícita. Consiste en calcular la diferencia entre el tono de una nota y de la siguiente. De este modo se consigue una abstracción de la tonalidad, preservando el patrón melódico. Para representar la duración de una nota, se codifica mediante una letra: $n$ para las negras, $c$ para las corcheas, $s$ para las semicorcheas, $f$ para las fusas, etc.

## 3. Resultados experimentales

Los algoritmos de tratamiento de partituras musicales se han implementado en Matlab, como un conjunto de *scripts*. En este apartado se presentan los

**Cuadro 1.** Evaluación del rendimiento del OMR desarrollado.

| | Número | Porcentaje |
|---|---|---|
| Total Notas | 1786 | 100 % |
| Falsos Positivos | 63 | 3.53 % |
| Falsos Negativos | 13 | 0.73 % |
| Errores en el Tono | 61 | 3.42 % |
| Errores en la Duración | 524 | 29.34 % |

resultados de identificación de las partituras analizadas. Estas partituras forman parte del repertorio de invenciones de J.S. Bach. Se trata de piezas polifónicas de música de contrapunto. En la tabla 1 se muestra el número de notas contenidas en cada compás y el número y tipo de los errores. Los errores considerados son: (i) falsos positivos (notas que no aparecen en la partitura y que el sistema ha detectado), (ii) falsos negativos (notas que aparecen en la partitura y que el sistema no ha detectado), (iii) errores cometidos en la determinación del tono de la nota y (iv) errores cometidos en la determinación de la duración de la nota.

Por lo tanto, se detectan correctamente más del 95 % de todas las notas contenidas en las partituras analizadas. Del total de notas halladas, menos del 4 % de las veces se comenten errores en el cálculo del tono de dichas notas. Los peores resultados se obtienen en la determinación de la duración de las notas, donde el porcentaje de error se eleva al 29 %.

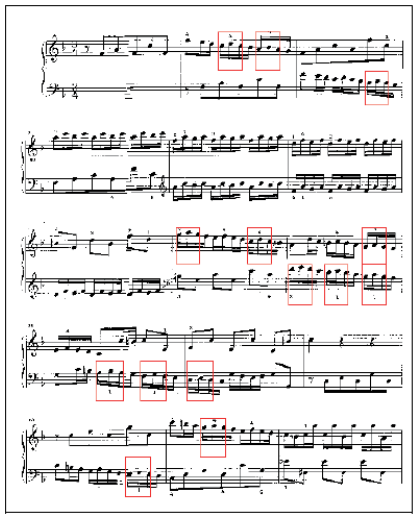


**Figura 3.** detección del patrón “2,-2,-2” en una partitura.

Como característica adicional, se ha desarrollado una herramienta de búsqueda de patrones en la partitura. Esta herramienta puede ser de utilidad para el análisis del estilo de una composición. En la Figura 3 se muestra una partitura y el resultado de la identificación del patrón "2,-2,-2" (en notación diferencial) en recuadros.

## 4. Conclusiones

En todo sistema OMR el índice de fiabilidad es muy importante. En este trabajo, nos hemos centrado en la determinación precisa del tono de las notas contenidas en una partitura de polifonía barroca. Como resultado, se ha desarrollado una aplicación que, pese a distar de las funcionalidades que ofrecen sistemas OMR comerciales, proporciona unos resultados aceptables (índice de fiabilidad del 95 %), si bien es cierto que se omite parte de la información contemplada por los sistemas comerciales como por ejemplo, silencios y alteraciones. En este sentido, un trabajo futuro que se plantea está orientado a describir fielmente estas otras características relevantes de una partitura, así como la mejora de la tasa de errores correspondientes a la duración de las notas. Otro trabajo planteado consiste en la adaptación de los algoritmos propuestos al análisis de música armónica.

## Referencias

1. Buzzanca, G.: A Rule-Based Expert System for Musical Style Recognition. Proceedings of the 1st Int. Conf. "Understanding and Creating Music" (2001)
2. W. Chai, B. Vercoe: Folk Music Classification using Hidden Markov Models. Proc. of the International Conference on Artificial Intelligence (2001)
3. Hori, T., Wada, S., Howzan T., Kung, S.Y.: Automatic music score recognition/play system based on decision based neural network. IEEE 3rd Workshop on Multimedia Signal Processing (1999) 183–184
4. Madsen, S. T., Widmer, G.: Evolutionary Search for Musical Parallelism. LNCS 3449 (2005) 488–497
5. Brown, A. R.: Exploring Rhythmic Automata. LNCS 3449 (2005) 488–497
6. Gartland-Jones, A., Copley, P.: The Suitability of Genetic Algorithms for Musical Composition. Contemporary Music Review, **22**:3 (2003) 43-55
7. Bainbridge, D.; Bell, T.C.: Dealing with superimposed objects in optical music recognition. Sixth International Conference on Image Processing and Its Applications, vol 2 (1997) 756–760
8. Otsu, N.: A Threshold Selection Method from Gray-Level Histograms, IEEE Transactions on Systems, Man, and Cybernetics, **9**:1 (1979) 62–66
9. Cruz-Alcázar, P.P., Vidal-Ruiz, E.: Modeling Musical Style using Grammatical Inference Techniques: a Tool for Classifying and Generating Melodies. Proceedings of the WEDELMUSIC'03 IEEE (2003) 77–84

# La visión artificial y las operaciones morfológicas en imágenes binarias

Jesús Cáceres Tello

Servicios Informáticos
Profesor Asociado del Dpto. Ciencias de la Computación
Escuela Técnica Superior de Informática, Universidad de Alcalá
jesus.caceres@uah.es

**Resumen.** Este artículo muestra una introducción a la visión artificial como una parte importante de la inteligencia artificial y la aplicación de esta al tratamiento de las imágenes, más concretamente a las imágenes con formato binario, su tratamiento morfológico será el tema a destacar.

## 1 Introducción

La visión artificial es una subrama de la inteligencia artificial orientada a permitir al computador a que pueda entender una imagen o conjunto de ella.

Todo esto se consigue median múltiples técnica:

- La detección, segmentación, localización y reconocimiento de ciertos objetos en imágenes (por ejemplo, caras humanas).
- La evaluación de los resultados (ej.: segmentación, registro).
- Registro de diferentes imágenes de una misma escena u objeto, i.e., hacer concordar un mismo objeto en diversas imágenes.
- Seguimiento de un objeto en una secuencia de imágenes.
- Mapeo de una escena para generar un modelo tridimensional de la escena; tal modelo podría ser usado por un robot para navegar por la escena.
- Estimación de las posturas tridimensionales de humanos.
- Búsqueda de imágenes digitales por su contenido.

## 2 Elementos de un sistema de visión artificial

La visión artificial es una técnica basada en la adquisición de imágenes, generalmente en dos dimensiones, para luego procesarlas digitalmente mediante algún tipo de CPU (computadora, microcontrolador, DSP, etc), con el fin de extraer y medir determinadas propiedades de la imágenes adquiridas. Se trata, por tanto, de una tecnología que combina las computadoras con las cámaras de video para adquirir, analizar e interpretar imágenes de una forma equivalente a la inspección visual humana.

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 570-575, 2006.

Un sistema de visión artificial se compone básicamente de los siguientes elementos:

1. Fuente de luz.
2. Sensor de Imagen.

## 2.1 Fuente de Luz

Es un aspecto de vital importancia ya que debe proporcionar unas condiciones de iluminación uniformes e independientes del entorno, facilitando además, si es posible, la extracción de los rasgos de interés para una determinada aplicación.

La fuente de luz es un factor de vital importancia en los sistemas de visión artificial y afectan de forma crucial a los algoritmos de visión que se vayan a utilizar bajo esas condiciones.

## 2.2 Sensor de Imagen

Es el encargado de recoger las características del objeto bajo estudio. Los sensores de imagen son componentes sensibles a la luz que modifican su señal eléctrica en función de la intensidad luminosa que perciben. La tecnología más habitual en este tipo de sensores es el CCD (charge coupled devices o dispositivos de acoplamiento de carga) donde se integra en un mismo chip los elementos fotosensibles y el conjunto de puertas lógicas y circuitería de control asociada. En éstos, la señal eléctrica que transmiten los fotodiodos es función de la intensidad luminosa que reciben, su espectro, y el tiempo de integración (tiempo durante el cual los fotodiodos son sensibles a la luz incidente).

**Resolución**

Existen diferentes arquitecturas de sensores. En primer lugar están los sensores lineales. En éstos, el sensor es una línea de fotodiodos. Esta arquitectura permite la utilización de sensores de 1x1024, 1x2048, 1x4096, e incluso 1x6000 píxeles, lo que la hace muy adecuada para trabajar con altas resoluciones sobre superficies en movimiento. Para condiciones de iluminación muy exigentes o velocidades de trabajo muy altas existe la posibilidad del uso de sensores TDI (time delay integrated). Esta tecnología consiste en el uso de varias líneas de captura sobre la misma línea del objeto, con el fin de sumar su carga y obtener así una mayor sensibilidad. En segundo lugar están los sensores de área. Estos alcanzan resoluciones habituales de 1024x1024, aunque existen en el mercado algunas casas que disponen de cámaras especiales con resoluciones de hasta 3072x2048. En este caso existen tecnologías de adquisición de imágenes, entrelazada y no entrelazada. El método entrelazado captura las líneas pares e impares que forman una imagen en instantes d e tiempo diferentes. La tecnología de no entrelazado (progressive scan) captura todas las líneas en el mismo instante de tiempo. Es más costoso económicamente, pero indispensable para trabajar con objetos en movimiento.

**Cuantización**
La cuantización (conversión analógica-digital) determina el número de bits usados para representar la información capturada. Por ejemplo, usando un sistema blanco y negro de 8 bits tenemos 256 niveles diferentes mientras que, usando un sistema de 10 bits, obtendríamos 1024 niveles de gris diferentes, lo que permite una mayor definición.

- Tarjeta de captura o adquisición de imágenes: es la interfaz entre el sensor y la computadora o módulo de proceso que permite al mismo disponer de la información capturada por el sensor de imagen. Las tarjetas de captura de imagen permiten transferir la imagen de la cámara a la memoria de la computadora con el fin de que ésta pueda realizar el procesamiento adecuado a las imágenes.
- Algoritmos de análisis de imagen: es la parte inteligente del sistema. Su misión consiste en aplicar las transformaciones necesarias y extracciones de información de las imágenes capturadas, con el fin de obtener los resultados para los que haya sido diseñado. Los algoritmos relacionados con visión artificial son muy variados y abarcan numerosas técnicas y objetivos.
- Computadora o módulo de proceso: es el sistema que analiza las imágenes recibidas por el sensor para extraer la información de interés en cada uno de los casos implementando y ejecutando los algoritmos diseñados para la obtención de los objetivos.
- Sistema de respuesta en tiempo real: con la información extraída, los sistemas de visión artificial pueden tomar decisiones que afecten al sistema productivo con el fin de mejorar la calidad global de producción. Se trata de automatismos que responden electromecánicamente con el fin de corregir o evitar, por ejemplo, en los sistemas de producción, las causas generadoras de los problemas de detección.

## 3 Operaciones morfológicas en imágenes binarias

La morfología matemática es una herramienta muy utilizada en el procesamiento de imágenes. Las operaciones morfológicas pueden simplificar los datos de una imagen, preservar las características esenciales y eliminar aspectos irrelevantes. Teniendo en cuenta que la identificación y descomposición de objetos, la extracción de rasgos, la localización de defectos e incluso los defectos en líneas de ensamblaje están sumamente relacionados con las formas, es obvio el papel de la morfología matemática.

La morfología matemática se puede usar, entre otros, con los siguientes objetivos:

- Preprocesamiento de imágenes (supresión de ruido, simplificación de formas).
- Destacar la estructura de objetos (extraer el esqueleto, marcado de objetos, envolvente convexa, ampliación, reducción).
- Descripción cualitativa de objetos (área, perímetro, diámetro, etc).

### 3.1 Representación de imágenes binarias

Definiremos una imagen binaria como una función de dos variables discretas a[m,n]que puede tomar dos valores, '0' o '1', dependiendo del nivel de gris de la imagen (una imagen binaria tiene dos niveles: blanco y negro). Se puede proponer una definición alternativa si consideramos que una imagen consiste en un conjunto de coordenadas discretas (también pueden ser reales pero no es el objetivo de este estudio). En este sentido, el conjunto corresponde a todos aquellos puntos o píxeles que pertenecen a la imagen. Por lo tanto, se puede decir que en morfología matemática los conjuntos representan objetos en una imagen. Por ejemplo, el conjunto de todos los píxeles negros en una imagen binaria constituye una descripción completa de la misma (Fig. 1).

Como ya habíamos adelantado, en las imágenes binarias, los conjuntos en cuestión pertenecen al espacio Z2, donde cada elemento del conjunto es una 2-upla (vector 2D) cuyas coordenadas son las coordenadas [m,n] de un píxel blanco (o negro, según la convención) de la imagen.

En la figura 2 se pueden ver dos conjuntos, A y B. Observemos que se ha colocado un sistema de coordenadas. El conjunto (u objeto) A consiste en los puntos { [2,3]; [2,4]; [2,5]; [1,3]; [1,4]; [1,5]; [0,5] } mientras que el B contiene los puntos { [0,0]; [0,1]; [1,0] }. En este punto se debe acotar que en la mayoría de los lenguajes de programación los arrays de elementos que serán los encargados de contener la imagen no admiten índices negativos y en general menores a 1. Por lo tanto, será necesario realizar una pequeña modificación al sistema de coordenadas que consiste en un simple desplazamiento para poder operar sobre una imagen (contenida en un array).

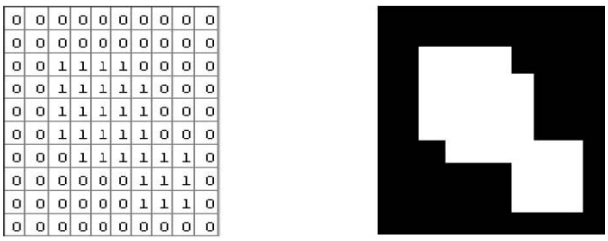



**Fig. 1**: Representación binaria de una imagen

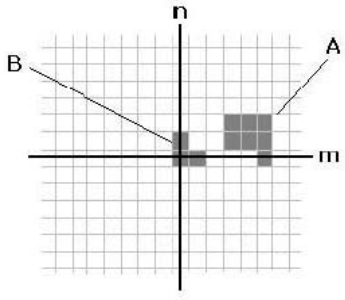



**Fig. 2**: Representación de una imagen binaria en el eje de coordenadas (Arrays de elementos)

### 3.2 Dilatación y erosión

Estas operaciones son fundamentales en el procesamiento morfológico. De hecho, la mayoría de los algoritmos morfológicos están bas ados en estas dos operacio nes. Otras operaciones importantes son las de Apertura y Clausura (Opening y Closing) debido a que hacen uso de la combinación de las operaciones de dilatación y erosión como veremos más adelantes.

Además existen otras operaciones que combinan de diferentes maneras las operaciones anteriormente citadas como son:

- Transformación Hit-or-Miss (o Ganancia - Pérdida)
- Extracción de Frontera (Boundary Extraction)
- Afinado o Adelgazamiento (Thinning)
- Engrosamiento (Thickening)
- Relleno de Región (Region Filling)
- Esqueleto
- Poda (Pruning)
- Traslación
- Reflexión
- Resta de imágenes

**Dilatación**
Tomar cada píxel del objeto (con valor “1”) y setear al valor “1” todos aquellos píxeles pertenecientes al fondo (background) que tienen una conectividad C (C=4, C=8, etc) con el píxel del objeto. En pocas palabras, poner a “1” los píxeles del fondo vecinos a los píxeles del objeto.

**Erosión**
Tomar cada píxel del objeto que tiene una conectividad C con los píxeles del fondo y resetearlo al valor “0”. En otras palabras, poner a “0” los píxeles del objeto vecinos a los píxeles del fondo.

**Apertura y Clausura (Opening and Closing)**
Como hemos visto, los conceptos de dilatación y erosión están muy relacionados; la primera operación expande la imagen mientras que la segunda la contrae. Generalmente estas dos operaciones se utilizan conjuntamente, bien la dilatación seguida de la erosión o viceversa. De cualquier manera, el resultado de la aplicación sucesiva de erosiones y dilataciones elimina detalles menores sin distorsionar la forma global del objeto.

La operación de Apertura se define como la aplicación de la erosión seguida de la operación de dilatación. Esta operación suaviza los contornos de un objeto y elimina las protuberancias finas.

Por otro lado, la operación de Clausura o Closing es la aplicación de la dilatación seguida por la aplicación de la operación de erosión sobre una imagen. Esta operación, al igual que la operación anterior, suaviza los contornos además de eliminar agujeros pequeños y rellenar brechas en el contorno.

## 6. Conclusiones

La visión artificial (machine vision) es la adquisición automática de imágenes sin contacto y su anális is también automático con el fin de extraer la información necesaria para controlar un proceso o una actividad.Las aplicaciones en las que interviene la visión artificial son cada vez más numerosas, entre loas que podemos destacar las siguientes:

- Control de calidad
- Ordenación por calidades (grading)
- Manipulación de materiales
- Test y calibración de aparatos
- Monitorización de procesos

## Referencias

David Navarro.¿Visión de lo abstracto desde lo simbólico?
http://www.redcientifica.com/doc/doc199903310012.html
Darío Maravall. Reconocimiento de Formas y Visión Artificial. Editorial RA-MA.
Rafael C. González. Digital Image Procesing. Editorial Adddison-Wesley.
Jain. Fundamentals of Digital Image Processing. Editorial Prentice-Hall.

# Una comparativa entre el álgebra de rectángulos y la lógica SpPNL

Antonio Morales y Guido Sciavicco

Departamento de Ingeniería de la Información y las Comunicaciones
Universidad de Murcia
Murcia 30100 - Campus de Espinardo, España
{morales,guido}@um.es

**Resumen** En Inteligencia Artificial está ampliamente aceptado el importante papel que desempeña el razonamiento espacial. Por un lado, el razonamiento cualitativo espacial puede tratarse mediante teorías puramente existenciales en forma de sistemas de satisfacción de restricciones sobre un conjunto de relaciones mutuamente excluyentes y disjuntas. Por otro lado, las lógicas modales y de primer orden proporcionan un poder expresivo mayor a costa de un peor comportamiento computacional. En este artículo, se compara una lógica modal, llamada *Spatial Propositional Neighborhood Logic* (SpPNL), con el Álgebra de Rectángulos, que es la extensión bidimensional del Álgebra de Intervalos de Allen para el razonamiento temporal. Se demuestra como es posible comprobar la consistencia de una red de restricciones del Álgebra de Rectángulos mediante una fórmula en SpPNL. También se muestra como dicha lógica permite expresar ciertas restricciones espaciales intuitivas que no pueden representarse en el Álgebra de Rectángulos.

## 1. Introducción

Las teorías y técnicas de razonamiento espacial desempeñan un importante papel en las distintas aplicaciones de la inteligencia artificial. Del mismo modo que ocurre con otras formas de razonamiento cualitativo (p.e., en razonamiento cualitativo temporal), el razonamiento espacial puede verse bajo distintos (y de algún modo complementarios) puntos de vista. Por un lado, podemos verlo desde un punto de vista *algebraico*, es decir, como teorías puramente existenciales formuladas como sistemas de satisfacción de restricciones sobre un conjunto de relaciones topológicas, direccionales o combinadas, mutuamente excluyentes y disjuntas. Por otro lado, está el punto de vista de la lógica, donde podemos distinguir entre teorías de *primer orden*, también sobre relaciones topológicas, direccionales o combinaciones de ambas; y *lógicas modales* (normalmente proposicionales), donde un lenguaje modal se interpreta sobre estructuras de Kripke que representan el espacio. Además, el razonamiento espacial puede clasificarse en topológico, direccional o combinado, dependiendo del tipo de relaciones utilizadas para razonar sobre los objetos espaciales.

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 576-587, 2006.

Centrándonos en el nivel algebraico, la teoría conocida como *Region Connection Calculus* (RCC), presentada por Randell, Cui y Cohn [16], es probablemente la aproximación topológica más importante al razonamiento cualitativo espacial. Una especialización importante de RCC es RCC8 [7, 16], que incluye un conjunto de 8 relaciones espaciales, mutuamente excluyentes y disjuntas dos a dos. En general, el cálculo con relaciones topológicas se basa en un espacio topológico, es decir, un par $\mathcal{T} = \langle U, I \rangle$, donde $U$ es un conjunto y $I$ es el operador *interior*, que cumple ciertas propiedades de cierre. Dados dos conjuntos cualesquiera $O_1, O_2 \subseteq U$, cabe preguntarse por la relación binaria que puede darse entre ambos. Hay muchas posibilidades para definir este conjunto de relaciones. Por ejemplo, RCC8 está formado por las relaciones *igual*, *desconectado*, *conectado por fuera*, *coincide parcialmente*, *parte interior tangencial*, *parte interior no-tangencial* y las inversas de las dos últimas[1].

El trabajo más influyente sobre la aproximación algebraica al razonamiento espacial con direcciones es probablemente la extensión espacial del Álgebra de Intervalos de Allen, realizada por Güsgen [9] y por Mukerjee y Joe [13]. El Álgebra de Rectángulos (AR), ha sido estudiado posteriormente por Balbiani, Condotta y Del Cerro [2, 3]. En el AR los objetos tratados son rectángulos con lados paralelos a los ejes de una base ortogonal en un espacio Euclídeo bidimensional. Estos rectángulos pueden considerarse como los objetos reales de interés o como una forma de aproximar regiones[2]. Una región se define, según la mayoría de los autores, como un conjunto puntos limitado en el espacio, cerrado, conectado y con límite conectado, o como una unión finita de tales conjuntos. Otra aproximación, diferente, al razonamiento espacial con relaciones direccionales es la propuesta por Skiadopulos y Koubarakis [19, 20].

Por último, debemos mencionar el trabajo de Sharma [18], donde se explora la posibilidad de integrar restricciones topológicas y direccionales, elaborando un conjunto de reglas de composición para restricciones mixtas. Otro trabajo, realizado por Sun y Li, en el que se combinan varios tipos de relaciones espaciales se puede ver en [21].

El enfoque algebraico consiste generalmente en definir objetos y sus relaciones básicas, y en razonar, dado un conjunto finito de objetos $O_1, \ldots, O_k$, comprobando la consistencia de un conjunto de restricciones (que forman una red) entre ellos. Un sistema lógico presenta un poder expresivo mayor. Permite, por ejemplo, representar información negativa o disyunciones lógicas de restricciones entre objetos diferentes, aunque con un comportamiento computacional peor. Sin embargo, se ha realizado mucho esfuerzo en explotar el poder expresivo de los sistemas lógicos para comprobar la consistencia de una red de restricciones algebraicas dada. El enfoque lógico de este problema consiste básicamente en definir un lenguaje lógico $L$ (proposicional, modal o de primer orden) tal

[1] *Equal* (*eq*) , *disconnected* (*di*), *externally connected* (*ec*), *tangential proper part* (*tpp*), *inverse of tangential proper part* (*tppi*), *non-tangential proper part* (*ntpp*), *inverse of non-tangential proper part* (*ntppi*), y *partially overlap* (*po*).

[2] Por ejemplo, el rectángulo mínimo que contiene a una región, o *Minimal Bounding Rectangle* o *Minimal Bounding Box*.

que, dada una red $N$ exista una fórmula $\phi$ de $L$ tal que $\phi$ es satisfacible si y sólo si $N$ es consistente. Concretamente, sobre el uso de la lógica modal para razonamiento espacial con este fin, es importante mencionar el trabajo de Bennett [4, 5], ampliado posteriormente por Bennett, Cohn, Wolter y Zakharyaschev en [6]. En [4], Bennett propone interpretar las regiones como subconjuntos del espacio topológico, y demuestra como es posible explotar el cálculo proposicional y el cálculo proposicional constructivo, junto con algunos vínculos de *meta-nivel* concernientes a la deducción entre fórmulas, para razonar con relaciones topológicas. De ese modo, un problema con restricciones espaciales topológicas se puede solucionar comprobando la satisfacibilidad de una fórmula lógica. En [5], Bennett extiende con lenguajes modales esta forma de abordar el razonamiento espacial. Bennett considera la lógica modal S4, e interpreta el operador modal como el operador topológico *interior*. Además, en el mismo trabajo, se define y se estudia un operador modal de *envoltura convexa*, mediante la traducción de una axiomatización de primer orden en un esquema modal. En [6], los autores consideran un sistema multi-modal para razonamiento espacio-temporal basado en el trabajo previo de Bennett. Otro trabajo relacionado con el tema se puede encontrar en [14], donde Nutt da los fundamentos formales para la traducción de relaciones topológicas en lógica modal, introduciendo relaciones topológicas generalizadas. Cabe señalar que los resultados de Bennett, Cohn, Wolter, Zakharyaschev y Nutt aprovechan las propiedades de modelo finito y de decidibilidad de la lógica proposicional clásica, la lógica modal S4 y de alguna de sus extensiones. En [1] puede verse una reciente investigación sobre las teorías matemáticas más importantes en el dominio espacial, desde un punto de vista modal.

En este artículo consideramos el mismo problema pero con respecto a relaciones direccionales; problema que, por lo que sabemos, no ha recibido prácticamente ninguna atención hasta el momento. Esta situación puede deberse a la ausencia de lógicas modales decidibles para razonamiento espacial con relaciones direccionales, y a la falta interpretaciones sencillas de lógicas modales conocidas en términos de relaciones direccionales. En [12] se presenta una lógica modal para razonamiento espacial llamada *Spatial Propositional Neighborhood Logic* (SpPNL), donde se demuestra que el problema de la validez es semidecidible. Además, en [12] se muestra como es posible aplicar las técnicas de razonamiento temporal al razonamiento espacial en SpPNL. A pesar de la debilidad del resultado, esta lógica presenta un mejor comportamiento que otras lógicas modales espaciales como las presentadas en [10], que son, en general, altamente indecidibles. SpPNL utiliza cuatro modalidades, llamadas $\langle \mathrm{E} \rangle, \langle \mathrm{W} \rangle, \langle \mathrm{N} \rangle$ y $\langle \mathrm{S} \rangle$ que permiten movernos entre rectángulos no vacíos en un espacio bidimensional $\mathbb{D}^2 = \mathbb{D} \times \mathbb{D}$, donde $\mathbb{D} = \langle D, < \rangle$ es un conjunto linealmente ordenado. Así, por ejemplo, el hecho de que la letra proposicional $p$ se satisfaga en un objeto determinado (rectángulo), y que en otro objeto posicionado al este del primero (compartiendo un lado con él) se satisfaga la letra proposicional $q$, puede expresarse en SpPNL mediante la fórmula $p \wedge \langle \mathrm{E} \rangle \mathrm{q}$. Demostraremos que en SpPNL es posible expresar el conjunto completo de relaciones del AR (básicas y no básí-

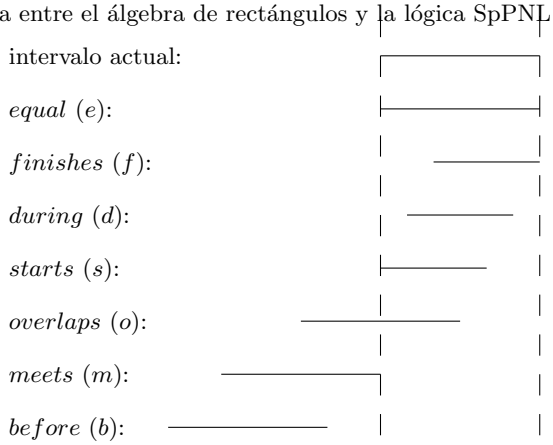



**Figura 1.** Relaciones de Allen entre intervalos.

cas), permitiéndonos comprobar por lo menos la no consistencia de una red en el AR (un resultado similar, en el contexto temporal, se puede encontrar en [17]). Es importante destacar que en una lógica como SpPNL es posible expresar un conjunto importante de sentencias espaciales que no podrían ser expresadas en el AR.

El resto del artículo se organiza de la siguiente manera. En la siguiente sección recordamos los conceptos básicos del Álgebra de Rectángulos y en la Sección 3 presentamos SpPNL. En la Sección 4 mostramos como SpPNL tiene el suficiente poder expresivo para representar cualquier red del AR. Antes de concluir, en la Sección 5, damos algunos ejemplos de como escribir en SpPNL algunas expresiones del lenguaje natural.

## 2. El álgebra de rectángulos

En el Álgebra de Rectángulos [2, 3, 13] los objetos tratados son rectángulos con lados paralelos a los ejes de una base ortogonal en un espacio Euclídeo bidimensional $\mathbb{D}^2 = \mathbb{D} \times \mathbb{D}$, donde $\mathbb{D} = \langle D, < \rangle$ es un conjunto linealmente ordenado. Una forma habitual de representar un rectángulo consiste en utilizar las coordenadas de dos de sus esquinas opuestas. Así, un rectángulo $O$ se puede definir como el par $\langle (d_x, d_y), (d'_x, d'_y) \rangle$ tal que $d_x < d'_x, d_y < d'_y$ donde $d_x, d'_x,\ d_y, d'_y \in D$. Una *relación básica* entre dos rectángulos $O_1$ y $O_2$ es un par $R = (r_i, r_j)$, donde $r_i$ y $r_j$ (llamadas *componentes*) son relaciones de Allen entre intervalos (las relaciones de Allen se muestran en la Fig. 1). Como norma, denotaremos por $r^{-1}$ a la relación inversa de una relación básica $r$ de Allen.

Así, por ejemplo, si el rectángulo $O_1$ está incluido completamente en el rectángulo $O_2$, y ningún lado de $O_1$ toca ningún lado de $O_2$, entonces la relación entre $O_1$ y $O_2$ es $(d, d)$, donde $d$ es la relación de Allen *during*. De esta forma, hay $13^2 = 169$ relaciones básicas posibles entre dos rectángulos cualesquiera, como se puede ver en la Fig. 2. En el AR se definen las operaciones

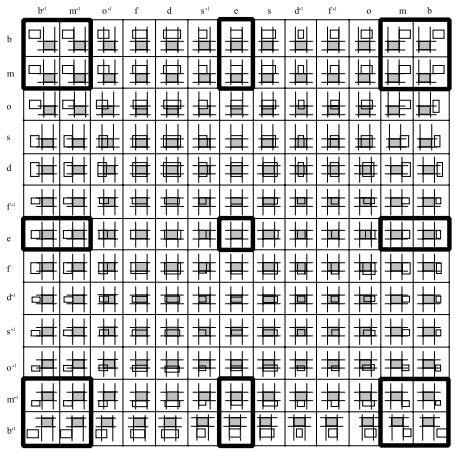


**Figura 2.** Las 169 relaciones básicas entre dos rectángulos. Los cuadros resaltados contienen las 25 relaciones básicas que pueden expresarse directamente en SpPNL.

básicas de *inversión*, *composición*, e *intersección*. Una AR-*relación*, que escribiremos como $\hat{R}$, es un conjunto $\{R_1, R_2, \ldots, R_n\}$, donde, para cada $i$, $R_i$ es una relación básica del AR. Se denota por $\mathcal{B}$ al conjunto de relaciones básicas del AR; así pues, toda relación del AR pertenece al conjunto $2^{\mathcal{B}}$, lo cual significa que hay $2^{169}$ relaciones posibles.

**Definition 1.** *Sea* $\mathbb{O}(\mathbb{D})$ *el conjunto de todos los rectángulos definidos en un espacio Euclídeo* $\mathbb{D}^2$*. Llamaremos* AR-restricción *a la expresión* $(O_i\ \hat{R}\ O_j)$*, donde* $O_i, O_j \in \mathbb{O}(\mathbb{D})$ *y* $\hat{R}$ *es una* AR-*relación. Una* red-AR *es un par* $\langle \mathcal{O}, \mathcal{R} \rangle$ *donde* $\mathcal{O} \subseteq \mathbb{O}(\mathbb{D})$ *y* $\mathcal{R}$ *es un conjunto de* AR-restricciones *entre elementos de* $\mathcal{O}$*.*

Dada una *red*-AR $N$, el principal problema es saber si es consistente o no. Una red-AR es consistente si y sólo si la información espacial representada por $N$ es coherente.

**Definition 2.** *Una* red-AR $\langle \mathcal{O}, \mathcal{R} \rangle$ *es* consistente *si y sólo si existe una instancia concreta de todo el conjunto* $\mathcal{O}$ *que respete todas las restricciones en* $\mathcal{R}$*.*

Por lo general, se sabe que la comprobación de la consistencia de una red-AR es un problema NP-completo; en [2, 3] se estudian fragmentos tratables del AR.

## 3. Sintaxis y semántica de SpPNL

El lenguaje de SpPNL está formado por un conjunto de variables proposicionales $\mathcal{AP}$, las conectivas lógicas $\neg$ y $\vee$, y las modalidades $\langle \mathrm{E} \rangle, \langle \mathrm{W} \rangle, \langle \mathrm{N} \rangle$ y $\langle \mathrm{S} \rangle$. El resto de conectivas lógicas, así como las constantes lógicas $\top$ y $\bot$, se definen de la forma habitual. Las *fórmulas* bien formadas de SpPNL, denotadas por $\phi, \psi, \ldots$, se definen recursivamente de la siguiente manera (donde $p \in \mathcal{AP}$):

$$\phi = p \mid \neg\phi \mid \phi \vee \psi \mid \langle \mathrm{E}\rangle\phi \mid \langle \mathrm{W}\rangle\phi \mid \langle \mathrm{N}\rangle\phi \mid \langle \mathrm{S}\rangle\phi.$$

Dado un conjunto linealmente ordenado $\mathbb{D} = \langle D, <\rangle$, llamaremos *marco espacial* al par $\mathbb{D}^2 = \mathbb{D} \times \mathbb{D}$, y denotaremos por $\mathbb{O}(\mathbb{D})$ al conjunto de todos los *objetos*, es decir, $\mathbb{O}(\mathbb{D}) = \{\langle (d_x, d_y), (d'_x, d'_y)\rangle \mid d_x < d'_x,\ d_y < d'_y,\ d_x, d'_x, d_y, d'_y \in D\}$. La semántica de SpPNL viene dada en términos de *modelos* del tipo $M = \langle \mathbb{D}^2, \mathbb{O}(\mathbb{D}), V\rangle$, donde $\mathbb{D}^2$ es un marco espacial, y $V : \mathbb{O}(\mathbb{D}) \mapsto 2^{\mathcal{AP}}$ es una *asignación espacial*. La relación de *verdad* para una fórmula bien formada $\phi$ en un modelo $M$ y un objeto $\langle (d_x, d_y), (d'_x, d'_y)\rangle$ es estándar para las letras proposicionales y los operadores clásicos, y para los operadores modales viene dada por las siguientes cláusulas:

- $M, \langle (d_x, d_y), (d'_x, d'_y)\rangle \Vdash \langle \mathrm{E}\rangle\phi$ si y sólo si existe $d''_x \in \mathbb{D}$ tal que $d'_x < d''_x$, y $M, \langle (d'_x, d_y), (d''_x, d'_y)\rangle \Vdash \phi$;
- $M, \langle (d_x, d_y), (d'_x, d'_y)\rangle \Vdash \langle \mathrm{W}\rangle\phi$ si y sólo si existe $d''_x \in \mathbb{D}$ tal que $d''_x < d_x$, y $M, \langle (d''_x, d_y), (d_x, d'_y)\rangle \Vdash \phi$;
- $M, \langle (d_x, d_y), (d'_x, d'_y)\rangle \Vdash \langle \mathrm{N}\rangle\phi$ si y sólo si existe $d''_y \in \mathbb{D}$ tal que $d'_y < d''_y$, y $M, \langle (d_x, d'_y), (d'_x, d''_y)\rangle \Vdash \phi$;
- $M, \langle (d_x, d_y), (d'_x, d'_y)\rangle \Vdash \langle \mathrm{S}\rangle\phi$ si y sólo si existe $d''_y \in \mathbb{D}$ tal que $d''_y < d_y$, y $M, \langle (d_x, d''_y), (d'_x, d_y)\rangle \Vdash \phi$.

Como es habitual, denotaremos con [E] (resp., [W], [N], [S]) al operador dual de $\langle \mathrm{E}\rangle$ (resp., $\langle \mathrm{W}\rangle$, $\langle \mathrm{N}\rangle$, $\langle \mathrm{S}\rangle$), y con $M \Vdash \phi$ para expresar que $\phi$ es una fórmula *válida* en $M$.

Es fácil ver que sólo 25 de las 169 AR-relaciones básicas posibles se pueden expresar directamente en SpPNL (ver Fig. 2); pero, como veremos en la próxima sección, bajo ciertas condiciones, es posible expresarlas *todas* en SpPNL mediante su correspondiente fórmula.

**Theorem 1.** *El problema de la validez para* SpPNL *interpretado en la clase de todos los espacios Euclídeos* $\mathbb{D}^2$*, donde* $\mathbb{D}$ *es un conjunto linealmente ordenado, es recursivamente enumerable.*

La demostración del teorema anterior (dada en [12]) se basa en una traducción polinómica no trivial entre las fórmulas de SpPNL y las fórmulas de la lógica PNL ([8, 11]), que es una lógica temporal decidible basada en intervalos, que preserva la validez.

## 4. Cómo comprobar la consistencia de una red-AR en SpPNL

En esta sección mostraremos como es posible comprobar la consistencia de una red-AR $N$ comprobando la satisfacibilidad de la correspondiente fórmula $\phi(N)$ en SpPNL. Para esto, primero mostraremos como es posible expresar una relación del AR (básica o no básica) en SpPNL.

Considérese la siguiente expresión:

$$hor(\phi) = [\mathrm{W}][\mathrm{W}][\mathrm{E}]\phi \wedge [\mathrm{W}][\mathrm{E}][\mathrm{E}]\phi \wedge [\mathrm{E}][\mathrm{E}][\mathrm{W}]\phi \wedge [\mathrm{E}][\mathrm{W}][\mathrm{W}]\phi$$

El operador $hor(\phi)$ establece que la fórmula $\phi$ se satisface en todo rectángulo cuya coordenada $y$ sea la misma que la del rectángulo actual. El operador $ver(\phi)$ se puede definir de forma parecida. Esto quiere decir que en SpPNL es posible expresar el operador *diferencia*:

$$[\neq](\phi) = hor(\phi \wedge ver(\phi)),$$

y por lo tanto, simular la *modalidad universal* y los *nominales*:

$$u(\phi) = \phi \wedge [\neq]\phi \quad \text{y} \quad n(p) = p \wedge [\neq](\neg p),$$

donde $n(p)$ significa que $p$ no se satisface en ningún otro rectángulo que no sea el actual. A continuación, utilizaremos la simulación de nominales para traducir AR-relaciones básicas en fórmulas de SpPNL. En general, considérese una restricción-AR $(O_i\ \hat{R}\ O_j)$, donde $O_i$, $O_j$ son objetos, y $\hat{R}$ es una AR-relación. Supóngase que el conjunto de variables proposicionales $\mathcal{AP}'$ contiene dos letras proposicionales $p(O_i)$ y $p(O_j)$. Ahora, queremos escribir una fórmula $\phi_{\hat{R}}(O_i, O_j)$ tal que, para cualquier modelo espacial $M = \langle \mathbb{D}^2, \mathbb{O}(\mathbb{D}), V\rangle$, tenemos que $M$, $\langle (d_x, d_y),(d'_x, d'_y)\rangle \Vdash \phi_{\hat{R}}(O_i, O_j)$ si y sólo si, dándose el caso en el que $p(O_i) \in V(\langle (d_x, d_y), (d'_x, d'_y)\rangle)$, entonces $p(O_j) \in V(\langle (d''_x, d''_y), (d'''_x, d'''_y)\rangle)$ para algún rectángulo $\langle (d''_x, d''_y), (d'''_x, d'''_y)\rangle$ tal que $\langle (d_x, d_y), (d'_x, d'_y)\rangle\ \hat{R}\ \langle (d''_x, d''_y), (d'''_x, d'''_y)\rangle$.

**Definition 3.** *Dada una restricción-*AR $(O_i\ \hat{R}\ O_j)$*, decimos que la fórmula* $\phi_{\hat{R}}(O_i, O_j)$*, expresada en* SpPNL*,* respeta $(O_i\ \hat{R}\ O_j)$ *si y sólo si* $\phi_{\hat{R}}(O_i, O_j)$ *satisface la propiedad anterior.*

Para empezar consideraremos primero las relaciones básicas. Serán necesarias 169 fórmulas distintas para traducir todas las AR-relaciones básicas. Podemos dividir dichas relaciones en tres grupos como se indica a continuación.

**Relaciones directas.** Son las 25 relaciones básicas que se pueden expresar directamente en SpPNL. Por ejemplo, la relación básica $(O_1\ (e, e)\ O_2)$ puede expresarse mediante la fórmula $p(O_1) \rightarrow p(O_2)$. Otro ejemplo sería la relación básica $(O_1\ (b, b)\ O_2)$, que puede expresarse con la fórmula $p(O_1) \rightarrow \langle \mathrm{E}\rangle\langle \mathrm{E}\rangle\langle \mathrm{N}\rangle\langle \mathrm{N}\rangle p(O_2)$.

**Lemma 1.** *Para toda* AR*-relación básica* $(O_i\ R\ O_j)$*, existe una fórmula en* SpPNL $\phi_R(O_i, O_j)$ *que respeta* $(O_i\ R\ O_j)$.

**Relaciones parcialmente indirectas.** Una AR-relación parcialmente indirecta $R = (r_i, r_j)$ es una AR-relación básica en la que exactamente una de sus componentes puede expresarse directamente en SpPNL. Centrándonos en un sólo eje, hay 5 relaciones de Allen que se pueden expresar en SpPNL, que son $\{e, m, b, m^{-1}, b^{-1}\}$. Esto quiere decir que hay 80 AR-relaciones parcialmente indirectas: si $R = (r_i, r_j)$ es una relación parcialmente indirecta, entonces $r_i$ puede

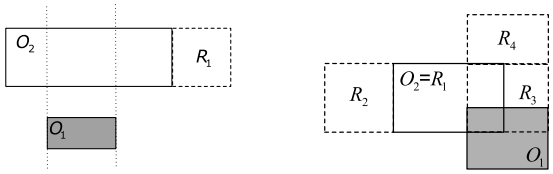



**Figura 3.** Ejemplos de relaciones entre rectángulos. Relación $O_1$ $(b, d)$ $O_2$ (a), y relación $O_1$ $(o, o^{-1})$ $O_2$ (b).

tomarse de un conjunto de 5 relaciones de Allen, y $r_j$ del conjunto que queda de 8 relaciones de Allen, o al contrario; por ejemplo, la relación $(b, d)$ es parcialmente indirecta. Resulta que, si $(O_i\ R\ O_j)$ es parcialmente indirecta, haciendo uso de como mucho dos (simulaciones de) nominales es posible escribir una fórmula $\phi_R(O_i, O_j)$ en SpPNL que respeta $R$.

Considere por ejemplo la relación $O_1$ $(b, d)$ $O_2$ representada en la Fig. 3 (a). La variable proposicional denotada por $R_1$ representa un nominal que se puede utilizar para expresar la relación $(b, d)$. Considérese la fórmula $\phi_{(b,d)}(O_1, O_2) = p(O_2) \rightarrow \langle \mathrm{E} \rangle\, n(p_{R_1}) \wedge \langle \mathrm{W} \rangle\, \langle \mathrm{E} \rangle \langle \mathrm{E} \rangle\, (\langle \mathrm{E} \rangle \langle \mathrm{E} \rangle\, n(p_{R_1}) \wedge \langle \mathrm{S} \rangle \langle \mathrm{S} \rangle\, (p(O_1)))$, donde $p(O_1)$, $p(O_2)$ son variables proposicionales que representan objetos, y $p_{R_1}$ es una variable proposicional utilizada para simular un nominal.

**Proposition 1.** *La fórmula* $\phi_{(b,d)}(O_1, O_2)$ *respeta la* AR*-relación* $(b, d)$*.*

*Demostración.* Supóngase que existe un modelo espacial $M$ tal que $M, \langle (d_x, d_y), (d'_x, d'_y) \rangle \Vdash \phi_{(b,d)}(O_1, O_2)$. Esto quiere decir que $M, \langle (d_x, d_y), (d'_x, d'_y) \rangle \Vdash p(O_2) \rightarrow \langle \mathrm{E} \rangle\, n(p_{R_1}) \wedge \langle \mathrm{W} \rangle \langle \mathrm{E} \rangle\, \langle \mathrm{E} \rangle\, (\langle \mathrm{E} \rangle \langle \mathrm{E} \rangle\, n(p_{R_1}) \wedge \langle \mathrm{S} \rangle \langle \mathrm{S} \rangle\, (p(O_1)))$. Suponga ahora que $p(O_2) \in V(\langle (d_x, d_y), (d'_x, d'_y) \rangle)$. En este caso, tenemos que en algún objeto $\langle (d'_x, d_y), (\hat{d_x}, d'_y) \rangle$ tal que $d'_x < \hat{d_x}$ se satisface $p_{R_1}$, y no se satisface en ninguna otra parte. Así pues, como en el objeto $\langle (d_x, d_y), (d'_x, d'_y) \rangle$ se satisface $\langle \mathrm{W} \rangle \langle \mathrm{E} \rangle\, \langle \mathrm{E} \rangle\, (\langle \mathrm{E} \rangle \langle \mathrm{E} \rangle\, n(p_{R_1}) \wedge \langle \mathrm{S} \rangle \langle \mathrm{S} \rangle\, (p(O_1)))$, la única forma posible de poner $p(O_1)$ es justo sobre un objeto $\langle (d''_x, d''_y), (d'''_x, d'''_y) \rangle$ tal que $d_x < d''_x$, $d'''_x < d'_x$, y $d'_y < d_y$. Por lo tanto, $\langle (d''_x, d''_y), (d'''_x, d'''_y) \rangle$ cumple la relación $(b, d)$ con respecto a $\langle (d_x, d_y), (d'_x, d'_y) \rangle$. □

El siguiente lema requeriría una demostración similar para cada uno los 80 casos distintos.

**Lemma 2.** *Para cada* AR*-relación parcialmente indirecta* $(O_i\ R\ O_j)$*, existe una fórmula* $\phi_R(O_i, O_j)$ *en* SpPNL *que respeta* $R$*.*

**Relaciones Indirectas.** Una AR-relación indirecta $R = (r_i, r_j)$ es una AR-relación básica tal que ninguna de sus componentes puede ser expresada directamente en SpPNL. Como hemos visto antes, 8 de las 13 relaciones de Allen (implicadas en las proyecciones de dos objetos sobre el mismo eje) no pueden ser expresadas directamente en SpPNL. Dichas relaciones pertenecen al conjunto

$I - \{e, m, b, m^{-1}, b^{-1}\}$, donde $I$ es el conjunto de todas las relaciones de Allen. Esto quiere decir que hay 64 AR-relaciones indirectas: si $R = (r_i, r_j)$ es una relación indirecta, entonces $r_i$ y $r_j$ pueden tomarse de un conjunto de 8 relaciones de Allen; por ejemplo, la relación $(o, o^{-1})$ es indirecta. Resulta que, si $(O_i\ R\ O_j)$ es indirecta, haciendo uso de como mucho cuatro (simulaciones de) nominales es posible escribir una fórmula $\phi_R(O_i, O_j)$ en SpPNL que respeta $(O_i\ R\ O_j)$.

Considérese por ejemplo la relación $O_1\ (o, o^{-1})\ O_2$, representada gráficamente en la Fig. 3 (b). Las variables proposicionales $R_1, R_2, R_3$ y $R_4$ representan letras proposicionales que pueden utilizarse como simulaciones de nominales para conseguir expresar la relación $(o, o^{-1})$. Considérese la fórmula $\phi_{(o,o^{-1})}(O_1, O_2) = p(O_2) \rightarrow n(p_{R_1}) \wedge \langle \mathrm{W} \rangle n(p_{R_2}) \wedge \langle \mathrm{E} \rangle\ \langle \mathrm{E} \rangle\ \langle \mathrm{W} \rangle ([\mathrm{W}] \neg p_{R_1} \wedge [W]\ [W] \neg p_{R_1} \wedge \langle \mathrm{W} \rangle\ \langle \mathrm{W} \rangle\ p_{R_2} \wedge (n(p_{R_3}) \wedge \langle \mathrm{N} \rangle n(p_{R_4}) \wedge \langle \mathrm{S} \rangle\ \langle \mathrm{S} \rangle\ \langle \mathrm{N} \rangle\ (p(O_i) \wedge [N]\ \neg p_{R_3} \wedge [N][N] \neg p_{R_3} \wedge \langle \mathrm{N} \rangle\ \langle \mathrm{N} \rangle\ p_{R_4})))$.

**Proposition 2.** *La fórmula* $\phi_{(o,o^{-1})}(O_i, O_j)$ *respeta la* AR*-relación* $(o, o^{-1})$.

*Demostración.* Supóngase que existe un modelo espacial $M$ tal que $M, \langle (d_x, d_y), (d'_x, d'_y) \rangle \Vdash \phi_{(o,o^{-1})}(O_i, O_j)$, y $M, \langle (d_x, d_y), (d'_x, d'_y) \rangle \Vdash p(O_2)$. Esto quiere decir que el mismo rectángulo $\langle (d_x, d_y), (d'_x, d'_y) \rangle$ hace verdadero a $p_{R_1}$, y que $p_{R_1}$ es falso en cualquier otro lugar. Además, por $\langle \mathrm{W} \rangle n(p_{R_2})$, tenemos que en algún rectángulo $\langle (d''_x, d_y), (d_x, d'_y) \rangle$ con $d''_x < d_x$ la letra proposicional $p_{R_2}$ es verdadera, y no lo es en ninguna otra parte. Ahora, la única forma de poner $p_{R_3}$ es en algún rectángulo $\langle (\overline{d_x}, \overline{d_y}), (\overline{d'_x}, \overline{d'_y}) \rangle$ tal que su proyección en el eje $x$, $[\overline{d_x}, \overline{d'_x}]$, se superpone al intervalo $[d_x, d'_x]$. De esta forma, $p_{R_3}$ es verdadera en $\langle (\overline{d_x}, \overline{d_y}), (\overline{d'_x}, \overline{d'_y}) \rangle$ pero no en ninguna otra parte, y $p_{R_4}$ es verdadera en algún rectángulo $\langle (\overline{d_x}, d'_y), (\overline{d'_x}, d''_y) \rangle$, con $d'_y < d''_y$, y en ningún otro lugar. Usando la misma estrategia que antes, $p(O_1)$ debe ponerse en algún rectángulo $\langle (\overline{d_x}, \hat{d}_y), (\overline{d'_x}, \hat{d}'_y) \rangle$ tal que su proyección en el eje $y$ $[\hat{d}_y, \hat{d}'_y]$ tenga la relación *overlaps* con $[d_y, d'_y]$. □

De nuevo, examinando los diferentes 64 casos es posible probar el siguiente lema.

**Lemma 3.** *Dada una* AR*-relación básica indirecta* $(O_i\ R\ O_j)$*, existe una fórmula* $\phi_R(O_i, O_j)$ *en* SpPNL *que respeta* $R$.

Ahora consideremos una AR-restricción genérica del tipo $(O_i\ \hat{R}\ O_j)$, donde $\hat{R} = \{R_1, \ldots, R_n\}$. Dicha restricción se interpreta como una disyunción lógica: $(O_i\ \hat{R}\ O_j)$ se cumple si y sólo si se cumple $\bigvee_{i=1}^{n}((O_i\ R_i\ O_j))$. Por lo tanto, obtenemos los siguientes resultados.

**Lemma 4.** *Dada una* AR*-restricción del tipo* $(O_i\ \hat{R}\ O_j)$*, donde* $\hat{R} = \{R_1, \ldots, R_n\}$*, la fórmula* $\phi_{\hat{R}}(O_i, O_j) = \bigvee_{i=1}^{n}(\phi_{R_i}(O_i, O_j))$ *en* SpPNL *respeta* $(O_i\ \hat{R}\ O_j)$.

Por último, nos queda por demostrar el resultado principal, es decir, que para toda red-AR existe una fórmula en SpPNL que es satisfacible si y sólo si $N$ es consistente. Considérese una red-AR cualquiera $N$, y sea $\{O_1, \ldots, O_k\}$ el conjunto de todos los objetos representados en $N$. Ahora, para una restricción

dada del tipo $(O_i\ \hat{R}_p\ O_j)$, donde $\hat{R}_p = \{R_1, \ldots, R_n\}$ y $1 \leq p \leq n$, abusando un poco de notación, denotaremos por $\phi_{\hat{R}}(O_i, O_j, \mathfrak{N}_p)$ a la fórmula en SpPNL que respeta $\hat{R}_p$, donde $\mathfrak{N}_p$ es el conjunto de todas las variables proposicionales $p_{R_l}$ utilizadas para simular nominales en $\phi_{\hat{R}}(O_i, O_j)$. Considérese la siguiente fórmula:

$$\phi(N) = \bigwedge_{i=1}^{k}(e(n(p(O_i)))) \wedge \bigwedge_{i=1}^{n}(u(\phi_{\hat{R}_i}(O_l, O_m, \mathfrak{N}_i))),$$

donde: $e(\phi)$ es el operador *existencial* (que se puede definir en SpPNL a partir del operador universal); para cada $i$, $p(O_i)$ es una letra proposicional; y para todo $1 \leq i, j \leq n$, si $i \neq j$ entonces $\mathfrak{N}_i \cap \mathfrak{N}_j = \emptyset$.

**Theorem 2.** *Sea $N = \langle \mathcal{O}, \mathcal{R} \rangle$ una* red-AR*. Entonces $N$ es consistente si y sólo si la fórmula $\phi(N)$ en* SpPNL *es satisfacible.*

## 5. Cómo aplicar SpPNL para razonar en el espacio

Considérese un espacio Euclídeo bidimensional basado en un conjunto linealmente ordenado $\mathbb{D} = \langle D, < \rangle$, y considérese el conjunto de todos los rectángulos no vacíos sobre dicho espacio. Se sabe que, en este caso, algunas relaciones del AR se pueden ver como relaciones topológicas (véase [15]). Por otra parte, para representar expresiones del lenguaje natural, las relaciones del AR pueden 'abstraerse' de varias formas. Por ejemplo, la relación *norte* podría coincidir con la relación $(e, m^{-1})$ (en este caso podría ser expresada directamente en SpPNL), o con la relación $\{(e, m^{-1}), (e, b^{-1}), (s, m^{-1}), (e, b^{-1}), (d, m^{-1}), (e, b^{-1})(f, m^{-1}), (f, b^{-1})\}$ (como en [20]). Se puede hacer una traducción similar para las relaciones topológicas. De esta manera, una red-AR puede representar expresiones tan naturales como: *La región (rectangular) $O_1$ está totalmente incluida en la región (rectangular) $O_2$, la cual está al sur de la región (rectangular) $O_3$*. Esta afirmación puede expresarse por una red-RA como:

$$I(O_1, O_2) \wedge S(O_2, O_3),$$

donde $I(O_1, O_2)$ representa a la restricción $O_1$ *es parte interior no-tangencial (ntpp) de* $O_2$, y $S(O_2, O_3)$ representa la restricción $O_2$ *está al sur de* $O_3$.

Sin embargo, existen muchos ejemplos de expresiones del lenguaje natural que no se pueden representar de la forma anterior. Por ejemplo, no sería posible representar una información *imprecisa* del tipo *la región (rectangular) $O_1$ está totalmente dentro de la región (rectangular) $O_2$ o la región (rectangular) $O_2$ está al sur de la región (rectangular) $O_3$*. Sin embargo, sí puede expresarse en SpPNL mediante la fórmula

$$\begin{aligned}\phi = e(n(p(O_1))) \wedge e(n(p(O_2))) \wedge e(n(p(O_3))) \wedge \\ (u(\phi_I(O_1, O_2, \mathfrak{N}_1)) \vee u(\phi_S(O_2, O_3, \mathfrak{N}_2))),\end{aligned}$$

donde $\mathfrak{N}_1$ y $\mathfrak{N}_2$ son dos conjuntos de letras proposicionales tales que $\mathfrak{N}_1 \cap \mathfrak{N}_2 \cap \{p(O_1), p(O_2), p(O_3)\} = \emptyset$, y $\phi_I(O_1, O_2, \mathfrak{N}_1)$ (resp. $\phi_S(O_1, O_2, \mathfrak{N}_2)$) es la fórmula en SpPNL para la restricción $I(O_1, O_2)$ (resp., $S(O_2, O_3)$).

Como ejemplo final de una expresión que no se pude representar en el AR, pero sí en SpPNL, consideramos el siguiente problema de tipo geográfico: *Conocemos un parque concreto (Pa). Sabemos que una de las siguientes afirmaciones es verdadera: 1) no hay casas (c) al norte del parque o 2) no hay casas (c) al sur del parque. También sabemos que la casa del Sr. García (Ga) está al norte del parque. Queremos saber si es posible que la casa del Sr. Pérez (Pe) esté al sureste del parque.* En este caso, interpretamos la relación *al norte de* como *en cualquier parte* al norte (teniendo sólo en cuenta la coordenada $y$), y de forma parecida para la relación *al sur de*. La siguiente fórmula en SpPNL:

$$\begin{aligned}\phi = {} & e(n(p(Pa))) \wedge e(n(p(Ga))) \wedge e(n(p(Pe))) \wedge \\ & (u(p(Pa) \rightarrow [\hat{N}]\neg c) \vee u(p(Pa) \rightarrow [\hat{S}]\neg c)) \wedge \\ & u(\phi_{\hat{N}}(Ga, Pa, \mathfrak{N}_1)) \wedge u(\phi_{\hat{SE}}(Pe, Pa, \mathfrak{N}_2)) \wedge \\ & u(p(Ga) \rightarrow c) \wedge u(p(Pe) \rightarrow c),\end{aligned}$$

representa el problema enunciado, donde $[\hat{N}]$ (resp. $[\hat{S}]$) es la modalidad (definible) para la relación *norte* (resp. *sur*) (con el significado dado anteriormente). Como se puede ver, la fórmula dada es insatisfacible.

## 6. Conclusiones y trabajo futuro

En este artículo hemos considerado una lógica modal, llamada Spatial Propositional Neighborhood Logic (SpPNL), para hacer razonamiento espacial. Hemos visto que, a pesar de su simplicidad, tiene el suficiente poder expresivo para representar cualquier red de restricciones del Álgebra de Rectángulos (AR). Además, hemos observado que en SpPNL somos capaces de representar situaciones espaciales intuitivas que no pueden expresarse en el AR. Como un posible trabajo futuro, pensamos investigar posibles fragmentos de la lógica que presenten un mejor comportamiento computacional, por ejemplo, restringiendo la forma de las fórmulas permitidas para algún dominio específico.

## Agradecimientos

Este trabajo ha sido financiado por el Ministerio de Educación y Ciencia con el proyecto TIC2003-09400-C04-01.

## Referencias

[1] M. Aiello and J. van Benthem. A modal walk through space. *Journal of Applied Non-Classical Logic*, 12(3-4):319–363, 2002.

[2] P. Balbiani, J.F. Condotta, and L. Fariñas del Cerro. A model for reasoning about bidimensional temporal relations. In *Proc. of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98)*, pages 124–130, 1998.

[3] P. Balbiani, J.F. Condotta, and L. Fariñas del Cerro. A new tractable subclass of the rectangle algebra. In *IJCAI*, pages 442–447, 1999.
[4] B. Bennett. Spatial reasoning with propositional logics. In Jon Doyle, Erik Sandewall, and Pietro Torasso, editors, *Proc. of Principles of Knowledge Representation and Reasoning Conference (KR'94)*, pages 51–62. Morgan Kaufmann, San Francisco, California, 1994.
[5] B. Bennett. Modal logics for qualitative spatial reasoning. *Journal of the Interest Group in Pure and Applied Logic (IGPL)*, 4(1):23–45, 1996.
[6] B. Bennett, A.G. Cohn, F. Wolter, and M. Zakharyaschev. Multi-dimensional modal logic as a framework for spatio-temporal reasoning. *Applied Intelligence*, 17(3):239–251, 2002.
[7] M.J. Egenhofer and R.D. Franzosa. Point set topological relations. *International Journal of Geographical Information Systems*, 5:161–174, 1991.
[8] V. Goranko, A. Montanari, and G. Sciavicco. Propositional interval neighborhood temporal logics. *Journal of Universal Computer Science*, 9(9):1137–1167, 2003.
[9] H. Güsgen. Spatial reasoning based on Allen's temporal logic. Technical Report ICSI TR89-049, International Computer Science Institute, 1989.
[10] C. Lutz and F. Wolter. Modal logics of topological relations. In *Proc. of Advances in Modal Logics 2004*, 2004.
[11] A. Montanari and G. Sciavicco. A decidability proof for propositional neighborhood logic. Contributed Talk, Trends in Logics III Conference, Warsaw - Ruciane Nida (Poland), 2005.
[12] A. Morales and G. Sciavicco. Using temporal logic for spatial reasoning: Spatial propositional neighborhood logic. In *Proc. of the International Symposium on Temporal Representation and Reasoning. In press*, 2006.
[13] A. Mukerjee and G. Joe. A qualitative model for space. In *AAAI*, pages 721–727, 1990.
[14] W.Ñutt. On the translation of qualitative spatial reasoning problems into modal logics. In *KI '99: Proc. of the 23rd Annual German Conference on Artificial Intelligence*, pages 113–124. Springer-Verlag, 1999.
[15] D. Papadias, T. Sellis, Y. Theodoridis, and M.J. Egenhofer. Topological relations in the world of minimum bounding rectangles: a study with R-trees. In *Proc. of the ACM SIGMOD Conference, San Jose, California*, pages 92–103, 1995.
[16] D.A. Randell, Z. Cui, and A. Cohn. A spatial logic based on regions and connection. In B.Ñebel, C. Rich, and W. Swartout, editors, *Proc. of the Third International Conference on Principles of Knowledge Representation and Reasoning*, pages 165–176, San Mateo, California, 1992. Morgan Kaufmann.
[17] G. Sciavicco. Temporal reasoning in propositional interval neighborhood logic. In *Proc. of the 2nd Language and Technology Conference, Poznan (Poland)*, pages 390 – 395, 2005.
[18] J. Sharma. *Integrated Spatial Reasoning in Geographic Information Systems: Combining Topology and Direction*. PhD thesis, University of Maine, 1996.
[19] S. Skiadopoulos and M. Koubarakis. Consistency checking for qualitative spatial reasoning with cardinal directions. In *Proc. of the 8th International Conference on Principles and Practice of Constraint Programming*, pages 341–355. Springer-Verlag, 2002.
[20] S. Skiadopoulos and M. Koubarakis. On the consistency of cardinal directions constraints. *Artificial Intelligence*, 163(1):91–135, 2005.
[21] H. Sun and W. Li. Integrated qualitative spatial reasoning. In *Proc. of the International Conference on Computational Intelligence*, pages 341–344, 2004.

# Deducción y generación de modelos de cardinalidad finita

Ángel Nepomuceno Fernández[1], Fernando Soler Toscano[1]
y Francisco J. Salguero Lamillar[2]

[1] Departamento de Filosofía y Lógica, Universidad de Sevilla,
C/ Camilo José Cela s/n, 41018 Sevilla
{nepomuce, fsoler}@us.es
[2] Dpto. de Lengua Española y Lingüística, Universidad de Sevilla,
C/ Palos de la Frontera s/n, 41004 Sevilla
salguero@us.es

**Resumen** Presentamos un procedimiento de tablas semánticas distinto de las estándar en el tratamiento dado a las sentencias de la clase $\delta$, proponiendo modificaciones de esta regla que eviten la generacíón de ramas infinitas. Las nuevas tablas se revelan útiles para el estudio de lenguajes lógico-formales con semántica finitaria, en particular como procedimiento para definir modelos finitos de sentencias satisfacibles. Mostramos también la aplicabilidad de estos resultados al análisis de la ambigüedad y otros fenómenos en la interpretación del discurso.

## 1. Introducción

Aunque el procedimiento de las tablas semánticas en primer orden, iniciado en [2], es semidecidible, si consideramos únicamente modelos finitos, mediante ciertas tablas cabe determinar de manera efectiva si una sentencia es o no satisfacible por modelos de cardinal finito $n \geq 1$. Este tipo de tablas surge, principalmente en [3] y [4], para abordar el problema de la generación de tablas infinitas de sentencias satisfacibles en dominios finitos y se desarrolla –en [10]– para hacer más eficientes los correspondientes algoritmos de refutación en modelos finitos. El método de tablas ha sido aplicado en ámbitos diversos, que van desde el estudio del razonamiento basado en modelos –ejemplo, [9]– hasta el tratamiento de problemas lingüísticos –como en [6] y en [8]–.

Tomando las aplicaciones al campo lingüístico como horizonte y con objeto de estudiar ciertos fenómenos en la interpretación del discurso, generalizamos el procedimiento de tablas modificadas, lo que permite examinar relaciones de consecuencia de carácter "supraclásico", así como método de obtención de modelos mínimos a partir de sentencias finitamente satisfacibles. Para ello partimos de un lenguaje formal de predicados de primer orden y en el siguiente apartado establecemos las principales nociones semánticas en términos de la teoría de modelos finitos, continuamos con un estudio de las propias tablas semánticas,

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 588-599, 2006.

estándar y modificadas, para concluir, en el último apartado, con una muestra resumida de apalicación en interpretación del discurso.

Así pues, $L$ representa el lenguaje formal, sin identidad ni functores, que posee un conjunto numerable de constantes individuales, de variables –ambos forman el conjunto de los términos– y de signos predicativos para cada aridad. Haremos uso de las nociones habituales de la sintaxis de estos lenguajes, tales como ocurrencia libre o ligada de variables, alcance de los signos lógicos, sustitución, etc. $\{\neg, \vee, \wedge, \rightarrow, \exists, \forall\}$ es el conjunto de las constantes lógicas. $L_{SEN}$ representa el conjunto de las sentencias de $L$. Para indicar que $A$ es una fórmula, anotaremos $A \in L$, si es una sentencia, $A \in L_{SEN}$ y si $\Gamma$ es un conjunto de fórmulas de $L$, $\Gamma \subseteq L$, en su caso, $\Gamma \subseteq L_{SEN}$. Si $A \in L$, $A(x/t)$ representa la fórmula resultante de sustituir en $A$ cada ocurrencia libre de la variable $x$ por el término $t$, siempre que éste no sea sufijo de un cuantificador bajo cuyo alcance ocurra $x$; así pues, $A(x/t) \in L$, en su caso $A(x/t) \in L_{SEN}$.

La semántica de $L$ se establece a partir de estructuras adecuadas a $L$, o $L$-estructuras, constan de un universo de discurso –o dominio– no vacío y una función interpretación para definir las denotaciones de constantes y signos predicativos como es habitual. $MOD(L)$ representa la clase de todas las $L$-estructuras. Las variables carecen de valores semánticos y sólo a las sentencias se asignará un valor de verdad del conjunto $\{0, 1\}$. Si $A \in L_{SEN}$ y $M \in MOD(L)$, mediante $M \vDash A$ o $M(A) = 1$ se indica que $M$ satisface $A$. Dadas las $L$-estructuras $M = \langle D, \Im \rangle$ y $M' = \langle D, \Im' \rangle$, donde $D \neq \emptyset$ es el dominio –en este caso, el mismo para las dos–, e $\Im$ e $\Im'$ las funciones interpretación, si $M$ y $M'$ difieren a lo sumo en cuanto a la interpretación de una constante $b$, es decir, si para cada constante $c \neq b$, $\Im(c) = \Im'(c)$ y para todo signo predicativo $R$, $\Im(R) \neq \Im'(R)$, mientras que $\Im(b)$ e $\Im'(b)$ pueden coincidir o tener distinto valor, ello lo expresaremos como $M =_b M'$; así, para $M \in MOD(L)$

1. $M(\forall x A) = 1$ si y sólo si –en adelante, syss– para toda $M' =_b M$, se verifica que $M(A(x/b)) = 1$
2. $M(\exists x A) = 1$ syss existe al menos una $M' =_b M$, $M(A(x/b)) = 1$.

## 2. Relaciones semánticas

Para la semántica nos interesan dominios finitos –una teoría de modelos finitos se presenta en [5]–. Para cada sentencia $A \in L$, diremos que $A$ es *finitamente satisfacible* syss existe $M \in MOD(L)$ tal que su dominio es de cardinal $n$ finito –menor que el de los números naturales, en símbolos, $n < \omega_0$– y $M \vDash A$. Si $M = \langle D, \Im \rangle$ y el cardinal de $D$ es $n \geq 1$, se indicará $|D| = n$, o bien $|M| = n$. La siguiente noción permite probar una versión del conocido teorema de Löwenheim-Skolem.

**Definición 1.** *$A \in L_{SEN}$ es $n$-satisfacible, para $n < \omega_0$ syss existe una $L$-estructura $M$, tal que $|M| = n$ y $M \vDash A$. Para indicar que $M$ tiene cardinal $n$ y satisface $A$, anotaremos $M \vDash_n A$.*

**Teorema 1.** *Si $A \in L_{SEN}$ es n-satisfacible, $1 \leq n < \omega_0$, entonces $A$ es m-satisfacible, para cada $m$ tal que $n \leq m < \omega$.*

*Demostración.* Se procede por inducción sobre el grado de complejidad de $A$. Si $A$ es $Rb_1...b_k$, $k \geq 1$, $M \vDash A$ y $|M| = n$, entonces, por definición, si $M = \langle D, \Im \rangle$, $\mid D \mid = n$ e $\langle \Im(b_1), ..., \Im(b_k) \rangle \in \Im(R)$; sea ahora $D^*$ tal que $D \subset D^*$ y $|D^*| = m > n$, definiendo $\Im^*$ de manera que, para cada constante $b_i$ que ocurra en la sentencia, $\Im^*(b_i) = \Im(b_i)$, además, $\Im(R) \subseteq \Im^*(R)$; una vez establecida $\Im^*$, sea $M^* = \langle D^*, \Im^* \rangle$, como, por definición, $\langle \Im^*(b_1), ..., \Im^*(b_k) \rangle \in \Im^*(R)$, se verifica que $M^* \vDash A$. Tomando como hipótesis de inducción que para cada sentencia de grado de complejidad $k \geq 1$, si es $n$-satisfacible, entonces también es $m$-satisfacible, para $m > n$, hemos de probar que ello se verifica para sentencias de grado de complejidad $k + 1$. Para abreviar, sólo nos referimos a los signos lógicos $\neg$, $\vee$ y $\exists$. Dado el supuesto, sea $A$ de la forma $\neg B$, grado de $B$ igual a $k \geq 1$; si existe $M$ tal que $|M| = n$ y $M \vDash B$, entonces existe $M^*$ tal que $|M^*| = m$, $m > n$ y $M^* \vDash B$; si $M = \langle D, \Im \rangle$, sea $M' = \langle D, \Im' \rangle$ de manera que $\Im'$ sea inversa de $\Im$, es decir, que para cada signo predicativo $R$ de aridad $k \geq 1$, $\Im'(R)$ sea justo complementario de $\Im(R)$; así, $M' \nvDash \beta$, entonces $M' \vDash \neg\beta$; asimismo, si $M^* = \langle D^*, \Im^* \rangle$, sea $M'^* = \langle D^*, \Im'^* \rangle$ tal que $M'^* \nvDash \beta$, de donde $M'^* \vDash \neg B$, pero $|M'| = n$ y $|M'^*| = m$. Consideremos que $A$ de la forma $B \vee C$, y la suma de los grados de $B$ y de $C$ es $k$; existe $M$ tal que $|M| = n$ y $M \vDash B$ o $M \vDash C$; naturalmente, $M \vDash B \vee C$; por otra parte, existe $M^*$ tal que $|M^*| = m$, $M^* \vDash B$ o $M^* \vDash C$; por evaluación de $\vee$, en efecto, $M^* \vDash A$ (cuyo grado es $k+1$). Por último, sea $A$ de la forma $\exists xB$, $k \geq 1$ es el grado de $B(x/b)$; si existe $M$ tal que $|M| = n$ y $M \vDash B(x/b)$, entonces existe $M'$ tal que $|M'| = m$, $m > n$ y $M' \vDash B(x/b)$; sea $M^* =_b M$ tal que $M^* \vDash \exists xB$ y sea $M'^* =_b M'$, tal que $M'^* \vDash \exists xB$, lo que es factible por evaluación de $\exists$, pero $|M^*| = n$ y $|M'^*| = m$.

Para cada conjunto de sentencias $\Gamma \subseteq L_{SEN}$ y cada sentencia $A \in L_{SEN}$, es conocida la relación de *consecuencia lógica*, en sentido clásico, como subconjunto de $\wp(L_{SEN}) \times L_{SEN}$, según la siguiente

**Definición 2.** *$\Gamma \vDash A$ syss se verifica que para toda $M \in MOD(L)$, si $M \vDash \Gamma$, entonces $M \vDash A$.*

Esta relación verifica las conocidas reglas estructurales de *reflexividad, permutación, contracción, monotonía* y *transitividad (o corte)* –las características de la relación clásica, así como las de otras relaciones que definen las llamadas *lógicas subestructurales*, se estudian tanto en [7] como en [12]–. Cualquier relación definida como subconjunto de $\wp(L_{SEN}) \times L_{SEN}$ que verifique reflexividad, monotonía y transitividad, se dice que es *supraclásica*. Por otra parte, la relación clásica $\vDash$ verifica también compacidad, es decir, que, para cada conjunto $\Gamma \subseteq L_{SEN}$ y $A \in L_{SEN}$, si $\Gamma \vDash A$, entonces existe $\Delta \subseteq \Gamma$ tal que su cardinal es menor que el cardinal de los naturales –en símbolos, $|\Delta| < \omega_0$– y $\Delta \vDash A$.

Definimos ahora otras nociones a partir de las precedentes.

**Definición 3.** *Para $1 \leq n < \omega_0$, para cada $\Gamma \subseteq L_{SEN}$ y $A \in L_{SEN}$, $A$ es $n$-consecuencia de $\Gamma$ syss para cada $M \in MOD(L)$ tal que $|M| \leq n$, si $M \vDash \Gamma$, entonces $M \vDash A$. Simbólicamente, lo representamos como $\Gamma \vDash_n A$.*

**Definición 4.** *Para cada* $\Theta \subseteq L_{SEN}$, $A \in L_{SEN}$, *y* $M \in MOD(L)$, $M$ *satisface* $A$ *módulo* $\Theta$ *—en símbolos,* $M \models_\Theta A$*— syss* $M \models \Theta \cup \{A\}$. *En ralación con la cardinalidad* $n < \omega_0$, $M$ *satisface* $A$ *módulo* $\Theta$ *—en símbolos,* $M \models_{\Theta/n} A$*— syss* $\mid M \mid = n$ *y* $M \models_\Theta A$.

**Definición 5.** *Para cada* $\Gamma, \Theta \subseteq L_{SEN}$, $A \in L_{SEN}$, $A$ *es consecuencia lógica de* $\Gamma$ *módulo* $\Theta$ *syss* $\Gamma, \Theta \models A$. *Simbólicamente,* $\Gamma \models_\Theta A$ *syss* $\Gamma, \Theta \models A$. *En cuanto a la cardinalidad* $n < \omega_0$, $A$ *es consecuencia lógica de* $\Gamma$ *módulo* $\Theta$ *limitada a un número* $n \geq 1$, *en símbolos* $\Gamma \models_{\Theta/n} A$, *syss* $\Gamma, \Theta \models_n A$.

Así pues, estas relaciones vienen definidas a partir de las nociones de $n$-satisfacibilidad, consecuencia módulo un conjunto de sentencias y $n$-consecuencia módulo un conjunto de sentencias.

**Teorema 2.** *Las relaciones* $\models_n$, $\models_\Theta$ *y* $\models_{\Theta/n}$ *son supraclásicas.*

*Demostración.* Para simplificar, escribimos $\Gamma, \Delta \models A$ en lugar de $\Gamma \cup \Delta \models A$ y $\Gamma, A \models B$ en lugar de $\Gamma \cup \{A\} \models B$. Hemos de comprobar que verifican reflexividad, monotonía y transitividad. En efecto

$$\text{si } A \in \Gamma, \text{ entonces } \Gamma \models_n A,\ \Gamma \models_\Theta A \text{ y } \Gamma \models_{\Theta/n} A.$$

Por otra parte,

$$\frac{\Gamma \models_n A}{\Gamma, \Delta \models_n A};\ \frac{\Gamma \models_\Theta A}{\Gamma, \Delta \models_\Theta A};\ \frac{\Gamma \models_{\Theta/n} A}{\Gamma, \Delta \models_{\Theta/n} A},$$

dado que, en cuanto a $\models_n$, si toda $M \in MOD(L)$, $\mid M \mid = n$, si $M \models \Gamma$, entonces $M \models A$, si $M \models \Gamma \cup \Delta$, entonces $M \models A$. En cuanto a las otras dos, teniendo en cuenta cómo se definen, se verifica

$$\frac{\Gamma, \Theta \models A}{\Gamma, \Delta, \Theta \models A};\ \frac{\Gamma, \Theta \models_n A}{\Gamma, \Delta, \Theta \models_n A},$$

respectivamente. Por último, de acuerdo con las definiciones, claramente se verifica

$$\frac{\Gamma, A \models_n B;\ \Gamma, B \models_n C}{\Gamma, A \models_n C};$$

y, dado que la relación clásica es transitiva,

$$\frac{\Gamma, \Theta, A \models B;\ \Gamma, \Theta, B \models C}{\Gamma, \Theta, A \models C},$$

también se verifica

$$\frac{\Gamma, A \models_\Theta B;\ \Gamma, B \models_\Theta C}{\Gamma, A \models_\Theta C}.$$

## 3. Tablas semánticas

Las reglas (clásicas) permiten generar, a partir de un conjunto finito de sentencias llamado *raíz*, al menos una de las cuales no es un literal, un conjunto de sucesiones de sentencias llamadas *ramas*; cada rama se va construyendo hasta que aparece un par de contradicción (dos literales complementarios) –sería rama cerrada–, o bien hasta que en cada sentencia no literal de la rama se haya ejecutado plenamente la regla correspondiente –sería rama abierta–. Una tabla es abierta si al menos una de sus ramas es abierta y cerrada en otro caso. Las reglas son las siguientes

1. Doble negación:
$$\frac{\neg\neg A}{A},$$
la rama continúa incorporando $A$.
2. Regla $\alpha$:
$$\frac{A_1 \wedge A_2}{A_1,\, A_2};\ \frac{\neg(A_1 \vee A_2)}{\neg A_1,\, \neg A_2};\ \frac{\neg(A_1 \rightarrow A_2)}{A_1,\, \neg A_2},$$
la rama continúa con cada uno de los componentes: $A_1$ y $A_2$ etc.
3. Regla $\beta$:
$$\frac{A_1 \vee A_2}{A_1 \mid A_2};\ \frac{\neg(A_1 \wedge A_2)}{\neg A_1 \mid \neg A_2};\ \frac{A_1 \rightarrow A_2}{\neg A_1 \mid A_2},$$
en este caso, la rama se subdivide en dos, lo que se indica mediante "|", continuando cada una con uno de los componentes,
4. Regla $\gamma$:
$$\frac{\forall x B}{B(x/b_1), B(x/b_2), ..., B(x/b_n)},$$
$B(x/b_i)$ es la sentencia resultante de sustituir en la fórmula $B$ cada ocurrencia de la variable libre $x$ por la constante $b_i$, para $i \leq n$; $b_1, ..., b_n$ son todas las constantes que ocurren en sentencias de la rama
5. Regla $\delta$:
$$\frac{\exists x B}{B(x/b_{n+1})},$$
donde $b_{n+1}$ es la primera constante que no ocurría en la rama.

La propiedad fundamental de las tablas para lógica clásica, que podemos denominar *estándar*, se expresa en el siguiente teorema, que enunciamos sin demostración para abreviar –versiones del mismo aparecen en [2] y en trabajos editados en [1]–.

**Teorema 3.** *Un conjunto (finito) $\Gamma \subseteq L_{SEN}$ es satisfacible syss la tabla semántica de raíz $\Gamma$, abreviadamente $T(\Gamma)$, es abierta.*

**Corolario 1.** *Un conjunto (finito) $\Gamma \subseteq L_{SEN}$ es no satisfacible syss $T(\Gamma)$ es cerrada.*

*Demostración.* Es consecuencia inmediata del teorema 3 por simple contraposición.

**Teorema 4.** *Para cada conjunto finito* $\Gamma \subseteq L_{SEN}$, $A \in L_{SEN}$, $\Gamma \vDash A$ *syss* $T(\Gamma \cup \{\neg A\})$ *es cerrada.*

*Demostración.* $\Gamma \vDash A$ syss $\Gamma \cup \{\neg A\}$ es no satisfacible syss $T(\Gamma \cup \{\neg A\})$ es cerrada, de acuerdo con teorema 1.

Con objeto de aplicar el procedimiento a problemas relativos a otras relaciones de consecuencia supraclásicas, definimos modificaciones de la regla $\delta$, de acuerdo con lo sugerido en [3] y [4], así como de la regla $\gamma$:

6. Regla $\delta^{[+]}$:

$$\frac{\exists x B}{B(x/t_1) \mid B(x/t_2) \mid, ..., \mid B(x/t_n) \mid B(x/t_{n+1})},$$

siendo $t_1, ..., t_n$ las constantes que ocurrían en la rama hasta llegar a $\exists x B$ mientras que $t_{n+1}$ es una constante nueva.

7. Regla $\delta^{[n]}$:

$$\frac{\exists x B}{B(x/t_1) \mid B(x/t_2) \mid, ..., \mid B(x/t_n)},$$

$t_1, ..., t_n$ son $n \geq 1$ constantes de $L$, siempre que el número de constantes de la rama sea $m \leq n$, de manera que si éstas son $b_k, ..., b_{k+t}$, donde $t+1 = m$, entonces $t_i$ es $b_{k+i-1}$ para cada $i \leq m$. En caso de que $m > n$, entonces no es aplicable la regla.

8. Regla $\gamma^{[n]}$:

$$\frac{\forall x B}{B(x/t_1),\ B(x/t_2),\ ...,\ B(x/t_n)},$$

con la misma restricción que en el caso anterior.

**Definición 6.** *Dada una raíz* $\Gamma \subseteq L_{SEN}$, $\Delta^+(\Gamma)$ *es la tabla construida con las reglas de doble negación,* $\alpha$, $\beta$, $\gamma$ *y* $\delta^{[+]}$. *Decimos que* $\Delta^+(\Gamma)$ *es una tabla* $\Delta^+$ *o, para abreviar, +-tabla. Si en* $\Gamma$ *no ocurre ninguna sentencia de la clase* $\delta$, *entonces* $\Delta^+(\Gamma) = T(\Gamma)$.

**Definición 7.** *Dada una raíz* $\Gamma \subseteq L_{SEN}$, $\Delta^n(\Gamma)$ *es la tabla construida con las reglas de doble negación,* $\alpha$, $\beta$, $\gamma^{[n]}$ *y* $\delta^{[n]}$, *siempre que 1) en* $\Gamma$ *el número de constantes sea* $m \leq n$ *y 2) si en* $\Gamma$ *no ocurre ninguna sentencia cuantificacional, entonces se añade a la raíz la clausura existencial de las sentencias de* $\Gamma$. *Decimos, en este caso, que se trata de una tabla* $\Delta^n$ *o, para abreviar, que es una n-tabla.*

**Definición 8.** *Dada una tabla (estándar, +-tabla o n-tabla) de raíz* $\Gamma$, *cada rama abierta* $\Phi$ *define una interpretación o modelo canónico* $M^c = \langle D, \Im^c \rangle$ *tal que*

*(1)* $D$ *es un universo Herbrand; es decir, los elementos de* $D$ *son todas las*

*constantes $t_i$ tales que $i \leq m$ y $t_1$, ..., $t_m$ ocurren en $\Phi$, $m \geq 1$.*

*(2) Para cada $b_i \in D$, $\Im^c(t_i) = t_i$, $i \leq m$.*

*(3) Para cada signo predicativo k-ádico $R$ que ocurra en sentencias de $\Phi$, y cada k-pla $\langle t_1, ..., t_k \rangle$ de elementos de $D$,*

$$\langle t_1, ..., t_k \rangle \in \Im^c(R) \text{ syss } Rt_1, ..., t_k \in \Phi.$$

Se pueden establecer algunos resultados. Para mayor comodidad, si $A \in L_{SEN}$ y $\{A\}$ es la raíz de una tabla, anotaremos $T(A)$, $\Delta^+(A)$ y $\Delta^n(A)$ en lugar de $T(\{A\})$, $\Delta^+(\{A\})$ y $\Delta^n(\{A\})$, respectivamente. Por otra parte, dada una rama $\Phi$ de una tabla, el resultado de añadir una sentencia $A$ a la rama lo representamos como $\Phi + A$.

**Teorema 5.** *Para cada conjunto finito $\Gamma \subseteq L_{SEN}$, $T(\Gamma) \subseteq \Delta^+(\Gamma)$.*

*Demostración.* En efecto, sea $\Phi$ una rama de $T(\Gamma)$ en la que todavía no se haya aplicado la regla $\delta$; podemos iniciar $\Delta^+(\Gamma)$ siguiendo el mismo orden de aplicación de las reglas, entonces tendremos una rama $\Phi'$ tal que $\Phi = \Phi'$; al aplicar la regla $\delta$, $\Phi$ continúa con $B(x/b_{n+1})$, se obtiene, pues, $\Phi + B(x/b_{n+1})$, siendo $b_n$ la constante de mayor subíndice que ocurre en $\Phi$; al aplicar la regla $\delta^{[+]}$, se obtienen las siguientes ramas $\Phi' + B(x/b_1)$, $\Phi' + B(x/b_2)$, ..., $\Phi' + B(x/b_{n+1})$, pero $\Phi' + B(x/b_{n+1}) = \Phi + B(x/b_{n+1})$. Iterando el proceso, siempre hallaremos una rama de $T(\Gamma)$ en $\Delta^+(\Gamma)$, por lo que $T(\Gamma) \subseteq \Delta^+(\Gamma)$.

**Corolario 2.** *Un conjunto finito $\Gamma \subseteq L_{SEN}$ es satisfacible syss $\Delta^+(\Gamma)$ es abierta.*

*Demostración.* Consecuencia inmediata de los teoremas 3 y 5.

**Teorema 6.** *$A \in L_{SEN}$ es n-satisfacible, $1 \leq n < \omega_0$, syss $\Delta^n(A)$ posee una rama abierta con las constantes $b_1, ..., b_n$.*

*Demostración.* $\Leftarrow$) Supongamos que $\Delta^n(A)$ es abierta: una rama $\Phi$ es abierta. Cada aplicación de $\delta^{[n]}$ produce $n$ bifurcaciones y el número de constantes en $\Phi$ es $n$. Sea $M^c$ el correspondiente modelo canónico. Por inducción sobre el modo de obtener la sentencia $B \in \Phi$, comprobamos que $M^c$ satisface todas las sentencias de $\Phi$ y, por tanto, a la raíz $A$. En la base, si $B$ es atómica, entonces es $Rt_1, ..., t_k$, por lo que $\langle \Im(t_1), ..., \Im(t_k) \rangle \in \Im(R)$, así que $M^c \vDash B$. Hipótesis de inducción: $M^c \vDash C$, para cada sentencia $C$ consecuente de una regla de la $n$-tabla hasta cierto grado. Si la regla es la de doble negación, entonces $M^c \vDash \neg\neg C$. Si se trata de la regla $\alpha$, entonces el antecedente sería $C \wedge C'$, $\neg(D \vee C')$ o $\neg(C \rightarrow C')$ —en el segundo caso, $C = \neg D$—, pero dado que $C' \in \Phi$, en el primer caso, o $\neg C' \in \Phi$ en el segundo y el tercero, $M^c \vDash C'$ o $M^c \vDash \neg C'$, respectivamente; basta con evaluar los signos lógicos correspondientes y tendremos que $M^c \vDash C \wedge C'$, o $M^c \vDash \neg(D \vee C')$ o $M^c \vDash \neg(C \rightarrow C')$. De manera análoga se comprueba si se trata del antecedente de la regla $\beta$. En cuanto a la regla $\gamma$, si $B$ es $C(x/b_k)$,

para algún $k \leq n$, también estarán en $\Phi$ todas las sentencias que constituyen el consecuente de la regla: $C(x/b_1), ..., C(x/b_n)$, y como $M^c \vDash C(x/b_i)$ para todo $i \leq n$, por evaluación de $\forall$, $M^c \vDash \forall x C$. Por último, si se trata de la regla $\delta^{[n]}$, $B$ tiene la forma $C(x/b_k)$, para algún $k \leq n$; si $M^c \vDash C(x/b_k)$, entonces $M^c \vDash \exists x C$. Así pues, $M^c$ satisface cada literal de $\Phi$, y cada sentencia es consecuente de alguna regla, de manera que $M^c$ satisface todas las sentencias de $\Phi$, por tanto, $M^c \vDash B$.

$\Rightarrow$) Supongamos que $A$ es $n$-satisfacible: existe $M \in MOD(L)$, $M \vDash_n A$. Se construye $\Delta^n(A)$ y hacemos inducción sobre el número de veces de aplicación de las reglas. Como base 0, aplicaciones; en ese momento la rama $\Phi = \{A\}$ y, por hipótesis, $M \vDash_n A$. Supuesto inductivo: se han hecho $k \geq 1$ aplicaciones y obtenido la rama $\Phi$ tal que $M \vDash_n \Phi$; en la aplicación $k+1$ la sentencia es $X$, tenemos las siguientes posibilidades de obtener $\Phi'$:

1. $X$ es $\neg\neg B$ y $\Phi' = \Phi + B$,
2. $X$ es $A_1 \wedge A_2$, $\neg(A_1 \vee A_2)$ o $\neg(A_1 \rightarrow A_2)$; $\Phi'$ será $\Phi + A_1 + A_2$, $\Phi + \neg A_1 + \neg A_2$ o $\Phi + A_1 + \neg A_2$, respectivamente,
3. $X$ es $A_1 \vee A_2$, $\neg(A_1 \wedge A_2)$ o $A_1 \rightarrow A_2$; $\Phi'$ será $\Phi + A_1$ o $\Phi + A_2$, $\Phi + \neg A_1$ o $\Phi + \neg A_2$, o $\Phi + \neg A_1$ o $\Phi + A_2$, respectivamente,
4. $X$ es $\forall x B$ y $\Phi' = \Phi + B(x/b_1) + ... + B(x/b_n)$,
5. $X$ es $\exists x B$ y $\Phi'$ es $\Phi + B(x/b_1)$ o $\Phi + B(x/b_2)$ o ... o $\Phi + B(x/b_n)$.

Se comprueba (omitimos detalles para abreviar) que en alguna de las formas de $\Phi'$, $M \vDash_n \Phi'$, de donde $\Phi'$ es abierta –para que fuese cerrada habría de contener $C$ y $\neg C$, pero $M \nvDash_n C \wedge \neg C$–. Hay pues una rama acabada y abierta. Por otra parte, en esta rama ocurren las constantes $b_1, ..., b_n$, ya sea porque ocurrían en $A$, porque apareciera en algún momento en la rama una sentencia de la forma $\exists x B$ o se introdujera, de acuerdo con la definición de $\Delta^n(A)$.

**Corolario 3.** *$A \in L_{SEN}$ no es $n$-satisfacible, $1 \leq n < \omega_0$, syss $\Delta^n(A)$ es cerrada. También podemos decir en este caso que $A$ es $n$-insatisfacible.*

*Demostración.* Es consecuencia inmediata del teorema 6 por contraposición.

**Corolario 4.** *Para cada conjunto finito $\Gamma \subseteq L_{SEN}$, $A \in L_{SEN}$, $\Gamma \vDash_n A$ syss $\Delta^n(\Gamma \cup \{\neg A\})$ es cerrada.*

*Demostración.* Consecuencia inmediata del teorema 3 y de las definiciones 1 y 4.

**Corolario 5.** *Para los conjuntos finitos $\Gamma, \Theta \subseteq L_{SEN}$, $A \in L_{SEN}$, $\Gamma \vDash_{\Theta/n} A$ syss $\Delta^n(\Gamma \cup \Theta \cup \{\neg A\})$ es cerrada.*

*Demostración.* Consecuencia inmediata del corolario 4 y de la definición 4.

### 3.1. Generación de modelos

Así pues, las tablas modificadas constituyen un procedimiento para estudiar las relaciones de consecuencia antes indicadas; asimismo permiten la obtención de modelos de sentencias satisfacibles en dominios finitos. Una cuestión que se plantea para el caso de sentencias finitamente satisfacibles es hallar su modelo mínimo (la $L$-estructura de menor cardinal que la satisface).

**Definición 9.** *Dada una sentencia $A \in L_{SEN}$, $M \in MOD(L)$ es un modelo mínimo de $A$ syss $M \vDash A$, y para cada $M' \in MOD(L)$ tal que $|M'| < |M|$, $M' \nvDash A$.*

**Teorema 7.** *Si el conjunto finito $\Gamma \subseteq L_{SEN}$ es finitamente satisfacible, entonces $\Delta^+(\Gamma)$ posee una rama abierta que define un representante de la clase de sus modelos mínimos.*

*Demostración.* Teniendo en cuenta la propiedad fundamental de las tablas estándar (teorema 3) y el corolario 2, existe al menos una rama acabada abierta. Sea $\Phi$ la primera rama no cerrada de $\Delta^+(\Gamma)$ en la que ocurren $n \geq 1$ constantes; y sean éstas $b_1, ..., b_n$. Naturalmente es definible un modelo canónico $M^c$, $|M^c| = n$, el cual satisface todas las sentencias de $\Phi$, por tanto a $\Gamma$; si $\Gamma$ fuera $m$-satisfacible para algún $m < n$, entonces sería definible un modelo canónico a partir de las constantes $b_1, ..., b_m$, lo cual es imposible, ya que ninguna rama con $m$ constantes es abierta.

**Teorema 8.** *Si el conjunto finito $\Gamma \subseteq L_{SEN}$ es finitamente satisfacible, entonces existe $n \geq 1$, tal que $\Delta^n(\Gamma)$ es abierta y cada modelo mínimo de $\Gamma$ es de cardinal $n$.*

*Demostración.* Si $\Gamma$ es finitamente satisfacible, existe $M$ tal que $M \vDash \Gamma$, $|M| = n < \omega_0$ y $M$ es un modelo mínimo. Ello equivale a que $\Delta^n(\Gamma)$ sea abierta. Se puede hallar $n$ mediante $\Delta^1(\Gamma)$, $\Delta^2(\Gamma)$, ..., $\Delta^{n-1}(\Gamma)$, todas ellas cerradas, hasta alcanzar la primera $\Delta^n(\Gamma)$ abierta.

Así pues, si un conjunto finito de sentencias es satisfacible, mediante las tablas $\Delta^n$ podemos obtener modelos canónicos de cardinal $n$, siempre que los modelos mínimos sean de cardinalidad $m \leq n$. Estos modelos se definen a partir de los literales de cada rama abierta de la tabla, por ello diremos que los conjuntos de tales literales "inducen tal o cual modelo".

**Definición 10.** *Si $\Phi$ es una rama finita de una tabla (estándar, +-tabla o n-tabla) de raíz $\Gamma$, con $m \geq 1$ constantes, diremos que el conjunto de sus literales es una base de $\Phi$. Si $B(\Phi)$ es una base de $\Phi$, $\overline{B(\Phi)}$ representa un conjunto de cierre de $\Phi$ y se define*

$$\overline{B(\Phi)} = \{\sim A / A \in B(\Phi)\},$$

*donde $\sim A$ representa el literal complementario de $A$.*

**Teorema 9.** *Una rama finita $\Phi$ de una tabla (estándar, +-tabla o n-tabla) es abierta syss $B(\Phi) \cap \overline{B(\Phi)} = \emptyset$.*

*Demostración.* Consecuencia inmediata de la definición 10: al ser $\Phi$ abierta, en $B(\Phi)$ no ocurren literales complementarios. En caso de que $B(\Phi) \cap \overline{B(\Phi)} \neq \emptyset$, tendría que haber algún literal $A \in B(\Phi)$ y $A \in \overline{B(\Phi)}$, con lo que $\sim A \in B(\Phi)$ y $\sim A \in \overline{B(\Phi)}$, contra la hipótesis de que $\Phi$ es abierta.

**Corolario 6.** *Para cada $\Gamma, \Theta \subseteq L_{SEN}$, tales que $\Phi$ y $\Phi'$ son ramas abiertas de sus respectivas n-tablas, $n < \omega_0$, entonces es definible $M$ inducida por $B(\Phi+\Phi')$, tal que $M \vDash_{\Theta/n} \Gamma$, syss*

$$B(\Phi + \Phi') \cap \overline{B(\Phi + \Phi')} = \emptyset.$$

*Demostración.* Dado que $M \vDash_{\Theta/n} \Gamma$, por definición $M \vDash_n \Theta \cup \Gamma$, de manera que $\Delta^n(\Gamma \cup \Theta)$ tiene alguna rama abierta con $n$ constantes, sea, por ejemplo, $\Phi^*$. Entonces

$$\Phi^* = \gamma_1 + ... + \gamma_m + \theta_1 + ... + \theta_k + (\Phi - \Gamma) + (\Phi' - \Theta),$$

pero $\Phi^*$ y $\Phi + \Phi'$ contienen los mismos literales, así que $B(\Phi^*) = B(\Phi + \Phi')$, además, como $\Phi^*$ es abierta, se verifica que $B(\Phi + \Phi') \cap \overline{B(\Phi + \Phi')} = \emptyset$. Recíprocamente, si $B(\Phi + \Phi') \cap \overline{B(\Phi + \Phi')} = \emptyset$, para un número $n \geq 1$ de constantes, entonces es definible $M$ tal que $M \vDash_n \Theta \cup \Gamma$, por lo que $M \vDash_{\Theta/n} \Gamma$.

Hemos de tener en cuenta que, de acuerdo con la notación adoptada, si $B(\Phi) = B_1$ y $B(\Phi') = B_2$, podemos abreviar $B(\Phi + \Phi')$ como $B_1 + B_2$.

## 4. Una aplicación lingüística

Estudiamos brevemente un ejemplo; sea la frase «*Una mujer encontró un gato en el jardín. Le gustó*». La estructura lógica de ésta, en la que aparece la expresión anafórica pronominal "le", además de la anáfora pronominal relacionada con el pronombre elidido que realiza la función de sujeto morfosintáctico del verbo "gustar", aunque también sea su objeto lógico, se puede representar como la sentencia

$$\exists x, y, z(mujer(x) \wedge gato(y) \wedge jardín(z) \wedge encontró(x, y, z)) \wedge \exists x, y(gustó(y, x)),$$

a la que llamaremos $A$. Esta sentencia es satisfacible, pero mediante tablas modificadas obtenemos sus modelos mínimos: es 1-satisfacible, por tanto, también 2- y 3-satisfacible. Aunque por limitacionos de espacio omitimos detalles, mediante aplicación del método hallamos que la 3-tabla tiene varias ramas abiertas. Así pues, veamos sólo las bases de algunas de ellas, las que combinan la primera constante que aparece en la tabla con las demás:

$$B_1 = \{mujer(b_1), gato(b_2), jardín(b_3), encontró(b_1, b_2, b_3), gustó(b_1, b_1)\},$$

$$B_2 = \{mujer(b_1), gato(b_2), jardín(b_3), encontró(b_1, b_2, b_3), gustó(b_2, b_1)\},$$
$$B_3 = \{mujer(b_1), gato(b_2), jardín(b_3), encontró(b_1, b_2, b_3), gustó(b_3, b_1)\}.$$

Como puede observarse, en los tres casos se resuelve la anáfora introducida por el pronombre "le" relacionando su valor semántico con la interpretación de alguna de las constantes anteriores en la tabla. El hecho de mantener fija la interpretación de $b_1$ se debe sólo a la simplificación que hacemos del análisis oracional para no entrar aquí en la discusión del valor del pronombre elidido, por las peculiares características sintáctico-semánticas del verbo "gustar" en español.

¿Por qué no optar por el modelo que nos proporciona la primera clase? En realidad la tabla da lugar a 3 bases que permiten definir otros tantos modelos de la sentencia, pero si consideramos $\Theta = \{\forall x \neg gustó(x, x)\}$, y hacemos su 3-tabla, hallamos la siguiente base de su única rama abierta:

$$B' = \{\neg gustó(b_1, b_1), \neg gustó(b_2, b_2), \neg gustó(b_3, b_3)\}.$$

Es evidente que desde $B_1 + B'$ el modelo contiene dos literales complementarios, $gustó(b_1, b_1)$ y $\neg gustó(b_1, b_1)$, los cuales también pertenecen a $\overline{B_1 + B'}$, así que no es definible $M$ inducida por $B_1 + B'$ tal que $M \vDash_{\Theta/3} A$; sin embargo, desde $B_2 + B'$ y desde $B_3 + B'$ son definibles sendas $M'$ y $M''$ tales que $M' \vDash_{\Theta/3} A$ y $M'' \vDash_{\Theta/3} A$, pues en ambos casos estamos ante bases sin literales complementarios. La ampliación de $\Theta$ añadiendo sentencias que representan cierto conocimiento lingüístico como, por ejemplo, acerca de preferencias de argumentos en las relaciones correspondientes, podría dar lugar a nuevas $L$-estructuras: sea $\Theta'$ una ampliación tal, entonces aplicando el método de tablas se podría obtener $M$ tal que $|M| = 3$ y $M \vDash_{\Theta'/3} A$. En estas consideraciones no hemos tenido en cuenta la relación triádica $encontró(x, y, z)$, pero en alguna de las tablas ocurriría con el mismo argumento; precisamente una sentencia del tipo $\forall x, y(\neg encontró(x, x, y) \wedge \neg encontró(x, y, x) \wedge \neg encontró(y, x, x))$ vendría a incrementar conocimiento lingüístico y al añadirla a $\Theta$, obteniendo así $\Theta'$, haría descartar los casos de repetición de argumentos, además, establecida la prioridad de casos en que el primer argumento coincida con el de $mujer(x)$, podrá arrojar luz sobre las preferencias para la otra relación (diádica) presente (finalmente se llegaría a $M$ tal que $M \vDash_{\Theta'/3} A$).

Este procedimiento nos permite obtener modelos estáticos, si bien la sucesiva aplicación de las n-tablas permite ampliar modelos que satisfacen ciertas sentencias iniciales. Estas últimas podrían representar un mayor conocimiento del mundo a la hora de aplicar el procedimiento a la explicación de determinados fenómenos lingüísticos, como la resolución de expresiones anafóricas pronominales en las que se da una presuposición de existencia, o bien, como se ha sugerido más arriba, podrían representar un conocimiento lingüístico ligado a un mayor conocimiento del mundo, lo que vendrá a determinar la interpretación que se adopte. Este conocimiento "extra", no codificado gramaticalmente en la oración, forma parte del contexto o la situación en la que la sentencia $A$ debe ser interpretada. Las diferentes configuraciones que puede tomar $\Theta$, por tanto, dan lugar a una interpretación dinámica de la sentencia $A$. Estas posibles configuraciones de $\Theta$ deben estar sujetas a reglas de selección de las constantes a utilizar

en la resolución de las expresiones anafóricas (v. gr.: orden de aparición en la n-tabla o cualquier otro tipo de relación de orden que pueda ser definida).

Por último, señalamos algunas líneas de investigación. De un lado, la determinación de un contexto, a partir de una información inicial, puede abordarse como el estudio de un proceso abductivo; se han presentado métodos para la resolución de problemas abductivos: hallar sentencias, mediante tablas, que junto con una teoría expliquen (de manera no trivial) la sentencia representativa de un hecho dado –por ejemplo, en [9] y en [11]–. De otro, la construcción de modelos basados en una perspectiva dinámica –como, por ejemplo, en [8]– admite un replanteamiento en términos de $n$-tablas, con las modificaciones pertinentes para poder tratar fórmulas con variables libres. Asimismo, resultará de interés la comparación de estos procedimientos con la aplicáción en semántica formal de la lógica de predicados dinámica, especialmente el estudio de las relaciones anafóricas, etc.

## Referencias


1. D'Agostino, M.; Gabbay, D. M.; Hähnle, R.; Possega, J.: *Handbook of Tableau Methods,* Dordrecth/Boston/London, Kluwer Academic Press (1999)
2. Beth, E.W.: Semantic entailment and formal derivability. Koninklijke Nederlandse Akademie van Wentenschappen, Proceedings of the Section of Sciences **18** (1955) 309–342
3. Boolos, G.: Trees and finite satsifactibility. Notre Dame Journal of Formal Logic (25) (1984) 110–115
4. Díaz Estévez, E.: Árboles semánticos y modelos mínimos. In: Actas del I Congreso de la Sociedad de Lógica, Metodología y Filosofía de la Ciencia en España, Universidad Complutense de Madrid (1993) 40
5. Ebbinnghaus, H. D.; Flum, G.: *Finite Model Theory.* Springer-Verlag, Berlín, Heidelberg, New York (1999)
6. Kohlhase, M.: Model Generation for Discourse Representation Theory ECAI (2000).
7. Makinson, D.: Bridges from Classical to Nonmonotonic Logic. King's College, London (2005).
8. Monz, C., de Rijke, M.: A Tableau Calculus for Pronoun Resolution. In Murray, N.V. (ed): Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX'99). Springer (1999) 247-262
9. Nepomuceno, A.: Scientific explanation and modified semantic tableaux. In Magnani, L., Nerssessian, N., Pizzi, C., eds.: Logical and Computational Aspects of Model-Based Reasoning. Applied Logic Series. Kluwer Academic Publishers (2002) 181–198
10. Peltier, N.: A More Efficient Tableaux Procedures for Simultaneous Search for Refutations and Finite Models. IJCAI (2003)
11. Reyes-Cabello, L., Aliseda-Llera, A., Nepomuceno-Fernández, A.: Towards abductive reasoning in first order logic. Logic Journal of the IGPL, 14, 2 (2006)
12. Schroeder-Heister, P., Dośen, K. (eds.): *Substructural Logics.* Oxford Science Publications. Oxford (1999)

# Programando con igualdad similar estricta

Ginés Moreno y Vicente Pascual

Departmento de Sistemas Informáticos
Universidad de Castilla-La Mancha, Campus Universitario, 02071 Albacete
{gmoreno,vpascual}@info-ab.uclm.es

**Resumen** En los últimos años, hemos sido testigos del importante papel que la lógica difusa o borrosa (*fuzzy logic*) ha jugado en el desarrollo de sofisticadas aplicaciones software en campos tan diversos como los sistemas expertos, inteligencia artificial, control industrial, medicina, etc. Con el objetivo de facilitar el desarrollo de tales aplicaciones, mucho más recientemente ha surgido el interés por diseñar lenguajes declarativos (funcionales, lógicos y/o difusos) que incorporen entre sus recursos expresivos el tratamiento de información imprecisa de forma natural, al tiempo que puedan ejecutarse de forma eficiente mediante, por ejemplo, el uso de mecanismos de evaluación perezosos. Para que ello sea factible, es necesario definir previamente una noción de igualdad apropiada que integre de forma natural todas estas características en un marco uniforme. En este trabajo, proponemos una potente noción de igualdad en este sentido, que es capaz de dar cuenta de propiedades perezosas y borrosas en un contexto declarativo integrado, y que además resulta facilmente implementable a un alto nivel de abstracción.
**Palabras clave:** Igualdad, Programación Declarativa, Lógica Difusa

## 1 Introducción

Debido a la vaguedad esencial del pensamiento humano, el tratamiento de la incertidumbre tiene una importancia creciente en inteligencia artificial e investigaciones relacionadas. Hoy en día, un considerable número de sistemas lógicos se han expresado como formalizaciones de conceptos vagos y razonamiento aproximado [20]. Mientras la lógica difusa es una amplia área de investigación donde argumentos teóricos y aplicaciones prácticas alusivos a estos sistemas son cuestiones de crucial interés, *la programación lógica* [14] por su parte, ha sido usada ampliamente en el pasado para resolución de problemas y representación del conocimiento. Sin embargo los lenguajes lógicos tradicionales no son capaces de tratar con la verdad parcial. La programación lógica difusa es un área de investigación interesante y todavía creciente que aúna los esfuerzos de la lógica difusa con los logros de la la programación lógica, para dotar a estos lenguajes tradicionales de herramientas expresivas que les permitan trabajar de forma natural con incertidumbre y razonamiento aproximado.

Por otra parte, después de más de tres décadas de investigación en el campo de la integración de paradigmas de programación declarativa (como el lógico,

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 600-612, 2006.

el funcional y el difuso), han sido muchas las contribuciones aportadas por la comunidad científica del área, que ha producido grandes avances tanto en los aspectos más teóricos como en los de aplicación más inmediata. Si bien es verdad que por un lado el paradigma lógico y el funcional, y por otro el lógico y el difuso, se han integrado con mayor o menor fortuna, no existen precedentes (a excepción de nuestra primera propuesta presentada en [18]) de una integración total de los tres marcos. En este trabajo damos una vuelta de tuerca más en este último sentido, centrándonos ahora en la fusión de las diferentes nociones de igualdad que cada uno de los tres estilos soporta por separado y teniendo en cuenta que esta noción tiene una importancia capital en el diseño del repertorio de recursos expresivos que todo lenguaje declarativo que se precie debe poseer.

Cuando en programación declarativa se utiliza el término "igualdad", son varias sus acepciones en función del paradigma considerado. A modo de resumen y sin ánimo de ser exhaustivos, tenemos las siguientes variedades:

- *Igualdad sintáctica [16, 11].* Es el modelo más simple de igualdad, usada en lenguajes lógicos puros como Prolog, que sólo considera iguales aquellos elementos que (tras un proceso de unificación) tienen exactamente la misma sintaxis y que por tanto, son totalmente idénticos. Por ejemplo, `f(a)` es sintácticamente igual a `f(a)` pero no a `g(b)`.
- *Igualdad semántica [3, 5].* Esta noción, también conocida como *E-igualdad* y que incluye a la anterior, aparece por primera vez en el contexto de los lenguajes lógico–funcionales como Curry donde, ante la presencia de reglas de reescritura que definen un conjunto de funciones matemáticas, dos llamadas a función diferentes pueden considerarse iguales si en algún momento de su evaluación convergen a valores sintacticamente idénticos.
- *Igualdad estricta [4, 19].* Esta variedad está muy relacionada con la anterior y se usa en lenguajes perezosos, bien sean funcionales puros (como Haskell) o bien lógico-funcionales (como Curry). En los contextos perezosos, ante la presencia de funciones no terminantes, se exige que la evaluación completa de dos funciones alcancen eventualmente valores constructores idénticos. Por ejemplo, si la evaluación de `f(a)` no termina, entonces no puede decirse que `f(a)` sea estrictamente igual a `f(a)`, a pesar de que ambas expresiones sean sintacticamente iguales. Y a la inversa, dos expresiones con distintas sintaxis, como por ejemplo `g(b)` y `h(c)` se consideran estrictamente iguales si al evaluarse generan un mismo valor constructor, como por ejemplo `0`.
- *Igualdad similar [21, 17].* Esta variante toma como base el concepto de similaridad tal y como es usado por el lenguaje lógico difuso Likelog para difuminar la clásica noción de "igualdad sintáctica" de Prolog. En este caso, y a modo de ejemplo, si en un determinado programa existen ecuaciones de similaridad entre los símbolos de función `f` y `g`, y las constantes `a` y `b`, entonces podría demostrarse que los objetos `f(a)` y `g(b)` son iguales con un determinado grado de similaridad (basado en las ecuaciones de similaridad).

En este trabajo presentamos una nueva versión de igualdad que combina las dos últimas variedades comentadas, para poder dar cuenta tanto de la noción

de pereza ("laziness") como de similaridad ("fuzziness") en un contexto lógico-difuso-funcional integrado. Lo mejor de nuestra aproximación es que es fácilmente implementable a un muy alto nivel de abstracción, con tan solo añadir automáticamente a un programa dado, un conjunto de reglas de reescritura que definen el símbolo de igualdad similar estricta ":=:" sobre símbolos constructores que se consideran similares. Más aún, esta acción se muestra complementaria a los desarrollos que presentamos en [18] para obtener una versión completamente difusa del lenguaje lógico-funcional Curry.

## 2 Programación lógico borrosa y similaridad

En las dos última décadas, se han desarrollado varios lenguajes de programación lógico difusos, donde básicamente el mecanismo de inferencia clásica de SLD-resolución ha sido reemplazado por alguna variante difusa del mismo, con el objetivo de poder tratar con verdad parcial y razonar con incertidumbre. La mayoría de estos lenguajes implementan el principio de resolución introducido por Lee[12], tales como Elf-Prolog [10], Frill [2], F-Prolog [13], Likelog [1] y el lenguaje basado en lógica multi-adjunta de [17]. En el presente trabajo estamos principalmente interesados en lenguajes como Likelog, fundamentado sobre la noción matemática de similaridad, ya que pensamos que admiten una extensión natural para tratar con funciones matemáticas y con un gran recurso computacional procedente del mundo de los lenguajes funcionales como es la pereza.

El concepto de "relacion de similaridad" es un ente matemático que proporciona una forma de manejar instancias alternativas de una entidad que pueden ser consideradas "iguales" con un cierto grado de verdad [23]. Las relaciones de similaridad están estrictamente relacionadas con relaciones de equivalencia y por tanto con los operadores de cierre. Comenzamos recordando que una T-norma $\wedge$ en $[0,1]$ es una operacion binaria: $\wedge : [0,1] \times [0,1] \rightarrow [0,1]$ la cual es asociativa, conmutativa, monótona creciente en ambas variables y con elemento neutro unidad. Para simplificar nuestros desarrollos, y de forma parecida a otras aproximaciones en lógica difusa [21] en lo que sigue supondremos que $x \wedge y$ es el mínimo entre los elementos $x, y \in [0,1]$.

**Definicion 1.** *Una relación de similaridad $\Re$ sobre un dominio $\mathcal{U}$ es un subconjunto difuso $\Re : \mathcal{U} \times \mathcal{U} \rightarrow [0,1]$ de $\mathcal{U} \times \mathcal{U}$ tal que $\forall x, y, z \in \mathcal{U}$, que cumple las siguientes propiedades: Reflexiva ($\Re(x,x) = 1$), Simétrica ($\Re(x,y) = \Re(y,x)$) y T-Transitiva ($\Re(x,z) \geq \Re(x,y) \wedge \Re(y,z)$).*

Una forma muy simple, pero efectiva, de introducir la noción de similaridad en programación lógica pura y que ha generando uno de los caminos de integración mas prometedores para el nuevo paradigma de la programación lógica difusa, consiste en modelar dicha noción mediante un conjunto de (las así llamadas) ecuaciones de similaridad de la forma $eq(s_1, s_2) = \alpha$, con el significado intencionado de que $s_1$ y $s_2$ son símbolos de predicados/constantes de la misma aridad con grado de similaridad $\alpha$. Esta aproximación se sigue por ejemplo en el lenguaje lógico difuso Likelog [1] donde un conjunto de cláusulas típicas de

Prolog se completa con otro conjunto de ecuaciones de similaridad que juegan un importante papel en tiempo de unificación difusa. Por supuesto, se asume que el conjunto de ecuaciones de similaridad debe ser seguro en el sentido de que cada ecuación conecta dos símbolos de la misma aridad, y que no se violan las propiedades de la Definición 1 (como ocurriría, por ejemplo, con el conjunto erróneo $\{eq(a,b) = 0.5, eq(b,a) = 0.9\}$, ya que estas ecuaciones, aparte de introducir riesgos de bucles infinitos cuando sean tratadas computacionalmente, en particular no verifican la propiedad de simetría).

*Ejemplo 1.* Dado el siguiente programa prolog compuesto por una simple cláusula con cuerpo vacío **p(a)**, la evaluación del objetivo **p(X)** proporciona la respuesta computada (substitución) $\{X \mapsto a\}$ con el significado de que, puesto que la constante **a** satisface el predicado **p** en el programa, entonces cuando la variable **X** sea substituida/enlazada a **a** en el objetivo, también se satisfará este. Sin embargo, si añadimos la ecuación de similaridad $eq(a,b) = 0.5$ al hecho anterior, obtendremos así el programa Likelog para el cual la evaluación del objetivo **p(X)** dará dos respuestas computadas (incorporando ahora a la correspondiente substitución al modo Prolog , su grado de verdad asociado) $\{X \mapsto a\}$ con grado de verdad 1 y $\{X \mapsto b\}$ con grado 0.5.

Es importante hacer notar que, ya que Likelog está orientado a la manipulación de bases de datos inductivas, donde no se permiten símbolos de función de aridad no nula, entonces las ecuaciones de similaridad sólo consideran congruencias entre dos predicados de la misma aridad o dos constantes (es decir, símbolos de función constructores sin parámetros). En este artículo superaremos esta limitación permitiendo también ecuaciones de similaridad entre cualquier pareja de símbolos de función constructoras no necesariamente constantes. Por otro lado, ya que a diferencia de Likelog, el lenguaje de sintaxis funcional que usaremos en este trabajo no dispone de símbolos propios de predicado, no será necesario introducir ecuaciones de similaridad entre este tipo de elementos. Este convenio no supone una limitación en la práctica, ya que usaremos la noción de "restriccion" en lugar de la de predicado para modelar sus efectos, siendo ésta una vía bastante estándar de simulación de predicados en lenguajes lógico funcionales.

Tal y como ya hicimos en [18], supondremos aquí que la relación de similaridad $\Re$ asociada a un programa dado $\mathcal{R}$, es la inducida desde el conjunto (seguro) de ecuaciones de similaridad de $\mathcal{R}$, verificando que el grado de similaridad de dos símbolos $s_1$ y $s_2$ es 1 si $s_1 \equiv s_2$ o por lo contrario este valor se obtiene recursivamente como un cierre transitivo del conjunto ecuacional, definido como: $T_r^t(R) = \bigcup_{f=1\cdots\infty} R^f$ donde $R^{f+1} = R^f \circ R$ para una T-Norma dada T. Este proceso es terminante y efectivo, pudiéndose demostrar que si el dominio es un conjunto finito de n elementos, entonces sólo es necesario calcular n-1 potencias. Finalmente, por simple suposición extendemos el conjunto resultante de ecuaciones por reflexividad [6, 22].

Mas recientemente hemos encontrado en [7] nuevas formas de sintetizar completamente la relación de similaridad $\Re$ asociada a un programa $\mathcal{R}$ desde el conjunto de ecuaciones germen o de partida. En particular se han propuesto dos métodos eficientes para obtener este objetivo: el cierre transitivo $\Re_c$, que es la

menor similaridad que contiene al conjunto de ecuaciones y la apertura transitiva $\Re_o$, que es la mayor similaridad contenida en el conjunto germen. Aunque no entraremos en detalles, el presente trabajo puede ser visto como una aplicación de tales métodos al diseño de lenguajes declarativos perezosos y difusos basados en relaciones de similaridad.

## 3 Reescritura, estrechamiento y pereza

Si en la sección anterior nos hemos dedicado a introducir el paradigma integrado de la programación lógico-difusa en torno al concepto de similaridad, en esta sección nos centramos en la otra gran vía de integración: la que origina la programación lógico-funcional e incorpora la noción de pereza. Como veremos, este recurso tiene importantes repercusiones a nivel de rendimiento computacional y, como nota motivadora, es reseñable el hecho de que tanto Likelog como cualquier otro lenguaje lógico son voraces, y por tanto no son capaces de realizar cómputos perezosos. En lo que sigue, proponemos el uso combinado de ecuaciones de similaridad (a la Likelog) junto con reglas de reescritura (en lugar de cláusulas de Horn) típicamente usadas por los lenguajes funcionales puros(Haskell) y los ya integrados lógico-funcionales (Curry).

La teoría de sistemas de reescritura de términos (SRT) ha sido ampliamente usada en programación declarativa para desarrollar lenguajes funcionales puros y lógico-funcionales integrados, tales com Haskell y Curry repectivamente. Un programa Haskell o Curry no es mas que un SRT, es decir, un conjunto de reglas de reescritura (en lugar de un conjunto de cláusulas como ocurre en los lenguajes lógicos). Las diferencias entre un programa funcional puro o lógico funcional no aparecen por tanto a nivel sintáctico, sino que emergen únicamente a nivel operacional, dependiendo de si usa reescritura o estrechamiento para ejecutarlos. A continuación introducimos una serie de convenciones notacionales habituales que nos serán útiles para definir ambos conceptos.

En lo que sigue, consideremos una signatura $\Sigma$ particionada en un conjunto $\mathcal{C}$ de constructores y un conjunto $\mathcal{F}$ de funciones definidas. El conjunto de términos constructores (con variables) se obtiene usando símbolos de $\mathcal{C}$ (y un conjunto de variables $\mathcal{X}$). El conjunto de variables que aparecen en un término $t$ se denota por $\mathcal{V}ar(t)$. Escribiremos $\overline{o_n}$ para la *lista* de objetos $o_1 \cdots, o_n$. Un *patrón* es un término de la forma $f(\overline{d_n})$ donde $f/n \in \mathcal{F}$ y $d_1, \cdots, d_n$ son términos constructores (con variables). Un término se dice lineal si no contiene múltiples ocurrencias de una misma variable. Una posición p en un término t está representada por una secuencia de números naturales ($\Lambda$ denota la secuencia vacía, es decir, la posición raíz). Las posiciones serán ordenadas por un orden prefijo: $p \leq q \, si \, \exists \omega \, tal \, que \, p \cdot \omega = q$. $t\,|_p$ denota el subtérmino de $t$ en un posición dada $p$ y $t\,[s]_p$ denota el resultado de reemplazar el subtérmino $t\,|_p$ por el término s. Denotaremos por $\{x_1 \mapsto t_1 \cdots x_n \mapsto t_n\}$ la substitución $\sigma$ con $\sigma(x_i) = t_i$ para i=1 ...n con $(x_i \neq x_j)$ si $i \neq j$ y $\sigma(x) = x$ para todas las demás variables $x$. La aplicación de la substitución $\sigma$ a un término t será denotada por $\sigma(t)$. Un conjunto de reglas de reescritura $l \to r$ tal que $l \notin \chi$ y $Var(r) \subseteq Var(l)$

se llama sistema de reescritura de términos (SRT). Los términos $l$ y $r$ son llamados término izquierdo y derecho de la regla, respectivamente. Un SRT $\mathcal{R}$ es lineal por la izquierda si las partes izquierdas de todas las regla son lineales. Un SRT se denomina basado en constructores (CB) si cada parte izquierda es un patrón. En el resto del artículo, un programa será siempre un SRT lineal por la izquierda, basado en constructores y sin solapamiento de reglas (esto es, las partes izquierdas de las reglas no unifican entre sí).

Un paso de reescritura (o reducción) consiste en la aplicación de una regla de reescritura a un término. Por tanto $t \rightarrow_{p,R} s$, si existe una posición $p$ en $t$, una regla de reescritura $R = (l \rightarrow r)$ y una substitución $\sigma$ tal que $t \mid_p = \sigma(l)$ y $s = t[\sigma(r)]_p$. La reescritura es el mecanismo operacional de los lenguajes funcionales puros, como Haskell. Sin embargo, la semántica operacional de los modernos lenguajes integrados lógico-funcionales tales como Curry, está usualmente basada en estrechamiento (necesario), que puede verse como el resultado de combinar instanciación de variables con reescritura. Formalmente, $t \leadsto_{p,R,\sigma} s$ es un paso de estrechamiento si $p$ es una posición no variable en $t$ y $\sigma(t) \rightarrow_{p,R} s$. Denotaremos por $t_p \leadsto^*_\sigma t_n$ a una secuencia de pasos de estrechamiento $t_1 \leadsto_{\sigma_1} \cdots \leadsto_{\sigma_n} t_n$ *con* $\sigma = \sigma_n \circ \cdots \circ \sigma_1$.

Con independencia de mecanismo de reescritura/estrechamiento usado para evaluar objetivos ante un programa dado, distinguiremos entre modelos de evaluación voraces (o de llamada por valor) y perezosos (o de llamada por nombre). Intuitivamente, cuando tenemos una expresión en la que varios subtérminos necesitan ser evaluados, una estrategia voraz intentará evaluar los subtérminos mas internos en cada paso de cómputo, mientras que por el contrario, una estrategia perezosa dará prioridad a los términos mas externos para ser procesados. Aunque una estrategia perezosa es siempre mas difícil de implementar que una voraz, los beneficios de la pereza han sido ampliamente difundidos y demostrados en la literatura especializada. Por ejemplo, en programas que definen estructuras de datos infinitas, los cálculos a efectuar sólo gozan de garantías de terminación cuando son realizados con estrategia perezosas.

En la siguiente sección nos centraremos en otro elemento importante en programación declarativa, que además está estrechamente relacionado con la semántica operacional de los lenguajes perezosos (ya bien sean funcionales puros, logico–funcionales o lógico-difuso-funcionales). Se trata de la llamada igualdad estricta, un punto crucial que, de paso, nos permitirá introducir relaciones de similaridad (a un bajo coste y a alto nivel de abstracción) sobre dichos lenguajes.

## 4 Igualdad similar estricta

Aunque en ocasiones resulta usual en programación logico-funcional simular los predicados típicos (nítidos) de la programación lógica pura por medio de funciones booleanas, una segunda y mas interesante alternativa de encarar este problema consiste en usar restricciones. Una restricción elemental es una restricción ecuacional $e_1 =:= e_2$ entre dos expresiones de un cierto tipo base. Así pues, $e_1 =:= e_2$ se satisface si ambos lados de la restricción son reducibles al mismo

término básico. Esta noción de igualdad, que por otra parte es la única posible en presencia de funciones no terminantes [8, 19], es también usada en lenguajes funcionales perezosos puros y se la conoce como igualdad estricta. En este tipo de construcciones, cuando uno de los miembros de la restricción está indefinido (porque su evaluación, por ejemplo, es no terminante), entonces la igualdad estricta no se cumple. De esta forma, para funciones que puedan no terminar, observamos el hecho de que la igualdad estricta no es reflexiva.

La restricciones ecuacionales se distinguen de las funciones booleanas estándar en el hecho de que las restriciones solo se evalúan para demostrar su satisfacibilidad. Por ejemplo, la restricción ecuacional $X =:= 0$ es satisfacible si la variable $X$ está cargada con 0. Por ello, la evaluación de $X =:= 0$ no devuelve un valor booleano true o false, ya que en el segundo caso, para devolver un valor falso, deberíamos cargar $X$ con todos los valores diferentes de 0. Esto es suficiente de forma semejante a la programación lógica pura, donde los predicados (o restricciones en nuestro caso), sólo son evaluados para ver que valores de sus variables los hacen ciertos (los satisfacen), sin importar las restantes situaciones de fallo.

Operacionalmente, una restricción ecuacional $e_1 =:= e_2$ se resuelve evaluando $e_1$ y $e_2$ a términos unificables. La restricción ecuacional también puede ser resuelta de forma incremental mediante evaluación perezosa entrelazada de las expresiones y la asignación de variables a términos constructores [15]. Las restricciones se pueden combinar mediante el operador de conjunción (ejecutable concurrentemente) escrito como $c_1 \& c_2$. Este mecanismo de evaluación puede ser implementado a muy alto nivel de abstracción suponiendo que cada programa incorpora el conjunto estandar de reglas de reescritura mostrado en la Figura 1, que define la semántica de los símbolos primitivos que modelan la igualdad estricta $=:=$ y $\&$ [9, 19]. Así, si $=:=$ representa la forma natural de usar la igualdad estricta sobre restricciones que simulan "predicados nítidos", nuestra siguiente tarea consiste en introducir un nuevo operador $:=:$, para modelar predicados difusos, por medio de las nuevas nociones de igualdad similar estricta y f-restricciones.

| | | | |
|---|---|---|---|
| "**StrEq**" | | *Igualdad estricta* | |
| $c$ | $=:=\ c$ | $\rightarrow$ `success` | $\forall c/0 \in \mathcal{C}$ |
| $c(x_1,..,x_n)$ | $=:=\ c(y_1,..,y_n)$ | $\rightarrow x_1 =:= y_1 \ \& \ldots \& \ x_n =:= y_n$ | $\forall c/n \in \mathcal{C}$ |
| `success` | `& success` | $\rightarrow$ `success` | |
| "**SimStrEq**" | | *Igualdad similar estricta* | |
| $c$ | $:=:\ d$ | $\rightarrow \Re(c,d)$ | $\forall c/0, d/0 \in \mathcal{C}$ |
| $c(x_1,..,x_n)$ | $:=:\ d(y_1,..,y_n)$ | $\rightarrow$ `min`$(\Re(c,d), x_1 :=: y_1, \ldots, x_n :=: y_n)$ | $\forall c/n, d/n \in \mathcal{C}$ |

**Fig. 1.** Reglas para $=:=$ (igualdad estricta) y $:=:$ (igualdad similar estricta)

Dado un conjunto de reglas de reescritura aumentado con un conjunto de ecuaciones de similaridad, y dada una f-restricción $e_1 :=: e_2$ sobre expresiones arbitrarias, nuestro objetivo ahora consiste en reducir ambas expresiones $e_1$ y $e_2$ a términos básicos $v_1$ y $v_2$, mediante el uso de las reglas de reescritura, y a partir de ahí, comparar los términos resultantes $v_1$ y $v_2$, teniendo en cuenta la relación de similaridad $\Re$ inducida por el conjunto de ecuaciones de similaridad que pertenecen al programa dado (ver Figura 1). Ahora en lugar de success, si la f-restricción es satisfacible, entonces debe devolver un número real en el intervalo [0,1] que represente el grado de similaridad entre las salidas $v_1$ y $v_2$.

Básicamente el conjunto de reglas de reescritura que definen :=: en la Figura 1 procede como sigue: el grado de similaridad entre dos símbolos constructores de aridad cero, es el que proporciona la relacion de similaridad $\Re$ inducida. Por otro lado cuando comparamos dos términos-dato (obtenidos después de reducir los parámetros originales de la f-restricción a sus formas constructoras) con argumentos, es necesario computar recursivamente el grado de similaridad entre cada par de argumentos de los términos, junto con la relación de similaridad entre los símbolos constructores que encabezan estos términos-datos. En la siguiente sección ilustramos con un ejemplo el comportamiento lógico-difuso-perezoso de este mecanismo.

## 5 Un ejemplo ilustrativo

En el siguiente programa $\mathcal{P}$, consideraremos que los términos-dato son construidos con constantes `a`, `b` y `c`, símbolos constructores (de aridad 1) `r` y `s`, y el constructor binario típico : para modelar listas. Por el contrario `f`, `g` y `h` son símbolos de función definidos por el siguiente conjunto de reglas de reescritura (nótese el uso de letras mayúsculas para denotar las variables):

$$\begin{array}{lllll} R_1: & \texttt{f(X)} & \rightarrow & \texttt{r(X)} : \texttt{f(X)} \\ R_2: & \texttt{g(b)} & \rightarrow & \texttt{s(c)} \\ R_3: & \texttt{h(X : Y)} & \rightarrow & \texttt{X} \end{array}$$

Intuitivamente, la funcion `f` genera una lista infinita de la forma `r(X)` : `r(X)` : `r(X)` : ... (introduciendo el riesgo de no terminación en procesos no-perezosos o voraces), la función `g` produce el término-dato `s(c)` cuando es llamada con argumento `b`, y finalmente la funcion `h` devuelve el (primer) elemento cabeza de una lista dada. Supongamos que $\mathcal{P}$ también contiene el siguiente conjunto de ecuaciones de similaridad entre símbolos constructores de la misma aridad: $E_1: \ \texttt{eq(a,b)} = 0.8, \quad E_2: \ \texttt{eq(b,c)} = 0.6, \quad E_3: \texttt{eq(r,s)} = 0.5.$

Para generar la relación de similaridad basada en el conjunto previo de ecuaciones de similaridad, es suficiente aplicar el algoritmo de "transitivización" de [7] para obtener su cierre transitivo, o apertura transitiva, según se desee. Suponemos que es el usuario quien decide una u otra opción, dependiendo de si quiere usar grados de similaridad bajos ("pesimista") o altos ("optimista"), al ejecutar el programa. De cualquier manera, debido a la simplicidad de nuestro conjunto de ecuaciones de similaridad actual, tal decisión no es importante en

este caso, pues $\Re_c = \Re_o$. La relación de similaridad resultante (a la que llamaremos genéricamente $\Re$) inducida a partir de $E_1, E_2,\ y\ E_3$ es:

$$\Re = \begin{pmatrix} & \mathtt{a} & \mathtt{b} & \mathtt{c} & \mathtt{r} & \mathtt{s} \\ \mathtt{a} & 1 & 0.8 & 0.6 & 0 & 0 \\ \mathtt{b} & 0.8 & 1 & 0.6 & 0 & 0 \\ \mathtt{c} & 0.6 & 0.6 & 1 & 0 & 0 \\ \mathtt{r} & 0 & 0 & 0 & 1 & 0.5 \\ \mathtt{s} & 0 & 0 & 0 & 0.5 & 1 \end{pmatrix}$$

A partir de esta matriz y siguiendo el método descrito en la sección anterior, construimos el siguiente conjunto "SimStrEq" de reglas de reescritura para :=:

$R_4 :\ \mathtt{a} :=: \mathtt{a} \to \mathtt{1}$ $\quad R_5 :\ \mathtt{a} :=: \mathtt{b} \to \mathtt{0.8}$ $\quad R_6 :\ \mathtt{a} :=: \mathtt{c} \to \mathtt{0.6}$
$R_7 :\ \mathtt{b} :=: \mathtt{a} \to \mathtt{0.8}$ $\quad R_8 :\ \mathtt{b} :=: \mathtt{b} \to \mathtt{1}$ $\quad R_9 :\ \mathtt{b} :=: \mathtt{c} \to \mathtt{0.6}$
$R_{10} : \mathtt{c} :=: \mathtt{a} \to \mathtt{0.6}$ $\quad R_{11} : \mathtt{c} :=: \mathtt{b} \to \mathtt{0.6}$ $\quad R_{12} : \mathtt{c} :=: \mathtt{c} \to \mathtt{1}$

$R_{13} : \mathtt{r(X)} :=: \mathtt{r(Y)} \to \mathtt{min}(\mathtt{1}, \mathtt{X} :=: \mathtt{Y})$ $\quad R_{14} : \mathtt{r(X)} :=: \mathtt{s(Y)} \to \mathtt{min}(\mathtt{0.5}, \mathtt{X} :=: \mathtt{Y})$
$R_{15} : \mathtt{s(X)} :=: \mathtt{r(Y)} \to \mathtt{min}(\mathtt{0.5}, \mathtt{X} :=: \mathtt{Y})$ $\quad R_{16} : \mathtt{s(X)} :=: \mathtt{s(Y)} \to \mathtt{min}(\mathtt{1}, \mathtt{X} :=: \mathtt{Y})$

Ahora podemos de forma segura reemplazar el conjunto original de ecuaciones $E_1, E_2,\ Y\ E_3$ por la colección anterior de reglas de reescritura (que recoge toda la información sobre las similaridades entre símbolos constructores), las cuales son añadidas al programa original $\mathcal{P}$ para obtener el programa extendido $\mathcal{P}^+ = \mathcal{P} - \{E_1, E_2, E_3\} \cup SimStrEq = \{R_1, R_2, R_3\} \cup \{R_4, \cdots, R_{16}\} = \{R_1, \cdots R_{16}\}$. Es importante notar el hecho de que tanto $\mathcal{P}$ como $\mathcal{P}^+$ reflejan la misma información sobre funciones y similaridades, pero mientras $\mathcal{P}$ contiene reglas de reescritura y ecuaciones de similaridad, $\mathcal{P}^+$ sólo contiene reglas de reescritura, y por tanto, ya puede ser ejecutado directamente por reescritura/estrechamiento para resolver los objetivos, tal y como se muestra en la Figura 2 que detallamos a continuación.

– `Paso 1`. Es un paso de evaluación funcional puro, ya que aplica un paso de reescritura, usando la regla $R_1$, al único redex del objetivo original, es decir,

$$\begin{array}{rcl} \mathtt{h}(\underline{\mathtt{f(a)}}) :=: \mathtt{g(X)} & \leadsto^{1.1,\ R_1} & \underline{\mathtt{h(r(a) : f(a))}} :=: \mathtt{g(X)} \\ & \leadsto^{1,\ R_3} & \mathtt{r(a)} :=: \underline{\mathtt{g(X)}} \\ & \leadsto^{2,\ R_2}_{\{\mathtt{X} \mapsto \mathtt{b}\}} & \underline{\mathtt{r(a)}} \ \underline{:=:}\ \underline{\mathtt{s(c)}} \\ & \leadsto^{\Lambda,\ R_{14}} & \mathtt{min}(\mathtt{0.5}, \underline{\mathtt{a}} \ \underline{:=:}\ \underline{\mathtt{c}}) \\ & \leadsto^{2,\ R_6} & \mathtt{min}(\ \mathtt{0.5},\ \mathtt{0.6}) \\ & \leadsto & \mathtt{0.5} \end{array}$$

**Fig. 2.** Derivación para el objetivo $\mathtt{h(f(a))} :=: \mathtt{g(X)}$ en al programa extendido $\mathcal{P}^+$

el subtérmino subrayado localizado en la posición 1.1 (obsérvese que `f(a)` es el primer argumento de `h(f(a))`, el cual de nuevo es el primer argumento de la expresión total `h(f(a))` :=: `g(X)`).

- `Paso 2`. Este paso está también basado en reescritura (usando esta vez la regla $R_3$) pero en contraste con el anterior, ahora se puede observar el comportamiento perezoso de nuestro principio operacional. Precisamente el redex explotado (el término `h(r(a) : f(a))` en la posicion 1), también contiene un redex más interno (`f(a)`) que una estrategia voraz habría explotado infinitamente produciendo una evaluación no terminante con la regla $R_1$ (recuérdese que esta regla de reescritura define la generación de una lista infinita). Afortunadamnente nuestra estrategia perezosa evita este riesgo de entrar en bucle de un modo sencillo e inteligente: simplemente dando más prioridad a los redexes más externos frente a los internos al aplicar los pasos de derivación.
- `Paso 3`. Esta es la primera vez que realizamos un paso de estrechamiento propiamente dicho, ya que no esta basado en reescritura únicamente, sino también en instanciación de variables. Este paso introduce pues una dimensión lógica (por tanto, no solamente funcional) en la derivación. De hecho, antes de reducir el subtérmino `g(X)` en la posición 2 del objetivo original, el estrechamiento primero genera la sustitución $\texttt{X} \mapsto \texttt{b}$ y la aplica, para posteriormente reducir el redex resultante `g(b)` con la regla $R_2$.
- `Paso 4`. Y si en los tres pasos anteriores hemos evidenciado las características funcionales, perezosas y lógicas, respectivamente, de nuestra aproximación, en lo que resta focalizaremos nuestra atención sobre sus propiedades difusas. El cuarto paso explota la similaridad entre los constructores `r` y `s` simplemente realizando un paso de rescritura sobre la totalidad del objetivo (posición $\Lambda$) con la regla recien sintetizada $R_{14}$. Recuérdese que esta regla de reescritura de "SimStrEq" no estaba presente en el programa original $\mathcal{P}$ sino únicamente en el programa extendido $\mathcal{P}^+$. Esta regla recoge los grados de similaridad 0.5 entre `r` y `s` expresados en la ecuación de similaridad $\texttt{E}_3$ de $\mathcal{P}$. Obsérvese también en la expresión resultante la llamada recursiva a :=: con los parámetros actuales de los subtérminos `r(a)` y `s(c)`, que serán considerados en el siguiente paso.
- `Paso 5`. Otra vez estamos describiendo un paso difuso, el cual explota la similaridad de grado 0.6 entre las constantes `a` y `c` por medio de una regla de reescritura $R_6 \in \mathcal{P}^+$. Es importante resaltar que aunque no hay ecuación de similaridad en el programa original $\mathcal{P}$ relacionando directamente ambas constates, el correspondiente grado de similaridad es obtenido por transitividad a partir de $\texttt{E}_1$ y $\texttt{E}_2$ después de aplicar los metodos de [7].
- `Paso 6`. El último paso de derivación simplemente evalúa la T-norma del mínimo (con grados de similaridad concretos), que puede ser considerada como un operador primitivo del lenguaje.

Finalmente, la respuesta computada para el objetivo original puede ser interpretada como que "las expresiones `h(f(a))` y `g(X)` pueden ser reducidas a valores similares con grado de creencia 0.5 cuando la variable `X` se carga con el valor `b`".

Para finalizar, es importante remarcar dos interesantes propiedades que disfruta nuestra técnica basada en la relajación de las nociones de igualdad similar e igualdad estricta por la más flexible de igualdad similar estricta:

- El método representa una forma simple, de bajo coste, para difuminar lenguajes lógico–funcionales y puede ser implementado a un alto nivel de abstracción. Para ello es suficiente con realizar un preproceso puramente sintáctico, que genera reglas de reescritura a partir de ecuaciones de similaridad sin alterar el mecanismo operacional (reescritura/estrechamiento) de base.
- Además, es bastante eficaz: las relaciones de similaridad entre símbolos constructores de cualquier aridad (recuérdese que otros lenguajes lógico difusos voraces tales como Likelog sólo tratan con constantes) son apropiadamente explotadas al comparar términos-dato obtenidos como salida de expresiones más complejas.

## 6 Conclusiones y trabajo futuro

El desarrollo de sistemas informáticos con tratamiento de información imprecisa es un área de gran actualidad que ya ha dado frutos significativos implantados a nivel industrial. La programación declarativa, en sus distintos estilos, ha jugado un papel importante en el diseño de tales aplicaciones. Conjugar lo mejor de estos estilos, como son el tratamiento de la igualdad, la pereza, la incertidumbre, etc... resulta una tarea compleja a la que hemos intentado aproximarnos en este artículo. Más concretamente, en este trabajo hemos introducido una forma de combinar pereza y borrosidad en un marco de programación lógico–difuso–funcional integrado. El resultado ha sido un nuevo modelo de igualdad declarativa que aúna versiones anteriores de igualdad similar y estricta. Hemos mostrado que esta alternativa puede ser practicada a muy alto nivel de abstracción (sin manipular la semántica y el principio operacional del lenguaje original) con muy poco esfuerzo de implementación y con tal solo realizar un simple preprocesado puramente sintáctico. Para el futuro, nos planteamos probar propiedades formales este modelo, como por ejemplo, su corrección y completitud.

## Agradecimientos

Este trabajo ha sido financiado parcialmente por la UE (FEDER) y el Ministerio de Educación y Ciencia, a través del proyecto TIN 2004-07943-C04-03.

## References

1. F. Arcelli and F. Formato. Likelog: A logic programming language for flexible data retrieval. In *Proc. of the ACM Symposium on Applied Computing (SAC'99)*, pages 260–267. ACM, Artificial Intelligence and Computational Logic, 1999.
2. J. F. Baldwin, T. P. Martin, and B. W. Pilsworth. *Fril- Fuzzy and Evidential Reasoning in Artificial Intelligence.* John Wiley &; Sons, Inc., 1995.

3. P. Bosco, E. Giovannetti, and C. Moiso. Refined Strategies for semantic unification. In H. Ehrig, R. Kowalski, G. Levi, and U. Montanari, editors, *Proc. of TAPSOFT'87*, pages 276–290. Springer LNCS 250, 1987.
4. H.J. Bürckert. Lazy E-unification - a method to delay alternative solutions. Internal Report 87 R 34, Université de Nancy, France, 1987.
5. J.H. Gallier and S. Raatz. Extending SLD-resolution to equational Horn clauses using E-unification. *Journal of Logic Programming*, 6:3–43, 1989.
6. L. Garmendia and A. Salvador. Comparing transitive closure with other new T-transivization methods. In *Proc. Modeling Decisions for Artificial Intelligence*, pages 306–315. Springer LNAI 3131, 2004.
7. Luis Garmendia. An algorithm to compute the transitive closure, a transitive opening and a transitive approximation of a fuzzy proximity generating a binary partition tree. In *Proceedings to appear*, page 20. Publisher, 2005.
8. E. Giovannetti, G. Levi, C. Moiso, and C. Palamidessi. Kernel Leaf: A Logic plus Functional Language. *Journal of Computer and System Sciences*, 42:363–377, 1991.
9. M. Hanus (ed.). Curry: An Integrated Functional Logic Language. Available at `http://www.informatik.uni-kiel.de/~mh/curry/`, 2003.
10. M. Ishizuka and N. Kanai. Prolog-ELF Incorporating Fuzzy Logic. In Aravind K. Joshi, editor, *Proceedings of the 9th International Joint Conference on Artificial Intelligence (IJCAI'85). Los Angeles, CA, August 1985.*, pages 701–703. Morgan Kaufmann, 1985.
11. J.-L. Lassez, M. J. Maher, and K. Marriott. Unification Revisited. In J. Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pages 587–625. Morgan Kaufmann, Los Altos, Ca., 1988.
12. R.C.T. Lee. Fuzzy Logic and the Resolution Principle. *Journal of the ACM*, 19(1):119–129, 1972.
13. Deyi Li and Dongbo Liu. *A fuzzy Prolog database system.* John Wiley & Sons, Inc., 1990.
14. J.W. Lloyd. *Foundations of Logic Programming.* Springer-Verlag, Berlin, 1987. Second edition.
15. R. Loogen, F. López-Fraguas, and M. Rodríguez-Artalejo. A Demand Driven Computation Strategy for Lazy Narrowing. In J. Penjam and M. Bruynooghe, editors, *Proc. of PLILP'93, Tallinn (Estonia)*, pages 184–200. Springer LNCS 714, 1993.
16. A. Martelli and U. Montanari. An Efficient Unification Algorithm. *ACM Transactions on Programming Languages and Systems*, 4:258–282, 1982.
17. J. Medina, M. Ojeda-Aciego, and P. Vojtáš. Similarity-based unification: a multi-adjoint approach. *Fuzzy Sets and Systems*, 146(1):43–62, 2004.
18. G. Moreno and V. Pascual. Programming with Fuzzy–logic and Mathematical Functions. In I. Bloch, A. Petrosino, and A. Tettamanzi, editors, *Proc. of the 6th. International Whorshop on Fuzzy Logic and Applications, WILF'05, University of Milan, Crema (Italy)*, pages 89–98. Springer LNCS 3849, 2006.
19. J.J. Moreno-Navarro and M. Rodríguez-Artalejo. Logic Programming with Functions and Predicates: The language Babel. *Journal of Logic Programming*, 12(3):191–224, 1992.
20. H.T. Nguyen and E.A. Walker. *A First Course in Fuzzy Logic.* Chapman & Hall/CRC, Boca Ratón, Florida, 2000.
21. M.I. Sessa. Approximate reasoning by similarity-based SLD resolution. *Fuzzy Sets and Systems*, 275:389–426, 2002.

22. M. Wallace and S. Kollias. Computationally Efficient Incremental Transitive Closure of Sparse Fuzzy Binary Relations. In *Proc. of Fourth IEEE Int'l Symp. on Logic Programming*, pages 253–263. IEEE, New York, 2004.
23. L. A. Zadeh. Similarity relations and fuzzy orderings. *Informa. Sci.*, 3:177–200, 1971.

# BayesChess: Programa de ajedrez adaptativo basado en redes bayesianas

Antonio Fernández Álvarez y Antonio Salmerón Cerdán

Depto. Estadística y Matemática Aplicada
Universidad de Almería
afa109@alboran.ual.es, Antonio.Salmeron@ual.es

**Resumen** En este trabajo presentamos un programa de ajedrez capaz de adaptar su estrategia al usuario al que se enfrenta y de refinar la función de evaluación que guía el proceso de búsqueda en base a su propia experiencia de juego. La capacidad adaptativa y de aprendizaje se ha implementado mediante redes bayesianas. Mostramos el proceso de aprendizaje del programa mediante una experimentación consistente en una serie de partidas que evidencian una mejora en los resultados después de la fase de aprendizaje.

## 1. Introducción

Las redes bayesianas se han mostrado en los últimos años como una herramienta adecuada para la modelización de situaciones en las que interviene un gran número de variables y existe incertidumbre asociada al valor de las mismas en un momento dado [5,7]. Uno de los problemas en los que está cobrando especial importancia el uso de redes bayesianas es en la *clasificación* o reconocimiento de patrones, que consiste en predecir la clase a la que pertenece un objeto dado que se conoce el valor de algunos atributos del mismo. El problema de la clasificación está presente en la construcción de sistemas que se adapten al usuario, desde el momento en que el sistema tiene que determinar de qué tipo de usuario se trata y actuar en consecuencia.

En este trabajo describimos un programa de ajedrez capaz de adaptarse al usuario y ajustar su estrategia de juego según el estilo del usuario, siendo además capaz de aprender de su propia experiencia en el sentido de refinar la función de evaluación o heurística de búsqueda que se emplea para explorar el árbol de jugadas. Esta funcionalidad adaptativa y de aprendizaje automático se ha implementado usando redes bayesianas. Más concretamente, hemos empleado redes bayesianas orientadas a la clasificación, basándonos en la estructura *Naive Bayes*, especialmente adecuada en problemas en los que interviene un gran número de variables y la base de datos de aprendizaje es de tamaño limitado [6]. Hemos adoptado el nombre BayesChess por ser su particularidad el empleo de redes bayesianas.

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 613-624, 2006.

El objetivo no es conseguir un programa de ajedrez realmente competitivo con programas de la talla de Deep Blue [9] o Fritz [8], sino comprobar la viabilidad de construir sistemas adaptativos empleando redes bayesianas. Hemos elegido el ajedrez por considerar que éste presenta una serie de características que hacen apropiado el empleo de sistemas adaptativos:

- El programa interactúa constantemente con el usuario y ha de responder a las acciones del mismo.
- La imposibilidad de calcular todas las jugadas posibles hacen necesario el uso de una heurística.
- La validez de la heurística empleada puede contrastarse en base a los resultados obtenidos.
- Existen diferentes estilos de juego o estrategias que tanto el usuario como el programa pueden adoptar.

Por tanto, el objetivo de este trabajo es el uso de redes bayesianas para dotar de adaptatividad a un programa de ajedrez. En ese sentido, nos hemos concentrado en lo siguiente:

1. Refinamiento de la heurística de búsqueda, en base a la experiencia de juego del programa, mediante una red bayesiana.
2. Uso de una red bayesiana para clasificar el comportamiento del usuario y adoptar una estrategia en función del mismo.

El resto del trabajo se organiza como sigue: en la sección 2 revisamos algunos conceptos de redes bayesianas y clasificación. Continuamos exponiendo el diseño del motor de juego y la heurística de búsqueda en la sección 3. La actualización automática de dicha heurística se explica en la sección 4, mientras que la adaptación al comportamiento del usuario es el objeto de la sección 5. La experimentación llevada a cabo para evaluar el proceso de aprendizaje se describe en la sección 6, y el trabajo finaliza con las conclusiones en la sección 7.

## 2. Redes bayesianas y clasificación

Consideremos un problema caracterizado por un conjunto de variables $\mathbf{X} = \{X_1, \ldots, X_n\}$. Una red bayesiana [7,12] es un grafo dirigido acíclico donde cada nodo representa una variable del problema, y tiene asignada una distribución de probabilidad condicionada a sus padres. La presencia de un arco entre dos variables expresa la existencia de dependencia entre ambas, cuantificada por la distribución de probabilidad asignada a los nodos. En términos computacionales, una importante propiedad de las redes bayesianas es que la distribución conjunta sobre todas las variables de la red se factoriza de acuerdo con el concepto de *d*-separación [12] como sigue:

$$p(x_1, \ldots, x_n) = \prod_{i=1}^{n} p(x_i | pa(x_i)) \ , \tag{1}$$

donde $Pa(X_i)$ denota el conjunto de padres de la variable $X_i$ y $pa(x_i)$ una configuración concreta de valores de los mismos. Esta factorización implica que se puede especificar la distribución conjunta sobre todas las variables de la red con un importante ahorro de espacio. Por ejemplo, la distribución conjunta sobre las variables de la red de la figura 1, suponiendo que todas las variables son binarias, requeriría almacenar $2^5 - 1 = 31$ valores, mientras que haciendo uso de la factorización se puede representar la misma información usando sólo 11 valores (ver cuadro 1).

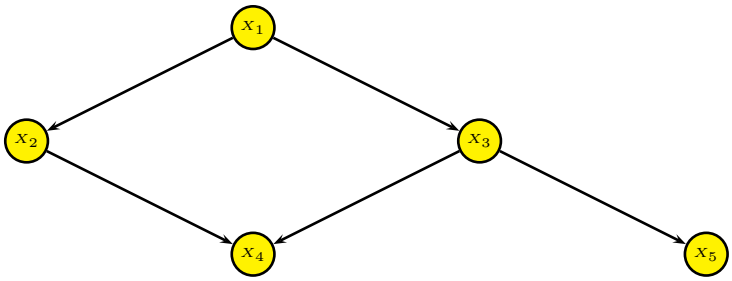



**Figura 1.** Ejemplo de red Bayesiana

**Cuadro 1.** Ejemplo de distribución factorizada para la red de la figura 1

| | |
|---|---|
| $p(X_1 = 0) = 0,20$ | $p(X_2 = 0\|X_1 = 0) = 0,80$ |
| $p(X_2 = 0\|X_1 = 1) = 0,80$ | $p(X_3 = 0\|X_1 = 0) = 0,20$ |
| $p(X_3 = 0\|X_1 = 1) = 0,05$ | $p(X_4 = 0\|X_2 = 0, X_3 = 0) = 0,80$ |
| $p(X_4 = 0\|X_2 = 1, X_3 = 0) = 0,80$ | $p(X_4 = 0\|X_2 = 0, X_3 = 1) = 0,80$ |
| $p(X_4 = 0\|X_2 = 1, X_3 = 1) = 0,05$ | $p(X_5 = 0\|X_3 = 0) = 0,80$ |
| $p(X_5 = 0\|X_3 = 1) = 0,60$ | |

Una red bayesiana puede usarse como clasificador si está compuesta por una variable clase, $C$, y un conjunto de variables predictoras $X_1, \ldots, X_n$, de forma que un individuo con características observadas $x_1, \ldots, x_n$ será clasificado como perteneciente a la clase $c^*$ obtenida como

$$c^* = \arg \max_{c \in \Omega_C} p(c|x_1, \ldots, x_n) \ , \tag{2}$$

donde $\Omega_C$ denota el conjunto de posibles valores de la variable clase $C$.

Obsérvese que $p(c|x_1, \ldots, x_n)$ es proporcional a $p(c) \times p(x_1, \ldots, x_n|c)$, y por tanto, resolver el problema de clasificación requeriría especificar una distribución sobre las $n$ variables predictoras para cada valor de la clase. El coste asociado

puede ser muy elevado. Sin embargo, usando la factorización determinada por la red este coste se reduce. El caso extremo es el clasificador conocido como *Naive Bayes* (ver, por ejemplo [3]), que supone que todas las variables predictoras son independientes entre sí dada la clase. Esto se traduce en una estructura como la representada en la figura 2.

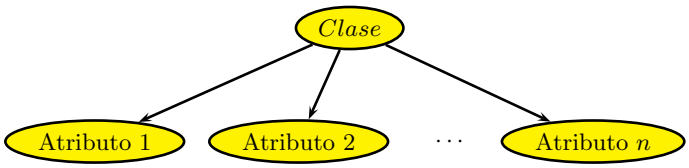



**Figura 2.** Topología de un clasificador Naïve Bayes

La fuerte suposición de independencia que hace este modelo se compensa con el reducido número de parámetros que hay que estimar, dado que en este caso se verifica que

$$p(c|x_1, \ldots, x_n) = p(c) \prod_{i=1}^{n} p(x_i|c) \ . \tag{3}$$

## 3. Diseño del motor de juego de ajedrez

El juego del ajedrez ha sido estudiado en profundidad por la Inteligencia Artificial, dentro de los llamados *juegos de información completa* (ver, por ejemplo, [11]). En este trabajo hemos considerado un motor de juego basado en el algoritmo mini-max con poda alfa-beta. Existen refinamientos en los algoritmos de búsqueda orientados a ajedrez, pero quedan fuera del ámbito de este trabajo. Sí es relevante la heurística utilizada para guiar el proceso de búsqueda pues, como describiremos más adelante, será actualizada mediante una red bayesiana aprendida en base a la experiencia de juego del programa.

La heurística que hemos considerado se basa en dos aspectos: material (piezas de cada jugador en el tablero) y situación de cada pieza (dependiendo de la casilla que ocupen en un momento dado, las piezas pueden ser más o menos valiosas). De forma adicional, hemos dado importancia también al hecho de dar jaque al rey adversario.

La evaluación del material se realiza asignando una puntuación a cada pieza. Hemos elegido la puntuación habitual en programas de ajedrez, mostrada en el cuadro 2. El rey no tiene puntuación, pues no es necesario al estar prohibida la captura del mismo.

En cuanto a la valoración de la posición de cada pieza, hemos empleado una matriz de $8 \times 8$ para cada ficha, de forma que cada celda contiene la puntuación

**Cuadro 2.** Puntuación de las piezas en la heurística empleada

| Pieza | Peón | Alfil | Caballo | Torre | Dama |
|---|---|---|---|---|---|
| Puntuación | 100 | 300 | 300 | 500 | 900 |

añadida al valor de la heurística en caso de que la pieza esté situada en ella. En el cuadro 3 puede verse un ejemplo de estas matrices, para el caso del caballo blanco. Nótese como se favorece la colocación del caballo en casillas centrales, donde tiene mayor capacidad de acción.

**Cuadro 3.** Pesos asociados a la posición del caballo blanco

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| -10 | -10 | -10 | -10 | -10 | -10 | -10 | -10 |
| -10 | 0 | 0 | 0 | 0 | 0 | 0 | -10 |
| -10 | 0 | 5 | 5 | 5 | 5 | 0 | -10 |
| -10 | 0 | 5 | 10 | 10 | 5 | 0 | -10 |
| -10 | 0 | 5 | 10 | 10 | 5 | 0 | -10 |
| -10 | 0 | 5 | 5 | 5 | 5 | 0 | -10 |
| -10 | 0 | 0 | 0 | 0 | 0 | 0 | -10 |
| -10 | -30 | -10 | -10 | -10 | -10 | -30 | -10 |

En total, la función heurística está definida por 838 parámetros ajustables, que son el valor de cada pieza, el valor de dar jaque y el número almacenado en cada celda de cada una de las matrices $8 \times 8$ definidas anteriormente.

## 4. Actualización automática de la heurística

En esta sección describimos el proceso de aprendizaje, o más precisamente de refinamiento de los parámetros de la heurística descrita en la sección 3. Con el auge de las técnicas de aprendizaje automático en los años 80, se consideró la aplicación de las mismas al ajedrez por computador, concluyéndose que sólo serían aplicables de forma marginal, como vías de extracción de patrones de libros de aperturas [13]. Sin embargo, más adelante surgieron aplicaciones de técnicas de clasificación, principalmente para la evaluación de posiciones de la fase final del juego [4].

Para el ajuste de los parámetros de la heurística, hemos considerado una red bayesiana con estructura tipo Naive Bayes con la salvedad de que en lugar de una variable clase hay dos: la *fase actual de la partida* (apertura, medio juego o final) y el *resultado de la partida* (ganar, perder, tablas). Como variables predictoras, se han empleado todos los parametros ajustables de la heurística descrita en la sección 3, lo que significa que la red cuenta con un total de 776 variables con la estructura mostrada en la figura 3. El elevado número de variables viene dado principalmente porque hay una variable por cada una de las 64 posibles ubicaciones de cada una de las piezas en el tablero. La razón por la que se ha usado una estructura de red tipo Naive Bayes es precisamente el alto número

de variables, ya que el uso de una estructura más compleja aumentaría drásticamente el tiempo necesario para evaluarla, lo que ralentizaría la evaluación de las posiciones durante la exploración del árbol de búsqueda.

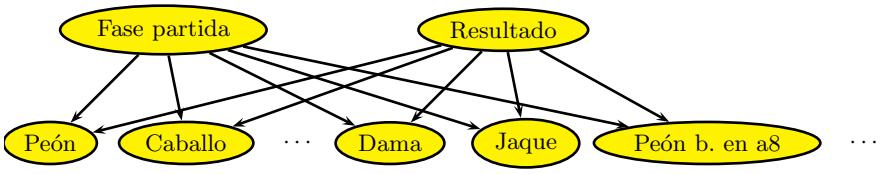



**Figura 3.** Estructura de la red bayesiana para el aprendizaje automático de la heurística

Los parámetros de la red bayesiana se estiman inicialmente a partir de una base de datos generada enfrentando a BayesChess contra él mismo, usando uno de los dos bandos la heurística tal y como se ha descrito anteriormente, y el otro una versión perturbada aleatoriamente, donde el valor de cada variable se incrementaba o decrementaba en un 20 %, 40 % o se mantenía a su valor inicial de forma aleatoria. En el cuadro 4 se puede ver el formato de dicha base de datos con unos casos de ejemplo. Vemos cómo para cada etapa de cada partida se ha generado una configuración de parámetros aleatorios que serán los que utilizará la heurística en la etapa concreta de la partida en juego. Al final de cada caso aparece el resultado de la misma. Evidentemente el resultado en cada *caso* de la base de datos perteneciente a una misma partida es el mismo, de ahí que aparezca repetido de forma consecutiva.

**Cuadro 4.** Ejemplo de casos de la base de datos de partidas

| Fase partida | Peon | Caballo | Alfil | Torre | Dama | Jaque | Peón a8 | Peon b8 | ... | Resultado |
|---|---|---|---|---|---|---|---|---|---|---|
| APERTURA | 120 | 180 | 240 | 700 | 540 | 42 | -6 | 0 | ... | PIERDEN |
| MEDIO | 120 | 360 | 360 | 600 | 1260 | 36 | 3 | 0 | ... | PIERDEN |
| FINAL | 120 | 420 | 180 | 400 | 720 | 42 | -3 | 3 | ... | PIERDEN |
| APERTURA | 140 | 360 | 420 | 300 | 1080 | 18 | 0 | 6 | ... | GANAN |
| MEDIO | 100 | 300 | 360 | 500 | 1260 | 42 | 3 | 0 | ... | GANAN |
| FINAL | 80 | 180 | 240 | 400 | 900 | 42 | 6 | 6 | ... | GANAN |
| APERTURA | 120 | 420 | 420 | 400 | 720 | 18 | -6 | 3 | ... | TABLAS |
| MEDIO | 140 | 300 | 360 | 500 | 1260 | 42 | 3 | 0 | ... | TABLAS |
| FINAL | 120 | 420 | 180 | 600 | 900 | 30 | 0 | 6 | ... | TABLAS |

Cada una de las tablas de probabilidad en esta red bayesiana requiere la estimación de 45 valores, ya que para cada uno de los 5 posibles valores de una variable habrá que diferenciar la fase de la partida y el resultado conseguido. En el cuadro 5 podemos observar un ejemplo de la tabla de probabilidad condicio-

nada de la variable *Peón*. Se han usado las abreviaturas A, M y F para las fases de partida y G, P y T para el resultado.

**Cuadro 5.** Ejemplo de tabla de probabilidad de la variable *Peón*

| | FASE PARTIDA | A | A | A | M | M | M | F | F | F |
|---|---|---|---|---|---|---|---|---|---|---|
| PEÓN | RESULTADO | G | P | T | G | P | T | G | P | T |
| | 60 | 0,2 | 0,1 | 0,3 | 0,3 | 0,2 | 0,3 | 0,2 | 0,3 | 0,2 |
| | 80 | 0,3 | 0,1 | 0,1 | 0,1 | 0,2 | 0,1 | 0,2 | 0,1 | 0,2 |
| | 100 | 0,1 | 0,1 | 0,2 | 0,4 | 0,2 | 0,1 | 0,1 | 0,1 | 0,2 |
| | 120 | 0,1 | 0,2 | 0,4 | 0,1 | 0,3 | 0,1 | 0,3 | 0,2 | 0,3 |
| | 140 | 0,3 | 0,5 | 0,1 | 0,2 | 0,1 | 0,4 | 0,2 | 0,3 | 0,1 |

El proceso de aprendizaje del juego no tiene límite, ya que dependerá del número de partidas existente en la base de datos. Por tanto, cuantas más partidas se hayan jugado más se refinarán los parámetros de la heurística, y por tanto mejor será el juego de la máquina, como veremos en la sección 6. Una vez concluido el entrenamiento inicial, BayesChess puede adoptar la heurística aprendida y a partir de ahí refinarla con nuevas partidas, ya contra oponentes humanos.

Una vez construida la red bayesiana, BayesChess la utiliza para elegir los parámetros de la heurística. El proceso de selección es consiste en instanciar las dos variables clase (fase de partida y resultado) y a partir de ahí se obtiene la configuración de parámetros que maximiza la probabilidad de los valores instanciados de las variables *Fase de partida* y *Resultado*. Para poder conocer siempre en qué fase de partida se encuentra el juego, hemos considerado que la apertura la forman las 10 primeras jugadas y el final de juego es cuando no hay damas o el número de piezas es inferior a 10. En otro caso, entendemos que la partida está en fase de medio juego. En cuanto a la otra variable a instanciar (resultado) obviamente no se conoce en el momento del juego, pero se puede usar para determinar el estilo de juego de BayesChess. Por ejemplo, si instanciamos la variable resultado a *ganar*, elegirá la configuración de parámetros que maximizan la probabilidad de ganar, aunque ésta sea menor que la suma de las probabilidades de perder y hacer tablas. Esto puede ser equivalente a considerar que BayesChess adopta una estrategia agresiva. Por contra, puede optarse por minimizar la probabilidad de perder, o lo que es lo mismo, maximizar la de ganar o hacer tablas. Esto puede derivar en una estrategia de juego más conservadora. La forma de elegir estas configuraciones es mediante inferencia abductiva [2,10]. En el caso concreto de la red bayesiana empleada por BayesChess, el cálculo es sencillo, pues la configuración que maximiza la probabilidad de una instanciación dada, es la que se obtiene de maximizar cada tabla de probabilidad por separado para dicha instanciación, debido a la factorización expresada en la ecuación (3).

## 5. Adaptación a la situación del oponente

En esta sección se describe cómo hacer que la heurística aprendida se adapte, en cada momento, a la estrategia de juego del usuario al que se enfrenta. Para

ello hemos considerado tres posibles estrategias que puede adoptar el usuario: atacante, posicional y mixta.

Hemos implementado un clasificador Naïve Bayes para determinar la estrategia de juego de un usuario tomando como referencia una serie de características que serán las variables de dicho clasificador. Dichas características son: el primer movimiento, la situación de los enroques (opuestos o no), número de piezas más allá de la tercera fila (excepto peones) y número de peones avanzados sobre la posición del rey. La estructura del clasificador puede verse en la figura 4.

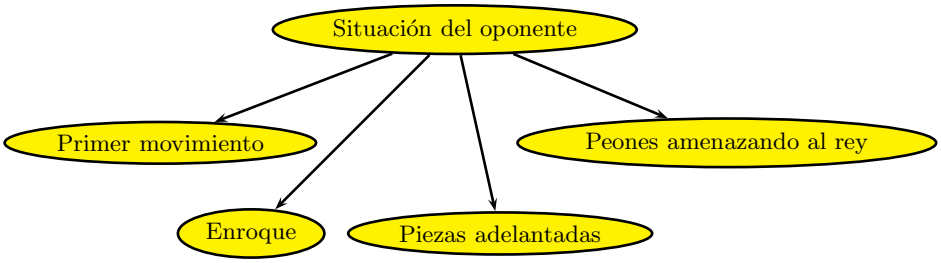



**Figura 4.** Estructura del clasificador de la estrategia del oponente

El entrenamiento del clasificador se ha realizado a partir de una base de datos de partidas de cuatro destacados jugadores profesionales conocidos por corresponderse con los tres estilos citados. En esas partidas, se han medido las variables anteriores y los valores obtenidos se han incluido en la base de datos de entrenamiento. En concreto, hemos seleccionado 2708 partidas de Robert Fischer y Gary Kasparov como modelo de juego atacante, 3078 de Anatoli Karpov como modelo de juego posicional o defensivo y, por último, 649 de Miguel Illescas como modelo de juego mixto. En el cuadro 6 se puede ver el formato de dicha base de datos con unos casos de ejemplo.

**Cuadro 6.** Ejemplo de casos de la base de datos de entrenamiento del clasificador de estrategias

| $1^{er}$ movimiento | Enroque | Piezas adelantadas | Peones amenazando rey | Estrategia |
|---|---|---|---|---|
| e4 | iguales | 2 | 1 | Atacante |
| Cf6 | opuestos | 0 | 2 | Atacante |
| Cf3 | iguales | 0 | 1 | Mixta |
| d4 | iguales | 1 | 0 | Defensiva |
| c5 | iguales | 0 | 0 | Atacante |
| c4 | opuestos | 1 | 2 | Defensiva |
| otro_n | iguales | 1 | 1 | Mixta |

Usando este clasificador, BayesChess determina la estrategia del oponente instanciando las cuatro variables predictoras y calculando, para los valores

instanciados, cuál es el valor de la variable *Situación del oponente* con mayor probabilidad.

### 5.1. Proceso de adaptación al oponente

Una vez que determinado el tipo de juego que está desarrollando el oponente, BayesChess decide su propia estrategia usando la red bayesiana de ajuste de los parámetros de la heurística como sigue:

- Cuando la estrategia del oponente es **atacante**, elige aquellos parámetros que **minimizan la probabilidad de perder**.
- Cuando la estrategia del oponente es **defensiva**, elige aquellos parámetros que **maximizan la probabilidad de ganar**.
- Cuando la estrategia del oponente se clasifique como mixta, elige **aleatoriamente** una de las dos opciones anteriores.

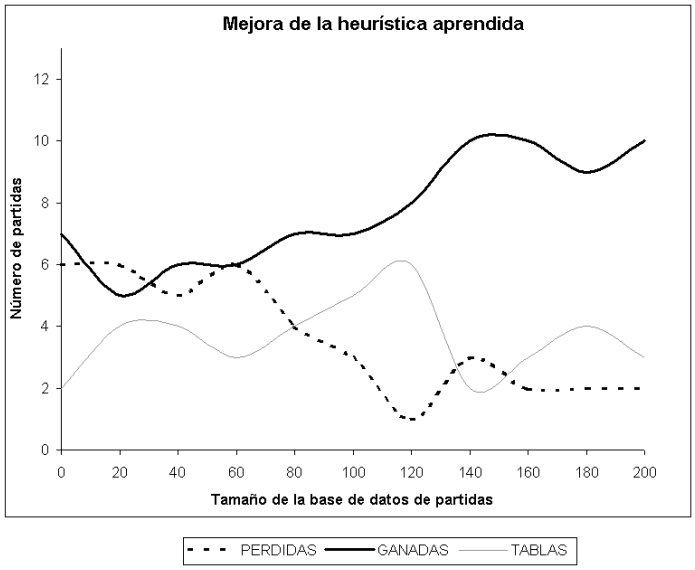



**Figura 5.** Evolución de los resultados entre la heurística aprendida y la fija

## 6. Experimentación

Hemos realizado dos experimentos para evaluar el proceso de aprendizaje de la heurística, en ambos casos usando una base de datos con 200 partidas jugadas entre la heurística inicial y una aleatoria.

El primer experimento consistió en realizar 7 torneos de 15 partidas entre BayesChess con la heurística fija y él mismo con la heurística aprendida con subconjuntos de más o menos partidas de la base de datos. En la figura 5 se observa cómo la heurística aprendida mejora sus resultados conforme crece el número de partidas jugadas.

El segundo experimento consistió en evaluar la puntuación asignada por la heurística a una posición determinada, concretamente a la mostrada en la figura 6, con distinto número de partidas en la base de datos. Se observa que en la posición de la figura 6, las blancas cuentan con un caballo y dos peones de ventaja, con lo que la valoración debe estar alrededor de -500 puntos (medidos desde el punto de vista de las negras). En la figura 7 puede observarse cómo efectivamente la heurística se acerca a dicho valor conforme crece el tamaño de la base de datos.

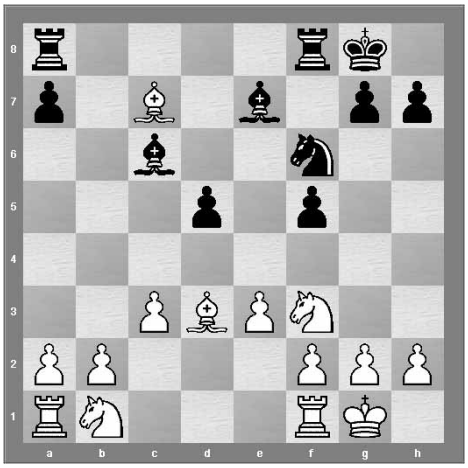



**Figura 6.** Tablero ejemplo para el segundo experimento

## 7. Conclusiones

En este hemos presentado BayesChess, un programa de ajedrez que adapta su comportamiento al enemigo al que se enfrenta y a su propia experiencia de juego. Los resultados de la experimentación indican que el aprendizaje conlleva una mejora práctica de los resultados y que la heurística ajusta sus parámetros hacia valores más precisos.

Pensamos que el uso de redes bayesianas aporta un valor añadido en la construcción de sistemas adaptables al usuario. En casos como BayesChess, donde el número de variables a tener en cuenta es muy alto, permiten hacer inferencias de forma eficiente utilizando topologías de red restringidas como el Naive Bayes.

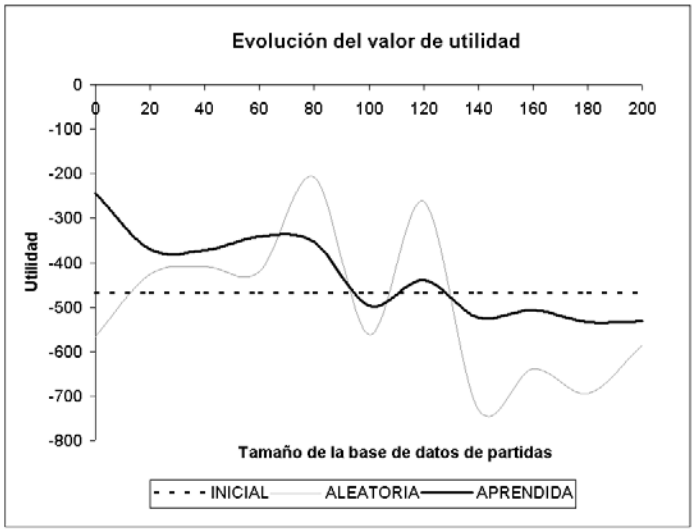



**Figura 7.** Evolución del valor de utilidad conforme aumenta el tamaño de la base de datos

No solamente el ajedrez, sino otros juegos de ordenador que requieran toma de decisiones por parte de la máquina, pueden beneficiarse del uso de redes bayesianas en la misma forma en que se propone en este trabajo. Un ejemplo inmediato son las damas, pero hay que tener en cuenta que en este caso la complejidad del juego es muy inferior, y por tanto la heurística, al menos a priori, no habrá de tener en cuenta tantas variables.

En un futuro esperamos mejorar el comportamiento puramente ajedrecístico de BayesChess refinando la implementación del algoritmo mini-max e introduciendo un nuevo clasificador que se emplee en los finales de juego. En este sentido, existen bases de datos con posiciones típicas de finales de torres y peones clasificadas como ganadoras, perdedoras o de tablas, que pueden ser utilizadas para entrenar el citado clasificador [1].

## Agradecimientos

Trabajo subvencionado por el Ministerio de Educación y Ciencia, proyecto TIN2004-06204-C03-01 y por fondos FEDER.

## Referencias

1. C.L. Blake and C.J. Merz. UCI repository of machine learning databases. www.ics.uci.edu/∼mlearn/MLRepository.html, 1998. University of California, Irvine, Dept. of Information and Computer Sciences.

2. L.M. de Campos, J.A. Gámez, and S. Moral. Partial abductive inference in Bayesian networks by using probability trees. In *Proceedings of the 5th International Conference on Enterprise Information Systems (ICEIS'03)*, pages 83–91, Angers, 2003.
3. R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern classification*. Wiley Interscience, 2001.
4. J. Fürnkranz. Machine learning in computer chess: The next generation. *ICCA Journal*, 19(3):147–161, 1996.
5. J.A. Gámez, S. Moral, and A. Salmerón. *Advances in Bayesian networks*. Springer, Berlin, Germany, 2004.
6. J. Hernández, M.J. Ramírez, and C. Ferri. *Introducción a la minería de datos*. Pearson, 2004.
7. Finn V. Jensen. *Bayesian networks and decision graphs*. Springer, 2001.
8. K. Muller. The clash of the titans: Kramnik - FRITZ Bahrain. *IGCA Journal*, 25:233–238, 2002.
9. M. Newborn. *Kasparov vs. Deep Blue: Computer chess comes of age*. Springer-Verlag, 1997.
10. D. Nilsson. An efficient algorithm for finding the M most probable configurations in Bayesian networks. *Statistics and Computing*, 9:159–173, 1998.
11. N. Nilsson. *Inteligencia artificial: una nueva síntesis*. McGraw-Hill, 2004.
12. J. Pearl. *Probabilistic reasoning in intelligent systems*. Morgan-Kaufmann (San Mateo), 1988.
13. S.S. Skiena. An overview of machine learning in computer chess. *ICCA Journal*, 9(1):20–28, 1986.

# Un nuevo algoritmo de selección de rasgos basado en la Teoría de los Conjuntos Aproximados

Yailé Caballero[1], Rafael Bello[2], Delia Alvarez[1],
Maria M. Garcia[2] y Analay Baltá[1]

[1] Departamento de Computación. Universidad de Camagüey, Cuba
{yaile, dalvarez}@inf.reduc.edu.cu
[2] Departamento de Computación. Universidad Central de Las Villas, Cuba
{rbellop, mmgarcia}@uclv.edu.cu

**Resumen.** La Teoría de los Conjuntos Aproximados es una técnica para el análisis de datos. En este trabajo se investiga la utilidad de los conjuntos aproximados y la Teoría de la Información, en la selección de rasgos relevantes para formar un buen reducto. Se crea e implementa un nuevo método con estos fines. Este nuevo método es un algoritmo glotón que a través de heurísticas llega a formar un buen reducto en tiempos aceptables. Este método se compara con otros existentes y se obtienen resultados satisfactorios.

## 1 Introducción

Con el crecimiento actual de los volúmenes de información en bases de datos tanto científicas como corporativas, la necesidad de determinar qué información es realmente importante se convierte en un reto para los desarrolladores debido a que se complejizan las tareas para la minería de datos (data mining) y el aprendizaje automatizado (machine learning).

La selección de rasgos en un conjunto de datos, es un problema en cuya solución se han utilizado diversas variantes dentro de la Inteligencia Artificial, debido a su utilidad en gran cantidad de áreas de la Informática y la Ciencia de la Computación, que tienen como denominador común reducir la dimensionalidad de los problemas de tal forma que grandes volúmenes de datos puedan ser manipulados rápidamente al extraer de ellos solo la información necesaria que los describa, sin perder la calidad del sistema, y permita obtener conocimiento sobre ellos; esto se usa fundamentalmente en problemas de clasificación, aplicaciones recientes de la selección de rasgos pueden ser encontradas en clasificación de contenido web, procesamiento de textos, diagnóstico médico, entre otras.

Los conjuntos de datos consisten en un sistema de información descrito por un conjunto de atributos y los objetos en sí que representan combinaciones de valores válidos dentro del dominio de cada atributo. De esta forma, la selección rasgos en un sistema de información de este tipo, consiste en obtener un subconjunto de atributos tal, que describa el sistema como si se tratara del conjunto completo. Esto quiere decir que el proceso se centra en encontrar los atributos más importantes dentro de los que

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 625-636, 2006.

se han utilizado para representar los datos y eliminar aquellos que se consideran irrelevantes y hacen más difícil el proceso de descubrimiento de conocimiento dentro de una base de datos. Dicho de otro modo, la selección de rasgos representa el problema de encontrar un subconjunto óptimo de características (rasgos o atributos) en una base de datos y según cierto criterio, tales que se pueda generar un clasificador con la mayor calidad posible a través de un algoritmo inductivo que corra sobre los datos, pero solo tomando en cuenta el subconjunto de atributos obtenido [Zho01].

El proceso de obtención de este subconjunto de atributos se caracteriza por utilizar una función de evaluación, por lo general heurística, que intenta obtener un subconjunto candidato por medio de una estrategia de búsqueda. Seleccionar los rasgos relevantes de un conjunto de datos es una tarea necesaria dentro del aprendizaje automatizado (machine learning), dada su importancia en el descubrimiento de reglas y / o relaciones en grandes volúmenes de datos entre otras aplicaciones, es por eso que la selección de características relevantes de un conjunto de datos con tiempos y costo de cómputo aceptable, ha sido en los últimos años tema de investigación de muchos autores en diferentes variantes. Confróntese, entre otros, los siguientes: [Wro95], [Deo98], [Cho96], [Liu98], [Koh94a], [Koh94b], [Mat96].

Una herramienta matemática muy potente dentro de esta tarea de selección es la Teoría de los Conjuntos Aproximados (Rough Sets Theory), esta fue propuesta por el profesor polaco Z. Pawlak y su equipo en 1982 [Paw82]. La filosofía de los Conjuntos Aproximados (Rough Sets) se basa en asumir que existe información asociada con cada objeto del universo de discurso [Kom99a]. Un conjunto de entrenamiento se representa por una tabla donde cada fila representa un objeto y cada fila un atributo, a este conjunto se le llama sistema de información, más formalmente, es un par S= (U, A) donde U es un conjunto no vacío y finito de objetos llamado universo y A es un conjunto no vacío y finito de atributos. Un sistema de decisión es cualquier sistema de información de la forma SD=(U, A$\cup${d}) donde d$\notin$A es un atributo de decisión, o sea, un atributo que indica a qué grupo pertenece el objeto.

El modelo de los Conjuntos Aproximados posee importantes ventajas dentro del análisis de datos, la principal consiste en que se basa únicamente en los datos originales y no requiere de información externa para obtener conocimiento sobre el sistema, de forma que no es necesario hacer suposiciones sobre este; la otra ventaja importante consiste en que esta herramienta permite analizar atributos tanto cuantitativos como cualitativos. Como se verá en el presente artículo, este modelo es muy útil dentro de la selección de rasgos, inicialmente se introducen la Teoría de los Conjuntos Aproximados (epígrafe 2) y a continuación se presenta una aplicación del mismo dentro de un algoritmo glotón (greedy) de selección basado en una función de evaluación heurística (epígrafe 3) junto con sus resultados experimentales y procesamiento estadístico (epígrafe 4).

## 2 La Teoría de los Conjuntos Aproximados

Con frecuencia son almacenados grandes volúmenes de información en bases de datos con diferentes objetivos; estos pueden ser adquiridos de mediciones obtenidas

por expertos humanos o de representaciones de hechos específicos de problemas de la vida cotidiana.

Una base de datos puede contener cierta cantidad de atributos que son redundantes u objetos que se encuentran repetidos en distintos niveles de esta, pero sobre todo sucede que contiene información insuficiente o incompleta.

La Teoría de los Conjuntos Aproximados emerge desde el contexto del aprendizaje supervisado, donde los conjuntos de datos se refieren a un universo de objetos descritos por un conjunto de atributos y cada objeto pertenece a una clase predefinida por uno de los atributos, llamado atributo de decisión.

Para una aproximación inicial, considérese que cada conjunto de datos está representado por una tabla, donde cada fila constituye un caso, un evento, un paciente o simplemente un objeto; y cada columna, un atributo que puede ser una variable, una observación, una columna, una propiedad, etc., tal que posee un valor específico para cada objeto. A esta tabla se le llama sistema de información, una definición formal del mismo según [Dun00], es:

$I = \{U, A, V_q, f_q\}_{q \in A}$ donde:

U es un conjunto finito y no vacío de objetos llamado universo.

A es un conjunto finito y no vacío de atributos.

$V_q$ es el conjunto de valores posibles para cada atributo de A de modo que $q : U \to V_q$ para todo $q \in A$.

$f_q$ es una función de información tal que $f_q : U \to V_q$.

Dado un sistema de información como el descrito anteriormente, existe una relación de equivalencia:

$$IND_I(B) = \{(x,y) \in U^2 / \forall a \in B \;\; a(x) = a(y)\}$$

en la que INDI es la relación de inseparabilidad basada en el subconjunto de atributos B. Si $(x,y) \in IND_I(B)$ entonces x e y son inseparables con respecto a B.

Una relación de inseparabilidad induce una partición del universo. Estas particiones pueden ser usadas para construir nuevos subconjuntos del universo. Los subconjuntos de mayor interés son aquellos que tienen incluso el mismo valor para el atributo de decisión, sin embargo, puede suceder que los atributos condicionales de algunos objetos sean inseparables, mientras que el de decisión sea diferente; la solución a problemas de este tipo se encuentra en el uso de los conjuntos aproximados, los cuales se definen a través de sus aproximaciones superior e inferior.

Dados un sistema de información $I = \{U, A, V_q, f_q\}_{q \in A}$, $B \subseteq A$ y $X \subseteq U$, entonces se puede aproximar X usando solo la información contenida en B construyendo las aproximaciones inferior y superior de X, denotadas $B^*$ y $B_*$ respectivamente, y definidas de la siguiente forma:

$B_* = \{x / [x]_B \subseteq X\}$ y $B^* = \{x / [x]_B \cap X \neq \phi\}$ [Paw91]

En [Bel02] se definen de la siguiente forma:

La aproximación inferior de un conjunto (con respecto a un conjunto dado de atributos) se define como la colección de casos cuyas clases de equivalencia están conte-

nidas completamente en el conjunto; mientras que la aproximación superior se define como la colección de casos cuyas clases de equivalencia están al menos parcialmente contenidas en el conjunto.

A partir de las aproximaciones inferior y superior, se pueden definir medidas para inferir conocimiento sobre las bases de casos a través de una serie de medidas que se definen sobre estos conceptos, una de ellas es al precisión de la clasificación, que se especifica según la cantidad de valores de decisión que posee el sistema, la medida, también llamada vaguedad del concepto o conjunto X con respecto a la relación B, se puede caracterizar matemáticamente por el coeficiente:

$$\alpha_B(X) = \frac{|B_*(X)|}{|B^*(X)|}$$

Donde $|X|$ denota la cardinalidad del conjunto X, y $0 \leq \alpha B(X) \leq 1$. Si $\alpha B(X)=1$, el conjunto X será duro o exacto con respecto a la relación de equivalencia B, mientras que si $\alpha B(X)<1$, el conjunto X es aproximado o vago con respecto a B. Esta magnitud mide el grado de perfección o integridad del conocimiento sobre el conjunto X considerando los atributos incluidos en la relación de equivalencia.

Un punto importante dentro del análisis de datos es el descubrimiento de dependencias entre estos. Intuitivamente, un conjunto de atributos D depende totalmente de un conjunto de atributos C, denotado $C \rightarrow D$, si todos los valores de los atributos de D están únicamente determinados por valores de los atributos de C. En otras palabras, D depende totalmente de C si existen dependencias funcionales entre los atributos de C y D. [Kom99]

Formalmente la dependencia puede ser definida de la siguiente forma: si C y D son subconjuntos de A entonces se dice que D depende de C en grado k ( $0 \leq k \leq 1$ ) si:

$$k = \gamma(C,D) = \frac{|POS_C(D)|}{|U|}$$ [Kom99]

Donde POS es la región positiva de la partición U/D con respecto a C, o sea, el conjunto de todos los elementos de U que pueden ser únicamente clasificados en bloques de la partición U/D por medio de C.

Si k = 1, se dice que D depende totalmente de C y si k<1, se dice que D depende parcialmente, en grado k, de C.

Si D depende totalmente de C entonces $IND(C) \subseteq IND(D)$, lo que significa que la partición generada por C es mucho mejor o más fina que la generada por D. A la dependencia de los atributos también se le llama calidad de la clasificación.

Los conjuntos aproximados se aplican eficientemente en la reducción de atributos o selección de rasgos sobre la base del concepto de reducto. Dado un sistema de información $I = (U, A)$, un reducto es un conjunto mínimo de atributos $B \subseteq A$ tal que $IND_I(B) = IND_I(A)$. En otras palabras, un reducto es un conjunto mínimo de atributos de A que preservan la partición del universo y de esta forma la habilidad de ejecutar clasificaciones como si se tratara del mismo conjunto A.

El uso de reductos en selección de rasgos ha sido estudiado por varios autores, entre ellos [Koh94], [Car98], [Pal99], [Kom99b], [Ahn00] y [Zho01].

Sin embargo, esta beneficiosa alternativa se encuentra limitada por El problema es que encontrar esos conjuntos mínimos de atributos es un problema NP-hard. Dado un sistema de información de $m$ rasgos, existen $2^m$ posibles subconjuntos de rasgos. Esto constituye un cuello de botella de la Teoría de los conjuntos aproximados [Kom99], dado que computar esa cantidad de reductos no es una tarea trivial que pueda ser resuelta con pocos recursos de hardware, sino que lo cierto es que seleccionar los rasgos relevantes de un conjunto de datos es una tarea necesaria dentro del aprendizaje automatizado (machine learning), dada su importancia en el descubrimiento de reglas y / o relaciones en grandes volúmenes de datos entre otras aplicaciones, es por eso que el cálculo de reductos o la selección de características relevantes de un conjunto de datos con tiempos y costo aceptable, ha sido en los últimos años tema de investigación de muchos autores. Confróntese, entre otros, los siguientes: [Wro95], [Deo98], [Cho96], [Liu98], [Koh94a], [Koh94b], [Mat96].

La teoría de los conjuntos aproximados posee un importante potencial para investigar problemas relacionados con bases de datos del mundo real, bases que con frecuencia suelen ser largas y dinámicas. Los conjuntos aproximados ofrecen un marco formal para el descubrimiento de conocimiento (knowledge discovery) [Paw91]; es por eso se pueden utilizar como una herramienta de selección para descubrir dependencias entre datos y reducir el número de atributos contenidos en un conjunto de datos obteniendo un reducto del conjunto inicial de atributos con un mínimo de pérdidas de información.

Diversos autores han propuesto métodos para el cálculo de reductos a través de los conjuntos aproximados [Yao99], [Koh94], [Paw91], [Zho01], [Dun00] y [Jen03].

## 3 RSReduct: Un nuevo método de selección de rasgos

RSReduct es un método que trata de encontrar un reducto de manera que éste sea lo suficientemente bueno para el análisis de datos en tiempos aceptables. Para ello se utiliza la búsqueda heurística como estrategia de búsqueda, debido a que si se tomara en cuenta la variante exhaustiva o completa el consumo de tiempo y recursos de cómputo sería bastante grande y la no determinística dificulta saber cuándo aparece un subconjunto mínimo.

El método consiste en un algoritmo glotón que comienza por un conjunto vacío de atributos, y a través de heurísticas va formando un reducto a través de la selección de los atributos uno a uno de una lista, hasta que se cumple la condición de parada; en la lista de atributos; estos se encuentran ordenados según el valor arrojado por la función de evaluación heurística para cada uno de estos.

Para la construcción de las funciones de evaluación heurística se siguen criterios del método ID3 con respecto a la entropía y la ganancia de los atributos, dependencia entre atributos mediante los Conjuntos Aproximados, así como la opción de otorgar costos a los atributos, es decir, manipulación de atributos con costos diferentes.

En este algoritmo se utilizan las medidas R(A) y H(A) que se proponen en [Piñ03].

$$R(A) = \sum_{i=1}^{k} \frac{|S_i|}{|S|} e^{(1-C_i)}$$

En la expresión k es el número de valores diferentes del rasgo A. $C_i$ es el número de clases diferentes presentes en los objetos que tienen el valor i para el rasgo A y $\frac{|S_i|}{|S|}$ es la frecuencia relativa del valor i en S (cantidad de objetos con el valor i en el rasgo A sobre la cantidad de objetos de toda la muestra). La principal idea de esta medida es maximizar la heterogeneidad entre objetos que pertenecen a clases diferentes y minimizar la homogeneidad entre aquellos que son de la misma clase, además $0 \le R(A) \le 1$.

H(A) también se obtiene a través del siguiente algoritmo:

Se calcula el vector R(T)= (R(A1),...). Para todos los atributos del problema se calcula su R(A) y así con todos los valores se forma el vector R (T).

Se determinan los n mejores atributos por los cálculos del paso anterior. El valor de n se puede seleccionar por el usuario. Como resultado de este paso se obtiene el vector RM=(R(Ai), R(Aj),..) con $n = |RM|$.

Se determinan las combinaciones de n en p (valores seleccionados por el usuario) desde los atributos seleccionados en el paso II. Se obtiene el vector de combinaciones $Comb = (\{a_i, a_j, a_k\}, ..., \{a_i, a_t, a_p\})$. Por ejemplo: si n=4 y p = 3 y los atributos seleccionados en II son: $(a_1, a_3, a_5, a_8)$ el vector de de combinaciones $C_p^n = \frac{n!}{p!(n-p)!}$ tendría cuatro componentes que serían: $Comb = (\{a_1, a_3, a_5\}, \{a_1, a_3, a_8\}, \{a_3, a_5, a_8\}, \{a_1, a_5, a_8\})$.

Se calcula el grado de dependencia de las clases con respecto a cada una de las combinaciones obtenidas en el paso anterior. Como resultado de este paso se obtiene el vector de dependencias: $DEP = (k(comb_1, d), ..., k(comb_r, d))$ donde k representa la medida para el grado de dependencia entre atributos de los conjuntos aproximados con respecto a los valores de decisión d.

Para cada atributo A se calcula H(A) según la siguiente ecuación: $H(A) = \sum_{\forall i / A \in comb_i} k(comb_i, d)$

La ganancia radial G(A) o ganancia de Quinlan, es una medida alternativa para seleccionar atributos [Mit97]:

$$G(A) = \frac{Ganancia(S, A)}{SplitInformation(S, A)}$$

donde:

$Ganancia(S,A) = Entropía(S) - \sum_{v \in valores(A)} \frac{|S_v|}{|S|} Entropía(S_v)$ donde, valores (A) es el conjunto de valores posibles por el atributo $A$ y $S_v$ es el subconjunto de S para el cual A tiene el valor v, es decir, $S_v = \{s \in S \mid A(S) = v\}$.

$Entropía(S) = \sum_{i=1}^{c} - P_i \log_2 P_i$ es la clásica medida del valor de entropía de un sistema de información propuesta por Shannon y donde $P_i$ es la proporción de S perteneciente a la clase i. La $Entropía = 0$ si todos los elementos en $S$ pertenecen a la misma clase y la $Entropía = 1$ si todas las clases tienen igual número de ejemplos.

$SplitInformation(S,A) = -\sum_{i=1}^{C} \frac{|S_i|}{|S|} log_2 \frac{|S_i|}{|S|}$ donde C son los valores del atributo A. Esta medida es la entropía de S con respecto al atributo A.

Para calcular C(A) existen dos variantes, ambas aparecen en [Mit97]:

Medida de Schlimmer y Tan: $C(A) = \frac{Gnancia^2(S,A)}{Cost(A)}$ donde Cost(A) es un parámetro entrado por el usuario que representa el costo del atributo A.

Medida de Núñez: $C(A) = \frac{2^{Ganancia(S,A)} - 1}{(Cost(A)+1)^w}$ donde Cost(A) es un parámetro entrado por el usuario que representa el costo del atributo A y w es un valor constante entre 0 y 1 que determina la importancia relativa del costo contra la información de la ganancia.

Sobre la base de las medidas ya planteadas, se proponen tres funciones de evaluación heurística para el algoritmo RSReduct, estas permiten obtener una medida de la relevancia de los atributos dentro del conjunto de datos

*Heurística 1.* $RG(A)=R(A)+H(A)$
*Heurística 2.* $RG(A)=H(A)+G(A)$
*Heurística 3.* $RG(A)=H(A)+C(A)$

A continuación, se muestran los pasos del algoritmo RSReduct:

**P1.** Formar la tabla de distinción

Sea *B* matriz binaria $(M^2 - M)/2$ x $(N+1)$. Cada fila corresponde a un par de objetos diferentes. Cada columna de esta matriz corresponde a un atributo, la última columna corresponde a la decisión (tratada como un atributo).

Sea b( (k,n), i) un elemento de *B* correspondiente al par $O_k, O_n$ y al atributo i, para $i \in \{1,..N\}$:

$$b((k,n),i) = \left\{ \begin{matrix} 1, si\ a_i(O_k) \neg \Re a_i(O_n) \\ 0, si\ a_i(O_k) \Re a_i(O_n) \end{matrix} \right\} para i \in \{1,...N\}$$

$$b((k,n),N+1) = \begin{Bmatrix} 0, si\, d_i(O_k) \neq d_i(O_n) \\ 1, si\, d_i(O_k) = d_i(O_n) \end{Bmatrix}$$

donde $\Re$ es una relación de similaridad en dependencia del tipo del atributo $a_i$.

**P2.** Para cada atributo "A" se calcula el valor de RG(A) por cualquiera de las tres heurísticas. Se forma una lista ordenada de atributos comenzando por el atributo más relevante (el que maximice RG(A)).

*Heurística1.* $RG(A)=R(A)+H(A)$
*Heurística2.* $RG(A)=H(A)+G(A)$
*Heurística3.* $RG(A)=H(A)+C(A)$

**P3.** Se tiene i=1, R= conjunto vacío y se tiene $A_1, A_2, ..., A_n$ lista ordenada de atributos según el paso 2, si $i \leq n$ entonces $R = R \cup A_i$, $i = i+1$.

**P4.** Si R satisface la Condición I parar (que significa terminar).

$\forall k,n \quad \forall a_i \in R \quad a_i(o_k) \Re a_i(o_n) \Rightarrow d(o_k) = d(o_n)$ (Condición I)

Encontrar un reducto significa encontrar un conjunto mínimo de atributos que cubran a B, es decir, un subconjunto que satisfaga:

$\forall (k,n)\, \exists i \in R : b((k,n),i) = 1 \cup b((k,n),N+1) = 1$

Esto significa que para cada par de objetos que se tomen, o son de la misma clase o tienen algún atributo de R para el cual los objetos son "similares", es decir, cumplen la relación $\Re$.

**P5.** En otro caso repetir desde el paso 3.

## 4 Resultados experimentales

El algoritmo RSReduct fue probado con diferentes conjuntos de datos disponibles en el sitio ftp de la Universidad de California. Algunas de estas bases de datos pertenecen a datos del mundo real como Votos, Lirio, Cáncer de mama, Corazón y Créditos; las otras representan resultados obtenidos en laboratorio como Ballons-a, Hayes-Roth, LED, M-of-N, Cáncer pulmonar y Hongos.

En un primer experimento se tomó la base de casos Cáncer de mama, la cual representa un estudio del cáncer de mama realizado en julio de 1988 en el Instituto de Oncología perteneciente al Centro Médico Universitario de Ljubljana, Yugoslavia. Dicho conjunto de datos contiene 286 ejemplos de los cuales 201 pertenecen a una clase y 85 a otra; los ejemplos son descritos por 9 atributos algunos de los cuales son lineales y otros nominales, su descripción aparece en la tabla 1. Luego de haber probado esta base de casos con el método RSReduct a través de las tres funciones de evaluación heurística definidas para este se obtuvieron los resultados de la tabla 2. Los resultados de la tabla 3 fueron obtenidos luego de haber usado las tres funciones de evaluación heurísticas definidas para RSReduct para diferentes bases de casos; se compilaron la longitud del reducto y el tiempo de ejecución, en segundos, para cada caso.

Para ilustrar un poco mejor cuán buena fue la reducción del método propuesto, el gráfico 1 muestra el tamaño original de las bases de casos y el tamaño al que fueron

reducidas al usar las tres heurísticas de RSReduct, en azul fuerte se representa la cantidad inicial de atributos en el conjunto de datos y en rojo, amarillo y azul claro los reductos obtenidos para las funciones de evaluación heurística 1, 2 y 3 respectivamente.

**Tabla 1** Descripción de la base de casos Cáncer de mama

| Nombre del atributo | Valores de su dominio |
|---|---|
| Edad | 10-19, 20-29, 30-39, 40-49, 50-59, 60-69, 70-79, 80-89, 90-99 |
| Menopausia | lt40 (antes de los 40), ge40 (después o a los 40), premeno (premenopausia) |
| Tamaño del tumor | 0-4, 5-9, 10-14, 15-19, 20-24, 25-29, 30-34, 35-39, 40-44, 45-49, 50-54, 55-59 |
| Nodos invertidos | 0-2, 3-5, 6-8, 9-11, 12-14, 15-17, 18-20, 21-23, 24-26, 27-29, 30-32, 33-35, 36-39 |
| Capas de nodos | Sí, No |
| Grado de malignidad | 1, 2, 3 |
| Mama | Izquierda, Derecha |
| Cuadrante de mama | Izquierdo-arriba, izquierdo-bajo, derecho-arriba, derecho-bajo, central |
| Irradiaciones | Sí, No |
| Clase | eventos-no-recurrentes, eventos-recurrentes |

**Tabla 2** Descripción de los reductos obtenidos para la base de casos Breast Cancer

| Heurística utilizada | Reductos obtenidos |
|---|---|
| Heurística 1 | Tamaño del tumor, Grado de malignidad, Edad, Menopausia. |
| Heurística 2 | Tamaño del tumor, Grado de malignidad, Capas de nodos, Nodos invertidos. |
| Heurística 3 (en sus dos variantes) | Tamaño del tumor, Grado de malignidad, Edad, Nodos invertidos, Irradiaciones. |

Estos resultados experimentales fueron comparados estadísticamente en aras de buscar diferencias con otros métodos de selección de rasgos implementados con técnicas de reconocimiento de patrones [Alv05], algoritmos de estimación de distribuciones [Alv05], y algoritmos genéticos [Ohr97]. Se usó la prueba de Kruskal-Wallis, esta es una prueba no paramétrica basada en suma de rangos que compara más de dos grupos relacionados entre sí de una vez con el objetivo de descubrir diferencias entre ellos. En la tabla 4 aparecen los resultados de la prueba de Kruskal-Wallis entre los algoritmos descritos que fueron comparados en cuanto a tiempo de ejecución de los mismos; como puede ser visto, para todos los casos los resultados fueron menores de 0.05 lo que indica un 95% de significancia estadística, en otras palabras, existen diferencias significativas entre los métodos.

La conclusión de este análisis es que en un tiempo más pequeño se puede obtener un reducto lo suficientemente bueno en relación con longitud y diferenciación entre clases, de modo que el nuevo método RSReduct disminuye el costo computacional de problemas de clasificación.

Para comprobar aún más la eficiencia del método, se calcularon la calidad y la precisión por clase de la clasificación para el conjunto de datos completo y el conjunto de datos reducido, para las bases de casos de la tabla 5.

**Tabla 3** Resultados experimentales de las tres funciones de evaluación heurística de RSReduct para las diferentes bases de casos.

| **Nombre de la base de casos** (Cantidad de casos, Cantidad de atributos) | **Heurística 1** | | **Heurística 2** | | **Heurística 3** | |
|---|---|---|---|---|---|---|
| | Tiempo (segundos) | Longitud del reducto | Tiempo (segundos) | Longitud del reducto | Tiempo (segundos) | Longitud del reducto |
| Ballons-a (20,4) | 5.31 | 2 | 3.12 | 2 | 16.34 | 2 |
| Lirio (150,4) | 40.15 | 3 | 30.79 | 3 | 34.73 | 3 |
| Hayes-Roth (133,4) | 36.00 | 3 | 32.30 | 3 | 39.00 | 3 |
| Bupa (345,6) | 74.20 | 6 | 89.00 | 6 | 89.00 | 6 |
| EColi (336,7) | 57.00 | 5 | 41.15 | 5 | 46.60 | 5 |
| Corazón (270,13) | 30.89 | 9 | 16.75 | 9 | 54.78 | 10 |
| Diabetes (Pima) (768,8) | 110.00 | 8 | 110.00 | 8 | 110.00 | 8 |
| Cáncer de mama (683,9) | 39.62 | 4 | 31.15 | 4 | 32.56 | 5 |
| Levadura (1484,8) | 82.00 | 6 | 78.00 | 6 | 85.70 | 6 |
| Dermatología (358,34) | 148.70 | 8 | 125.9 | 8 | 190.00 | 9 |
| Cáncer pulmonar (27,56) | 25.46 | 7 | 18.59 | 7 | 31.5 | 8 |
| LED (226,25) | 78.10 | 9 | 185.00 | 8 | 185 | 9 |
| M-of-N (1000,14) | 230.26 | 6 | 162.50 | 6 | 79.4 | 6 |
| Exacto (780,13) | 230.00 | 11 | 215.00 | 11 | 230 | 11 |
| Hongos (3954,22) | 86.20 | 8 | 64.10 | 8 | 67.2 | 8 |
| Créditos (876,20) | 91.20 | 14 | 86.01 | 14 | 90.2 | 15 |
| Votos (435,16) | 37.93 | 12 | 21.25 | 11 | 26.9 | 12 |

**Tabla 4** Significancia de la prueba de Kruskal-Wallis entre las tres funciones heurísticas de RSReduct con otros métodos de selección de rasgos.

| **Bases de casos representativas** | **Significancia de Heurística 1 vs otros métodos** | **Significancia de Heurística 2 vs otros métodos** | **Significancia de Heurística 3 vs otros métodos** |
|---|---|---|---|
| Cáncer de mama | 0.0039 | 0.0039 | 0.0039 |
| Cáncer pulmonar | 0.002 | 0.002 | 0.002 |
| Hongos | 0.0034 | 0.0034 | 0.0034 |
| Corazón | 0.0039 | 0.0039 | 0.0039 |
| Dermatología | 0.0265 | 0.0265 | 0.0265 |

Al realizar una simple inspección a la tabla 5, es posible observar que no hay diferencias en calidad y precisión de la clasificación entre la base de casos completas y el reducto que se obtuvo a través de la segunda función heurística de RSReduct, esto nos lleva a concluir, sobre la base de los conjuntos aproximados, que no existen pér-

didas de información, o al menos estas son lo suficientemente pequeñas, al emplear el nuevo método de selección de rasgos propuesto.

**Gráfico 1** Representación gráfica de los resultados experimentales de RSReduct para las diferentes bases de casos.

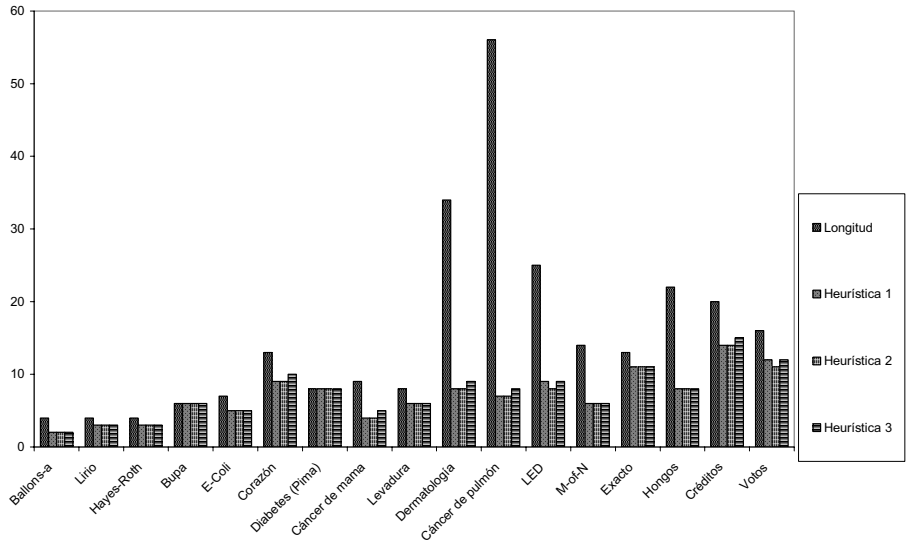



**Tabla 5** Medidas de inferencia basadas en los conjuntos aproximados para algunas bases de casos y el reducto obtenido por RSReduct para las mismas.

| **Bases de casos representativas** | **Calidad de la clasificación para la base completa** | **Calidad de la clasificación para el reducto** | **Precisión de la clasificación por clase para la base completa** | | **Precisión de la clasificación por clase para el reducto** | |
|---|---|---|---|---|---|---|
| Cáncer de mama | 1.0 | 1.0 | Clase 0 | 1.0 | Clase 0 | 1.0 |
| | | | Clase 1 | 1.0 | Clase 1 | 1.0 |
| Cáncer pulmonar | 0.8519 | 0.8519 | Clase 1 | 1.0 | Clase 1 | 1.0 |
| | | | Clase 2 | 0.6667 | Clase 2 | 0.6667 |
| | | | Clase 3 | 0.6364 | Clase 3 | 0.6364 |
| Hongos | 1.0 | 1.0 | Clase 1 | 1.0 | Clase 1 | 1.0 |
| | | | Clase 2 | 1.0 | Clase 2 | 1.0 |
| Corazón | 1.0 | 1.0 | Clase 0 | 1.0 | Clase 0 | 1.0 |
| | | | Clase 1 | 1.0 | Clase 1 | 1.0 |
| Dermatología | 1.0 | 0.9944 | Clase 1 | 1.0 | Clase 1 | 1.0 |
| | | | Clase 2 | 1.0 | Clase 2 | 0.9815 |
| | | | Clase 3 | 1.0 | Clase 3 | 0.9728 |
| | | | Clase 4 | 1.0 | Clase 4 | 0.9667 |
| | | | Clase 5 | 1.0 | Clase 5 | 1.0 |

## 5 Conclusiones

El nuevo método de selección de rasgos a través del concepto de reducto que se propone: RSReduct, fue analizado en cuanto a su eficiencia al ser probado con diferentes bases de casos, la mayoría de las cuales redujo en gran medida; también se realizaron comparaciones con otros métodos, obteniendo un 95% de significancia estadística que indica importantes diferencias entre RSReduct y los otros métodos, favorables para el primero en cuanto a que este es capaz de encontrar un reducto lo suficientemente bueno en tiempos aceptables. Para probar, además, que no existían pérdidas de información sobre el sistema en el reducto obtenido, se realizaron pruebas con las medidas de inferencia de calidad y precisión de los conjuntos aproximados sobre dicho reducto y sobre la base completa, para obtener los mismos valores.

## Referencias

[Ahn00] Ahn, B.S. et al.. The integrated methodology of rough set theory and artificial neural networks for business failure predictions. Expert Systems with Applications 18, 65-74. 2000.

[Alv05] Álvarez, Delia. Feature selection for data analysis using Rough Sets Theory. Thesis of Computer Science Engineering. Thesis Director: Yailé Caballero, M. Sc. University of Camagüey, Cuba. 2005.

[Bel98] Bell, D. and Guan, J. Computational methods for rough classification and discovery. Journal of ASIS 49, 5, pp. 403-414. 1998.

[Car98] Carlin, U.S. et al.. Rough set analysis of medical datasets and A case of patient with suspected acute appendicitis. In ECAI 98 Workshop on Intelligent data analysis in medicine and pharmacology.

[Jen03] Jensen R. and Qiang, S. "Finding rough sets reducts with Ant colony optimization". http://www.inf.ed.ac.uk/publications/online/0201.pdf 2003.

[Koh94] Kohavi, R. and Frasca, B. Useful feature subsets and Rough set Reducts. Proceedings of the Third International Workshop on Rough Sets and Soft Computing. 1994.

[Kom99a] Komorowski, J. Pawlak, Z. et al.. Rough Sets: A tutorial. In Pal, S.K. and Skowron, A. (Eds) Rough Fuzzy Hybridization: A new trend in decision-making. Springer, pp. 3-98. 1999.

[Kom99b] Komorowski, J. et al.. A Rough set perspective on Data and Knowledge. In The Handbook of Data mining and Knowledge discovery, Klosgen, W. and Zytkow, J. (Eds). Oxford University Press, 1999.

[Mit97] Mitchell, Tom., Hill, McGraw. Machine Learning. ISBN: 0-07- 042807-7. 1997. http://www.cs.cmu.edu/~tom/mlbook.html

[Ohr01] Ohrn, A.. ROSETTA Technical Reference Manual. Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway.

[Pal02] Pal, S.K. et al. Web mining in Soft Computing framework: Relevance, State of the art and Future Directions. IEEE Transactions on Neural Networks, 2002.

[Paw82] Pawlak, Z.. Rough sets. International Journal of Information & Computer Sciences 11, 341-356, 1982.

[Paw91] Pawlak, Z. Rough SetsTheoretical Aspects of Reasoning About Data. Kluwer Academic Publishing, Dordrecht, 1991. En: http://citeseer.ist.psu.edu/context/36378.html

[Paw94] Pawlak, Z. and Skowron, A. "Rough sets rudiments". Bulletin of International Rough Set Society. Volume 3, Number 3. http://www.kuenstliche-intelligenz.de/archiv/2001_3/pawlak.pdf

[Paw95] Pawlak, Z. "Rough Sets, Rough Relations and Rough functions". R. Yager, M.Fedrizzi, J. Keprzyk (eds.): Advances in the Dempster – Shafer Theory of Evidence, Wiley, New Cork, pp 251 – 271. 1995 http://citeseer.ist.psu.edu/105864.html

[Zho01] Zhong, N. et al.. Using Rough sets with heuristics for feature selection. Journal of Intelligent Information Systems, 16, 199-214. 2001.

# La Teoría de los Conjuntos Aproximados en la edición de conjuntos de entrenamiento para mejorar el desempeño del método k-NN

Yailé Caballero[1], Rafael Bello[2], Yaimara Pizano[1], Delia Alvarez[1], Maria M. Garcia[2] y Analay Baltá[1]

[1] Departamento de Computación. Universidad de Camagüey, Cuba
{yaile, dalvarez}@inf.reduc.edu.cu
[2] Departamento de Computación. Universidad Central de Las Villas, Cuba
{rbellop, mmgarcia}@uclv.edu.cu

**Resumen.** La Teoría de los Conjuntos Aproximados es una técnica para el análisis de datos. En este trabajo, se usa esta teoría para mejorar el desarrollo del método k-NN. La Teoría de los conjuntos aproximados es usada para editar y reducir el conjunto de entrenamiento. Se proponen dos métodos para editar conjuntos de entrenamiento, basados en las aproximaciones inferior y superior. Los resultados experimentales muestran un desarrollo satisfactorio del método k-NN usando estas técnicas.

## 1 Introducción

Uno de los objetivos del Aprendizaje Automático es la clasificación de ejemplos no vistos previamente. Comenzando con un conjunto de ejemplos, el sistema aprende cómo predecir la clase de cada uno basado en sus rasgos. El aprendizaje basado en instancias (IBL) es un método de aprendizaje de máquina el cual clasifica nuevos ejemplos comparándolos con los ya vistos almacenados en memoria. Una forma de resolver un problema de clasificación es encontrar el ejemplo más cercano almacenado, tomando en cuenta algunas funciones de similaridad; el problema se clasifica entonces según la clase de su vecino más cercano. Kibler y Aha mostraron que el más sencillo de los modelos del vecino más cercano puede producir resultados excelentes para una variedad de dominios [Aha91].

Una extensión básica al paradigma de IBL consiste en usar el k-Vecinos más Cercanos (k-NN) en vez de solo el más cercano; la clase asignada es de la mayoría de los k vecinos más cercanos tomando en cuenta la distancia (o similaridad) entre el problema y cada vecino más cercano. La búsqueda de los vecinos más cercanos puede ser una tarea muy costosa sobre todo en espacios de altas dimensiones. Dos elementos determinan el costo computacional del método k-NN: la cantidad de rasgos y la cantidad de objetos. Un problema del aprendizaje basado en instancias es que el tiempo de la clasificación aumenta mientras más ejemplos tenga el conjunto de entrenamiento (TS).

En la solución de problemas de clasificación hay dos aspectos muy importantes: la

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 637-645, 2006.

reducción del error de la clasificación y la reducción del costo computacional. El primero de estos aspectos está referido a que por ejemplo, los métodos del vecino más cercano son muy sensibles a la presencia de ejemplos incorrectamente etiquetados y a ejemplos cercanos a las fronteras de decisión, lo que trae consigo errores a la hora de realizar la clasificación. La reducción del error en la clasificación estará encaminada entonces a resolver problemas como este del vecino más cercano. Una vía de solución a este problema es a través de las diferentes técnicas de edición del conjunto de ejemplos.

La clasificación puede resultar en la práctica, tediosa y en ocasiones inaplicable si se dispone de conjuntos de entrenamiento numerosos y con datos de alta dimensionalidad. Estos dos factores determinan el costo computacional de la aplicación de la mayoría de los métodos de clasificación existentes:

1. La dimensionalidad de los datos, d.
2. El tamaño del conjunto de entrenamiento, N.

Existen diferentes estrategias que permiten la aplicación de los métodos de clasificación con un costo computacional menor. Estas estrategias pueden agruparse en dos grupos, de acuerdo a la filosofía en que se basan:

I. Reducir el conjunto de entrenamiento sin perder en la calidad de la clasificación.

II. Mejorar la eficiencia computacional de los métodos de clasificación.

Los métodos de edición siguen el principio de la estrategia del grupo I y se basan en la selección de un subconjunto del conjunto de entrenamiento que tenga las mismas propiedades que el conjunto original para, una vez editado, aplicar sobre este nuevo conjunto la clasificación.

Uno de los problemas presentes cuando se realiza la solución de problemas con técnicas de Inteligencia Artificial usando datos y no conocimiento explícito es lograr trabajar con la menor cantidad posible de información sin perder calidad en la solución que se encuentre. Un ejemplo claro aparece en la solución de problemas usando Conjuntos Aproximados, en el cual es de interés definir la menor relación de equivalencia posible. Una forma de reducción de los datos se logra cuando usando la relación se construyen las clases de equivalencias, pues basta con considerar un elemento de la clase en lugar de la clase completamente.

## 2 La Teoría de los Conjuntos Aproximados

La Teoría de Conjuntos Aproximados (Rough Sets Theory) fue introducida por Z. Pawlak en 1982 [Paw82]. Se basa en aproximar cualquier concepto, un subconjunto duro del dominio como por ejemplo una clase en un problema de clasificación, por un par de conjuntos exactos, llamados aproximación inferior y aproximación superior del concepto. La filosofía de los conjuntos aproximados está basada en la suposición de que con todo objeto de un universo $U$ está asociada una cierta cantidad de información (datos y conocimiento), expresado por medio de algunos atributos usados para describir el objeto [Kom99].

Esta teoría ha mostrado ser una excelente herramienta matemática para el análisis [Gre01], [Pol02]. Entre otras razones porque es posible tratar tanto con datos cuantitativos como cualitativos, y no se requiere eliminar las inconsistencias previas al

análisis, además, no son necesarias suposiciones sobre la independencia de los atributos ni ningún otro conocimiento sobre la naturaleza de los datos.

El conjunto de entrenamiento puede ser representado por una tabla donde las filas representan los objetos y las columnas los atributos que describen a dichos objetos. Esta tabla es llamada Sistema de Información, más formalmente, esto es un par S=(U, A), donde U es un conjunto finito no vacío de objetos llamado universo y A es un conjunto no vacío de atributos. Un Sistema de Decisión es cualquier sistema de Información de la forma SD=(U, A$\cup${d}), donde d$\notin$A es el atributo de decisión. Las definiciones clásicas de aproximación inferior y aproximación superior fueron introducidas originalmente respecto a una relación de indiscernibilidad la cual es asumida como una relación de equivalencia.

Se tiene B$\subseteq$A y X$\subseteq$U. B define una relación de equivalencia y X es un concepto. X puede ser aproximado usando solamente la información contenida en B, a través de la construcción de las aproximaciones inferior y superior de X, denotada por $B_*X$ y $B^*X$ respectivamente, donde $B_*X$={ x : [x]B $\subseteq$X } y $B^*X$={ x : [x]B $\cap$ X$\neq\phi$ }, y [x]B denota la clase de x de acuerdo a la relación indiscernible B. Los objetos que están contenidos en $B_*X$ son con seguridad miembros de X, mientras que los objetos contenidos en $B^*X$ son posibles miembros de X.

En el epígrafe 3.2 se presentan dos métodos de edición de conjuntos de entrenamiento, basados en las aproximaciones inferior y superior. Los resultados experimentales muestran un desarrollo satisfactorio del método k-NN usando estas técnicas.

# 3 Edición de conjuntos de entrenamiento usando la Teoría de los Conjuntos Aproximados

## 3.1. Acerca de la edición de conjuntos de entrenamiento

La selección de los ejemplos de un dominio a incluir en un conjunto de entrenamiento es un problema presente en todos los modelos computacionales que realizan inferencias a partir de ejemplos. El mismo se conoce como edición de conjuntos de entrenamiento.

La edición se hace con el objetivo de eliminar los ejemplos que inducen a una incorrecta clasificación, seleccionando un conjunto de referencia representativo y reducido. Las técnicas de Edición aunque también producen eliminación de ejemplos, tienen un objetivo fundamental: se persigue una muestra de entrenamiento de mejor calidad para tener mejor precisión con el sistema. Así, además de conseguir una reducción (en ocasiones notable) del conjunto de referencia proporcionan un conjunto de "calidad" que hace incrementar la tasa de acierto de los métodos de clasificación cuando se aplican sobre conjuntos editados y decrecer a su vez el costo computacional de la clasificación [Dev82], [Cor97].

Aha [Aha91] presentó una serie de algoritmos de aprendizaje basados en instancias. Los algoritmos de aprendizaje basado en instancias (Instance-based Learning, IBL) se basan en modelos ejemplos, cada concepto se representa por un conjunto de ejemplos, cada ejemplo puede ser una abstracción del concepto o una instancia indi-

vidual del concepto.

Brodley en 1993 [Bro93] introdujo un Modelo de Selección de Clase (MCS) sistema que usa un algoritmo de aprendizaje basado en instancias (que pretende ser aproximadamente basado en IB3) como la parte de un algoritmo más grande de aprendizaje híbrido. Tiende a evitar el ruido.

Skalak en 1994 [Ska94] usó el Random Mutation Hill Climbing para seleccionar los ejemplos a usar en S. Este método sólo resuelve parte del problema.

Cameron-Jones en 1995 [Cam95] usó una heurística de longitud de la codificación para determinar cuán bueno es el subconjunto S describiendo a T. Este algoritmo no es verdaderamente incremental, pero para distinguirlo bien de otras técnicas es llamado el método de Longitud de la Codificación Creciente (ELGrow).

En [Wil00], [Ain02], [Loz03] y [Kog04] aparecen varias con el propósito de reducir conjuntos de entrenamiento basadas en la teoría del vecino más cercano.

Existen varios algoritmos de edición de conjuntos de entrenamiento que se describen en [Bar02], los cuales están encaminadas a la detección y eliminación de patrones ruidosos y atípicos, con el objetivo de mejorar la exactitud de la clasificación. Algunas de ellos son ENN (Wilson, 1972), Todo k-NN (Tomek, 1976) y Algoritmo de Edición Generalizada (Koplowitz y Brown, 1978). Otro método de edición es el algoritmo Multiedit (Devijver y Kittler, 1980) reportado en [Dev82].

Recientemente se han reportado varios métodos de edición interesantes. Brighton y Mellish [Bri02] introdujeron el método de edición Iterative Case Filtering (ICF), el cual es basado en la regla de los Vecinos más Cercanos y el conjunto de asociaciones de un objeto *O*. En [San03] fueron introducidos tres métodos de edición: (i) Depuración, (ii) k-NCN (Nearest Centroid Neighborhood) y (iii) k-NCN iterativo. En [Jia04] fue propuesto el método NNEE (Neural Network Ensemble Editing). En [Olv05] se reportan dos esquemas de edición para reducir el tiempo de ejecución del método BSE. Los autores emplean ENN para filtrar ruidos. El Segundo esquema consiste en re-editar una muestra editada con el propósito de aumentar la precisión de la clasificación. En [Gar05] fue propuesto un nuevo método para la selección de ejemplos con datos mixtos incompletos en la descripción de los objetos (Mixed Incomplete Data (MID)), basado en una extensión de la regla del Vecino más Cercano.

### 3.2. Dos métodos para editar conjuntos de entrenamientos basados en la Teoría de los Conjuntos Aproximados

En la Teoría de los Conjuntos Aproximados resulta de gran interés el significado de la Aproximación Inferior de un sistema de decisión. En la Aproximación Inferior de cada clase de un sistema se agrupan aquellos objetos que con absoluta certeza pertenecen a su clase. Esto garantiza que los objetos contenidos en la Aproximación Inferior están exentos de ruidos.

En esta investigación se ha estudiado la aplicación de los conjuntos aproximados para la edición de conjuntos de entrenamiento. Se proponen dos métodos para editar los conjuntos de entrenamiento usando las aproximaciones inferior y superior fundamentalmente. Primero se usan las aproximaciones inferiores de las clases para crear el conjunto editado. En el segundo algoritmo se combinan las ideas del primero con

aplicar Edición Generalizada a la región límite de los conjuntos.

La idea básica de aplicar los conjuntos aproximados para editar los conjuntos de entrenamiento es la siguiente: en el conjunto de entrenamiento se colocan los ejemplos del sistema de decisión inicial que pertenecen a la aproximación inferior de cada clase, es decir, dado el dominio de una aplicación con m clases y la relación de equivalencia $B$, entonces $TS = B_*(D1) \cup B_*(D2) \cup ... \cup B_*(D_m)$. Esto es igual a decir que el conjunto de entrenamiento para el primer algoritmo será la región positiva del sistema de decisión. En esta forma, los objetos que están etiquetados incorrectamente o muy cerca de la frontera de decisión pueden ser eliminados del conjunto de entrenamiento, los cuales afectan la calidad de la inferencia. En la aproximación inferior de cada clase estarán aquellos objetos que con certeza pertenecen a dicha clase, por lo que de esta forma se garantiza la eliminación de cualquier presencia de ruido en el sistema de decisión.

Los estudios sobre la multiedición presentados en [Cor01] muestran que los objetos aislados incluidos en otras regiones o cerca de la frontera de decisión son eliminados frecuentemente. Por otra parte, estos objetos de la región límite serían muy interesantes si se pudiera re-etiquetar las clases a la cual pertenecen aquellas instancias incorrectamente etiquetadas, para lo cual se usan la Edición Generalizada, y con estas ideas se construye el segundo algoritmo que se propone.

**Algoritmo Edit1RS:**

**P1.** Construir el conjunto $B$, $B \subseteq A$. Preferiblemente, $B$ es un reducto del sistema de decisión.

**P2.** Formar los conjuntos $X_i \subseteq U$, tal que todos los elementos del universo $(U)$ que tienen valores $d_i$ en el atributo de decisión están en $X_i$.

**P3.** Para cada conjunto $X_i$, calcula su aproximación inferior $B_*(X_i)$.

**P4.** Construir el conjunto de entrenamiento editado SE como la unión de todos los conjuntos $B_*(X_i)$.

En este método Edit1RS sólo los elementos que están en las aproximaciones inferiores se toman en cuenta. También, es importante tomar en cuenta aquellos elementos que están en la frontera (BNB). El Algoritmo de Edición Generalizada consiste en remover algunos de los ejemplos sospechosos y cambiar las etiquetas de las clases de algunas instancias. Se puede considerar como una técnica para modificar la estructura de la muestra de entrenamiento (a través de re-etiquetar algunas de las instancias de entrenamiento) y no solamente para eliminar instancias atípicas [Bar02]. El segundo algoritmo está propuesto tomando en cuenta estas ideas.

**Algoritmo Edit2RS:**

**P1.** Construir el conjunto $B$, $B \subseteq A$. Preferiblemente, $B$ es un reducto del sistema de decisión.

**P2.** Formar los conjuntos $X_i \subseteq U$, tal que todos los elementos del universo $(U)$ que tienen valores $d_i$ en el atributo de decisión están en $X_i$.

**P3**. $S_E = \phi$.

**P4.** Para cada conjunto $X_i$:

i) Calcular su aproximación inferior $(B_*(X_i))$ y su aproximación superior $(B^*(X_i))$. $S_E = S_E \cup B_*(X_i)$.

ii) $T_i = B^*(X_i) - B_*(X_i)$.

**P5.** Calcular la unión de los conjuntos $T_i$. Se obtiene $T = \cup\ T_i$.

**P6.** Aplicar el método de Edición Generalizada a cada elemento en $T$ y el resultado es el conjunto $T'$.

**P7.** $S_E = S_E \cup T'$. El conjunto de entrenamiento editado se obtiene como el conjunto resultante en $S_E$.

La complejidad computacional de nuestros algoritmos no sobrepasa $O(ln^2)$, cercana al valor ideal de $O(n^2)$, mientras en el resto de los algoritmos (epígrafe 3.1) esta es de $O(n^3)$.

### 3.3. Resultados experimentales

Hemos estudiado el desempeño computacional de los algoritmos cuando ellos son empleados para editar la muestra que se usa en el clasificador k-NN. Se usaron sistemas de decisión construidos a partir de las bases de datos que fueron encontradas en: http://www.ics.uci.edu/~mlearn/MLRepository.html.

Los resultados que se muestran en las Tabla 1 y Tabla 2 reflejan una comparación de la precisión de la clasificación arrojada por el k-NN al emplear las bases de datos originales, sin editar, y las editadas usando varios métodos de edición reportados en la literatura por otros autores además de los dos nuevos métodos que se muestran en el epígrafe 3.2. Hemos considerado dos alternativas: (i) B contiene todos los rasgos, (ii) B es un reducto. Un reducto es un conjunto minimal de atributos contenido en A, que preserva la particionalidad del universo (permite diferenciar objetos de clases distintas) [Kom99].

Los resultados obtenidos por los métodos Edit1RS y Edit2RS fueron superiores a los obtenidos al clasificar el conjunto de entrenamiento sin editar, también los nuevos métodos aportaron similares resultados y en algunos de los casos superiores a los métodos ENN, Todo_KNN, Edición Generalizada y Multiedit. Se realizó una comparación entre estos dos nuevos métodos y los resultados de la clasificación para el método Edit2RS fueron superiores al Edit1RS. A partir de estos resultados se demostró además que al editar por los nuevos métodos la precisión en la clasificación es superior cuando se selecciona un reducto que cuando se trabaja con todos los rasgos.

Para comprobar eficientemente los resultados descritos con anterioridad se aplicó la prueba estadística de Validación Cruzada, para lo cual se dividió cada conjunto de datos en 5 muestras. A los resultados obtenidos de la prueba estadística de Validación Cruzada se les aplicó un Test de Student en el cual se obtuvo un valor de $P < 0.05$ para cada uno de los aspectos a demostrar.

**Tabla 1.** Precisión de la clasificación usando métodos clásicos

| **Nombre de la base de datos** (Cantidad de ejemplos, Cantidad de rasgos) | **Base de datos original** | **ENN** | **All k-NN** | **Edición Generalizada** | **MultiEdit** |
|---|---|---|---|---|---|
| Ballons-a (20,4) | 60.00 | 60.00 | 100 | 80.00 | 100 |
| Lirio (150,4) | 94.66 | 100 | 100 | 98.65 | 100 |
| Hayes-Roth (133,4) | 23.48 | 70.37 | 66.66 | 22.10 | 100 |
| Bupa (345,6) | 67.82 | 88.74 | 88.11 | 84.98 | 100 |
| E-Coli (336,7) | 86.60 | 98.28 | 98.06 | 97.70 | 100 |
| Corazón (270,13) | 82.22 | 97.30 | 98.40 | 96.10 | 100 |
| Diabetes (Pima) (768,8) | 73.04 | 94.32 | 96.63 | 90.43 | 100 |
| Cáncer de mama (683,9) | 96.77 | 99.24 | 99.69 | 99.26 | 100 |
| Levadura (1484,8) | 59.02 | 90.02 | 93.76 | 90.00 | 99.73 |
| Dermatología (358,34) | 97.48 | 98.56 | 100 | 99.14 | 100 |
| Cáncer pulmonar (27,56) | 48.14 | 50.00 | 75.00 | 55.55 | 0.00 |
| **Promedio** | 71.93 | 86.08 | 92.39 | 83.08 | 90.88 |

**Tabla 2.** Precisión de la clasificación usando métodos basados en los Conjuntos Aproximados

| **Nombre de la base de datos** (Cantidad de ejemplos, Cantidad de rasgos) | **Edit1RS** | | **Edit2RS** | |
|---|---|---|---|---|
| | **B= Reducto** | **B=Todos los rasgos** | **B= Reducto** | **B=Todos los rasgos** |
| Ballons-a (20,4) | 100 | 100 | 100 | 80.00 |
| Lirio (150,4) | 100 | 98.93 | 98.65 | 100 |
| Hayes-Roth (133,4) | 85.29 | 100 | 84.61 | 30.27 |
| Bupa (345,6) | 76.47 | 76.47 | 82.15 | 82.15 |
| E-Coli (336,7) | 91.89 | 61.53 | 95.70 | 96.49 |
| Corazón (270,13) | 95.41 | 89.54 | 93.65 | 92.36 |
| Diabetes (Pima) (768,8) | 80.00 | 80.00 | 90.36 | 90.36 |
| Cáncer de mama (683,9) | 98.01 | 98.27 | 97.65 | 97.35 |
| Levadura (1484,8) | 70.00 | 71.42 | 91.01 | 91.74 |
| Dermatología (358,34) | 93.46 | 98.78 | 95.01 | 98.26 |
| Cáncer pulmonar (27,56) | 77.77 | 48.14 | 72.22 | 48.14 |
| **Promedio** | 88.03 | 83.92 | 91.00 | 82.47 |
| | P= 90.00 | | P= 91.56 | |

P: promedio de los mejores resultados de los métodos Edit1RS y Edit2RS respectivamente, ya sea con todos los rasgos o con el reducto.

Los métodos de edición Edit1RS y Edit2RS arrojan mejores resultados en la clasificación que el conjunto de entrenamiento sin editar.

Los resultados obtenidos al clasificar los conjuntos de entrenamiento editados por

los métodos Edit1RS y Edit2RS fueron en la mayoría de los casos superiores a los editados por los métodos ENN, Todo_KNN, Edición Generalizada y Multiedit. El método Edit2RS obtuvo mejores resultados en la clasificación que el método Edit1RS. Para los métodos Edit1RS y Edit2RS los porcientos de efectividad en la clasificación resultaron mejores usando un reducto que trabajando con todos los rasgos.

Es importante destacar que los métodos propuestos en esta investigación se caracterizan por ser métodos muy simples y de fácil implementación, lo que los coloca en una posición ventajosa respecto a los ya existentes, teniendo en cuenta los resultados similares y en ocasiones superiores de los metodos Edit1RS y Edit2RS.

## 4 Conclusiones

En este artículo fue presentada la posibilidad de aplicar los elementos de la Teoría de los Conjuntos Aproximados para el análisis de datos cuando se usa el método k-NN.

Se propusieron dos métodos para la edición de conjuntos de entrenamiento. Los resultados experimentales muestran que es posible usar los Conjuntos Aproximados para construir conjuntos de entrenamiento editados con el fin de lograr un mejor desempeño del método k.NN. Nuestros métodos obtienen resultados similares y en ocasiones superiores al resto de los métodos. En consecuencia, pensamos que los nuevos métodos propuestos pueden ser tomados en cuenta para editar conjuntos de entrenamiento en el método k-NN. Los resultados obtenidos con los métodos Edit1RS y Edit2RS fueron superiores en la mayoría de los casos cuando B es un reducto.

Nuestros métodos son muy simples y de fácil implementación, lo que no ocurre exactamente con los métodos que se han reportado en la literatura.

## Referencias

[Aha91] Aha, David W. Kibler, Dennis. Marc K, Albert. Instance-Based Learning Algorithms. Machine Learning, 6, pp. 37-66. 1991.

[Ain02] Ainslie, M. C and Sánchez, J.S. Space Partitioning for Instance Reduction in Lazy Learning Algorithms. In 2nd Workshop on Integration and Collaboration Aspects of Data Mining, Decision Support and Meta-Learning, pages 13-18, 2002.

[Bar02] Barandela, Ricardo; Gasca, Eduardo, Alejo, Roberto. Correcting the Training Data. Published in "Pattern Recognition and String Matching", D. Chen and X. Cheng (eds.), Kluwer, 2002.

[Bri02] Brighton, H. and Mellish, C. Advances in Instance Selection for Instance-Based Learning Algorithms. Data Mining and Knowledge Discovery, 6, pp. 53-172, 2002.

[Bro93] Brodley, Carla E. Addressing the Selective Superiority Problem: Automatic Algorithm/Model Class Selection. Proceedings of the Tenth International Machine Learning Conference, Amherst, MA, pp. 17-24. 1993.

[Cam95] Cameron-Jones, R. M. Instance Selection by Encoding Length Heuristic with Random Mutation Hill Climbing. In Proceedings of the Eighth Australian Joint Conference on Artificial Intelligence, pp. 99-106. 1995.

[Cor01] Cortijo, J.B. Techniques of approximation II: Non parametric approximation. Thesis.

Department of Computer Science and Artificial Intelligence, Universidad de Granada, Spain. October 2001.

[Dev82] Devijver, P. and Kittler, J. Pattern Recognition: A Statistical Approach, Prentice Hall, 1982.

[Gar05] García, M. and Shulcloper, J. Selecting Prototypes in Mixed Incomplete Data. Lectures Notes in computer Science (LNCS 3773), pp. 450-460. Springer, Verlag, Berlin Heidelberg. New York. ISSN 0302-9743 ISBN 978-3-540-29850.

[Gre01] Greco, S. Et al. Rough sets theory for multicriteria decision analysis. European Journal of Operational Research 129, pp. 1-47, 2001.

[Jia04] Jiang Y., Zhou, Z.-H. Editing training data for kNN classifiers with neural network ensemble. In: Advances in Neural Networks, LNCS 3173, pp. 356-361, Springer-Verlag, 2004.

[Kog04] Koggalage, Ravindra and Halgamuge, Saman. Reducing the Number of Training Samples for Fast Support Vector Machine Classification. Vol. 2 No. 3, March 2004.

[Kom99] Komorowski, J. Pawlak, Z. et al.. Rough Sets: A tutorial. In Pal, S.K. and Skowron, A. (Eds) Rough Fuzzy Hybridization: A new trend in decision-making. Springer, pp. 3-98. 1999.

[Olv05] Olvera-López, José A., Carrasco-Ochoa, J. Ariel and Martínez-Trinidad, José Fco. Sequential Search for Decremental Edition. Proceedings of the 6th International Conference on Intelligent Data Engineering and Automated Learning, IDEAL 2005. Brisbane, Australia, vol 3578, pp. 280-285, LNCS Springer-Verlag, 2005.

[Paw82] Pawlak, Z.. Rough sets. International Journal of Information & Computer Sciences 11, 341-356, 1982.

[Pol02] Polkowski, L.. Rough sets: Mathematical foundations. Physica-Verlag, p. 574. Berlin, Germany. 2002.

[San03] Sánchez, J. S., Barandela, R., Marqués, A. I., Alejo, R., Badenas, J. Analysis of new techniques to obtain quality training sets. Pattern Recognition Letters, 24-7, pp. 1015-1022, 2003.

[Ska94] Skalak, D. B. Prototype and Feature Selection by Sampling and Random Mutation Hill Climbing Algorithms. In Proceedings of the Eleventh International Conference on Machine Learning (ML94). Morgan Kaufmann, pp. 293-301. 1994.

[Wil00] Wilson, Randall and Martinez, Tony R. Reduction Techniques for Instance-Based Learning Algorithms. Machine Learning, 38:257-286. Computer Science Department, Brigham Young University. USA 2000.

# Ajuste dinámico de profundidad en el algoritmo $\alpha\beta$ ($DDA\alpha\beta$)

Daniel Micol[1] y Pablo Suau[2]

[1] Escuela Politécnica Superior, Universidad de Alicante
San Vicente del Raspeig, 03690 Alicante
dmp18@alu.ua.es
[2] Dpto. de Ciencia de la Computación e Inteligencia Artificial
Universidad de Alicante
San Vicente del Raspeig, 03080 Alicante
pablo@dccia.ua.es

**Resumen** Desde los inicios de la teoría de computación en juegos, diversas técnicas han sido desarrolladas con el fin de mejorar la calidad de juego de la máquina a la vez que se reduce el tiempo de respuesta. Uno de los algoritmos más extendidos para la búsqueda en juegos es el conocido como $\alpha\beta$, el cual supone una variante respecto del algoritmo Minimax al introducir podas en el árbol de decisión. Actualmente hay numerosas mejoras aplicables a dicho algoritmo, aunque todas ellas con el mismo inconveniente: poseen un alto grado de incertidumbre y pueden acarrear pérdidas de rendimiento o cálculos erróneos. Proponemos una técnica basada en un ajuste dinámico del nivel de profundidad según el estado de la partida que supone una mejora fiable para el algoritmo $\alpha\beta$. Dicha técnica, a la que hemos dado el nombre de *Dynamic Depth Adjustment on the $\alpha\beta$ algorithm ($DDA\alpha\beta$)*, nunca podrá desembocar en pérdidas de rendimiento debido a que está basada en la suposición del peor caso del algoritmo. La experimentación aportada muestra una mejora de rendimiento y fiabilidad de los resultados en comparación con otros métodos. Además, existe la posibilidad de ser utilizado en combinación con otras técnicas.

*Palabras clave:* inteligencia artificial, búsqueda en juegos, algoritmos heurísticos

## 1. Introducción

El algoritmo $\alpha\beta$ es el método más extendido a la hora de realizar búsquedas en árboles de decisión aplicados a juegos formados a partir de un conjunto de elementos con ordenación total [4]. El objetivo del mismo es obtener la mejor jugada para la máquina, tratando de predecir las respuestas del usuario a sus movimientos para así conseguir la victoria en la partida. Está basado en el algoritmo Minimax, diferenciándose de éste en la inserción de operaciones de poda,

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 646-654, 2006.

lo cual reduce el número de nodos a evaluar en los árboles de decisión [2]. En el algoritmo $\alpha\beta$ dicho número depende de dos factores: las podas realizadas en el árbol, y la profundidad máxima del mismo.

El primero de estos factores no se puede conocer a priori, sino tan sólo después de haber completado el proceso. Existen técnicas para incrementar el número de podas, como es el caso de la profundización iterativa [3], tablas de refutación [1], búsquedas usando ventanas mínimas [5] y heurísticas [7], pero la mayoría posee cierto grado de incertidumbre que puede producir pérdidas de rendimiento [6]. Todos estos métodos reducen el tiempo de ejecución del algoritmo $\alpha\beta$, descartando, mediante heurísticas adecuadas, ramas que son consideradas poco relevantes para la obtención de la solución. Sin embargo, estas heurísticas pueden ser demasiado restrictivas, por lo que la búsqueda podría dejar de ser admisible.

Por otro lado, la profundidad máxima del árbol de decisión también determinará la cantidad de nodos hoja a evaluar. Conforme avanza una partida, el número de posibles movimientos se reducirá, y también lo hará la cantidad de nodos evaluables en el árbol de decisión. Basándonos en este hecho, proponemos aplicar técnicas que aprovechen la mencionada reducción de nodos hoja para incrementar el nivel de profundidad máximo del árbol y así poder obtener un mayor nivel de juego sin aumento del tiempo de respuesta de la máquina.

Este artículo se compone de una primera parte en la que introduciremos la técnica propuesta. En la segunda se explicará el método diseñado, y su aplicación a un juego en concreto será descrita en la tercera sección. Por último mostraremos los resultados en la cuarta parte, así como las conclusiones y trabajos futuros en la quinta y última.

## 2. Ajuste dinámico de profundidad

Por definición, el número máximo de nodos hoja en un árbol $\alpha\beta$ se corresponderá con la cantidad de posibles combinaciones de jugadas realizadas desde el estado actual de la partida hasta el nivel de profundidad máximo establecido a priori para dicho algoritmo [2]. Por lo general, dicho valor de profundidad se define al principio de la partida y es invariante durante el transcurso de ésta. No obstante, conforme el estado de una partida avanza, el número de posibles combinaciones de jugadas disminuye, y por lo tanto también lo hace el número de nodos hoja del árbol $\alpha\beta$. Puesto que el tiempo de respuesta del algoritmo depende en gran medida de la cantidad de nodos que son evaluados, se deduce que incrementando el nivel de profundidad conforme avanza la partida se podrá mantener constante la cantidad de nodos evaluados en cada turno de la máquina, siendo dicha cifra similar a la del inicio de la partida. Esto permitirá incrementar el nivel de profundidad sin aumento del tiempo de respuesta respecto al principio de la partida, lo que se traduce en un incremento de la calidad de juego de la máquina.

Tomaremos como base para nuestro método un juego teórico caracterizado por dos jugadores que juegan por turnos y en el que ambos intentan ganar, no existiendo el azar. Para resolver dicho juego se aplica el algoritmo $\alpha\beta$ y

suponemos que no hay límites de jugadas de un determinado tipo, es decir, que nunca hay poda de una rama por imposibilidad de jugar en esa posición. Hay que destacar que, a priori, el número de podas del algoritmo $\alpha\beta$ resulta impredecible, y la única forma de conocerlo es recorrer y evaluar los nodos de dicho árbol. Por lo tanto, y puesto que estamos realizando una formalización matemática, partiremos de la suposición de que no se realizará ninguna poda. Por esta razón, en cada nivel del árbol se genera un número exponencial de nodos cuyo factor de exponenciación es constante para todo el árbol y coincide con las posibilidades de movimiento.

Sea $\omega$ el nivel de profundidad en el estado inicial de la partida, $\psi$ el nivel de profundidad en el estado actual de la misma, $N_\omega$ el número de nodos hoja en el estado inicial de la partida y $N_\psi$ el número de nodos hoja en el estado actual de ésta, se deduce que el número de posibilidades de movimiento en el nivel inicial será $\sqrt[\omega]{N_\psi}$. Por lo tanto, suponemos que $\exists \Delta\omega / \sqrt[\omega]{N_\psi}^{(\omega+\Delta\omega)} = N_\omega \wedge \Delta\omega \in \mathbb{N}$, y podemos deducir lo siguiente:

$$\begin{aligned}
\sqrt[\omega]{N_\psi}^{(\omega+\Delta\omega)} &= N_\omega \\
\Rightarrow N_\psi^{\frac{(\omega+\Delta\omega)}{\omega}} &= N_\omega \\
\Rightarrow \log_{N_\psi} N_\psi^{\frac{(N+\Delta\omega)}{\omega}} &= \log_{N_\psi} N_\omega \\
\Rightarrow \frac{\omega+\Delta\omega}{\omega} \log_{N_\psi} N_\psi &= \log_{N_\psi} N_\omega \qquad (1) \\
\Rightarrow \frac{\omega+\Delta\omega}{\omega} &= \log_{N_\psi} N_\omega \\
\Rightarrow \omega+\Delta\omega &= \omega \log_{N_\psi} N_\omega \\
\Rightarrow \omega+\Delta\omega &= \omega \frac{\ln N_\omega}{\ln N_\psi}
\end{aligned}$$

Finalmente, y puesto que el nivel de profundidad será un número natural, tenemos que:

$$\psi = \omega + \Delta\omega \simeq \left[\omega \frac{\ln N_\omega}{\ln N_\psi}\right] \qquad (2)$$

Como se puede observar, el nivel de profundidad en el estado actual de la partida siempre será mayor o igual que en el estado inicial.

No obstante, el método descrito posee un inconveniente, ya que es incapaz de predecir el número de nodos hoja del árbol generado por el algoritmo $\alpha\beta$, por lo que, aun siendo capaz de proporcionar un nivel de profundidad, éste siempre será calculado para el peor caso del algoritmo desperdiciándose parte de la capacidad computacional de la máquina. Para solventar este problema se propone a continuación un método alternativo y se muestra cómo podría ser aplicado a un juego concreto.

## 3. El ajuste dinámico en el juego del *Conecta Cuatro*

En un juego real, las posibilidades de movimiento suelen estar limitadas en algún estado de la partida. Por ejemplo, en el ajedrez un peón no podrá avanzar si existe otra ficha en la siguiente fila y misma columna. Otro ejemplo es el *Conecta Cuatro*[1], donde no se podrá colocar una ficha en una columna que ya esté completa. En estos casos es posible realizar una aproximación más fiable que la descrita anteriormente mediante el cálculo de todas las posibilidades de juego, esto es, del número de nodos hoja que poseerá el algoritmo $\alpha\beta$ sin contar las podas, pero teniendo en cuenta las limitaciones mencionadas.

En algunas ocasiones es posible encontrar una expresión que permita calcular el número de nodos hoja de un árbol $\alpha\beta$ según el estado actual de la partida, pero en la mayor parte de los casos la obtención de dicha expresión no resulta trivial. Para aquellos en los que la expresión para el cálculo del número de nodos hoja sea compleja, es aconsejable realizar una modificación que simule el comportamiento del algoritmo $\alpha\beta$ pero sin evaluaciones. Esto proporcionará una aproximación bastante fiable del número de nodos final, con el acarreo de tiempo de ejecución correspondiente.

A continuación nos centraremos en el caso del juego del *Conecta Cuatro* para mostrar una variante simplificada del método propuesto. Ésta consiste en tener en cuenta tan sólo las columnas no llenas del tablero, sin considerar cuantas fichas se podrán depositar en cada una de ellas. Sea $\tau$ el número de columnas del tablero, $\xi$ el número de columnas libres, es decir, el número de posibles columnas en las que podremos poner una ficha, y $\omega$ el nivel de profundidad inicial, se deduce que $\tau^{\omega}$ es el número máximo de nodos hoja del árbol $\alpha\beta$, esto es, para el peor caso del algoritmo. Conforme se vayan insertando fichas en el tablero, $\xi$ irá decreciendo. Por lo tanto, al avanzar la partida, sería posible aumentar el nivel del árbol evaluando como máximo el mismo número de nodos que en el estado inicial de la partida. Es decir, si conocemos $\tau^{\omega}$, deberíamos encontrar un valor de $\Delta\omega$ tal que $\tau^{\omega} \geq \xi^{\omega+\Delta\omega}$. El valor de $\Delta\omega$ se puede extraer de forma aproximada tal como se muestra a continuación:

$$\begin{aligned} & \log_{\xi}(\tau^{\omega}) \geq \log_{\xi}(\xi^{\omega+\Delta\omega}) \\ \Rightarrow\ & \log_{\xi}(\tau^{\omega}) \geq (\omega+\Delta\omega)\log_{\xi}\xi \\ \Rightarrow\ & \log_{\xi}(\tau^{\omega}) \geq \omega+\Delta\omega \end{aligned} \tag{3}$$

Debido a que el nivel de profundidad es un número entero, es necesario realizar una aproximación:

[1] Las características y reglas de este juego pueden consultarse en múltiples páginas en Internet, como por ejemplo http://en.wikipedia.org/wiki/Connect_four.

$$\begin{cases} \log_{\xi}(\tau^{\omega}) \geq \omega + \Delta\omega \\ \psi = \omega + \Delta\omega \end{cases}$$
$$\Rightarrow \psi \leq \left[\log_{\xi}(\tau^{\omega})\right] \tag{4}$$
$$\Rightarrow \psi \leq \left[\frac{\ln(\tau^{\omega})}{\ln \xi}\right]$$

A partir del razonamiento anterior, se puede deducir, en cada turno de la máquina, cuánto se podría incrementar la profundidad del árbol de juego sin afectar al rendimiento. Este nuevo nivel se obtendrá a partir del factor de ramificación actual y el inicial del nodo raíz.

Esta técnica propuesta es aplicable a juegos en las que hay pocas restricciones de movimiento al igual que en el *Conecta Cuatro*. No obstante, para aplicarlo a otros juegos en los que las jugadas estén más restringidas (como por ejemplo el ajedrez o el otelo, juegos en los que no se puede colocar una ficha en cualquier posición del tablero), habría que añadir un factor que tuviera en cuenta dichas restricciones.

## 4. Experimentación y resultados

Todas las pruebas presentadas a continuación fueron realizadas sobre el juego del *Conecta Cuatro* para diferentes dimensiones del tablero. Primero se expondrán los resultados para el método general descrito en el segundo apartado, y posteriormente para el específico de dicho juego desarrollado en el tercer apartado. Las pruebas fueron realizadas mediante una aplicación programada en lenguaje Java, compilada usando JDK 1.4.2 y ejecutada en un Apple iMac G5 a 2 GHz, 512 MB de RAM y Mac OS X Tiger Versión 10.4.4.

En esta sección se muestran dos tipos de resultados, relacionando el nivel de profundidad con el número de fichas en el tablero, y este último valor con el tiempo de respuesta en milisegundos. El número de fichas en el tablero se corresponde con los movimientos ya realizados, y proporciona una idea de cual es el estado de la partida. Se supone que el juego finaliza cuando se llena el tablero, ya que para obtener un mayor número de medidas hemos forzado un empate entre los dos jugadores del *Conecta Cuatro*, no deteniendo la partida cuando se produjera un cuatro en raya. Además, hay que destacar que el valor del eje de coordenadas de las gráficas satura en 100 para los que muestran el nivel de profundidad y en 50 para los que hacen lo propio con el tiempo de respuesta. Esto es debido a que en el primer caso interesa observar la evolución de la profundidad máxima hasta el final de la partida, pero en cuanto al tiempo de respuesta, llega un momento en el que se estabiliza en un valor muy bajo cercano a cero, por lo que su representación a partir de este punto no aporta datos útiles.

El primer experimento realizado consistió en aplicar la versión general de la técnica propuesta a tableros de diferentes dimensiones para observar su comportamiento en situaciones variadas. Para ello, se optó por emplear tableros de juego cuadrados[2] de dimensiones 7, 8, 9 y 10 respectivamente. Los resultados muestran un notable incremento del nivel de profundidad, tal como muestra la Figura 1.

El incremento mencionado fue de una magnitud similar para todas las dimensiones de tablero, pero se produjeron en estados más avanzados de la partida conforme se aumentó el tamaño del mismo. Esto puede provocar que el aprovechamiento de la mejora producida por el método $DDA\alpha\beta$ ocurra cuando la partida ya esté prácticamente decidida y su aplicación sea de poca utilidad. En estos casos habrá que encontrar algún factor que haga incrementar con anterioridad el nivel de profundidad tratando de no comprometer el tiempo de respuesta de la máquina.

A continuación se analizan los tiempos de respuesta para los casos representados en la Figura 1. Tal como se muestra en la Figura 2, dichos tiempos no se vieron afectados significativamente por el incremento de profundidad producido, siendo siempre inferiores a los obtenidos con el nivel inicial de profundidad, por lo que el rendimiento del algoritmo $\alpha\beta$ no se ve perjudicado.

Se puede observar un incremento de los tiempos de ejecución de cada jugada hasta el turno número 25 aproximadamente, por lo que este incremento no es producido por el de la profundidad, ya que éste se produce posteriormente. Además, es posible darse cuenta de que, tal como se afirma en las secciones anteriores, el aumento de la profundidad máximo no repercute negativamente en el tiempo de respuesta de cada movimiento de la máquina.

Posteriormente se repitió la misma experimentación aplicando el método específico descrito para el juego del *Conecta Cuatro*. Su mayor facilidad de diseño e implementación tiene como contrapartida el hecho de que produce los incrementos de profundidad más tardíamente que el método general. Esto queda reflejado en la Figura 3, donde se observa además que los niveles máximos obtenidos tras la aplicación del método específico serán mayores que los producidos por el general.

Por último, cabe destacar que el incremento en cuestión sólo se producirá cuando se llene una columna, hecho que es muy variable, por lo que podrá producir mejoras casi imperceptibles según evolucione la partida.

Al igual que en el método general, en el específico el aumento de niveles de profundidad no afecta negativamente al rendimiento del algoritmo $\alpha\beta$. Esto queda reflejado en la Figura 4, donde se observa cómo a partir de los puntos de incremento de niveles representados en la Figura 3 no se produce un aumento del tiempo de ejecución, por lo que la eficiencia del algoritmo no queda comprometida.

Los resultados de la experimentación demuestran que en la mayor parte de los casos el algoritmo $DDA\alpha\beta$ específico aplicado al juego del *Conecta Cuatro* utiliza niveles de profundidad mayores que el general, aunque serán muy pocas

[2] Un tablero cuadrado es aquel que posee el mismo número de filas que de columnas.

las jugadas en las que se explote este incremento de nivel ya que se produce muy tardíamente. Además, el método general posee un comportamiento más estable que permite el uso de un nivel de profundidad alto del árbol durante un mayor número de jugadas, aunque no se llegue a una profundidad tan elevada como al aplicar el método específico. En ambos casos el tiempo de ejecución fue muy similar al que se hubiera obtenido si no se hubiera aumentado el nivel del árbol de forma dinámica.

## 5. Conclusiones y trabajo futuro

Los resultados obtenidos muestran que es posible incrementar de forma dinámica el nivel de profundidad del algoritmo $\alpha\beta$ de una forma segura y sin que conlleve incrementos en el tiempo de respuesta mediante la suposición del peor caso. Este incremento ayudará a la máquina a contemplar un mayor número de futuras jugadas, mejorando su forma de jugar. Además, es posible realizar cálculos estadísticos del nivel de podas medio de un árbol $\alpha\beta$ para reducir el número teórico de nodos a calcular y así realizar incrementos mayores del nivel de profundidad, con el riesgo que ello conlleva si las suposiciones no son ciertas.

El método general propuesto puede ser complejo de calcular e implementar para ciertos juegos con muchas posibilidades de movimiento por ficha, como es el caso del ajedrez. Para ello, se pueden utilizar variantes más sencillas de dicha técnica aplicadas a juegos concretos con la consiguiente pérdida de efectividad. Además, dicho método se basa en el peor caso del algoritmo ya que es incapaz de predecir el número de podas, por lo que en ocasiones no se alcanzará el máximo nivel de profundidad posible para un intervalo de tiempos de respuesta permitido.

Trabajos futuros pueden estar relacionados con cálculos estadísticos para predicciones sobre el número de nodos que evaluará un árbol $\alpha\beta$ para una aproximación más óptima del nuevo nivel de profundidad. Además, se podrán estudiar diversos juegos de mayor complejidad como es el caso del ajedrez para observar el comportamiento del método propuesto.

## Referencias

1. S.G. Akl and M.M. Newborn. The principle continuation and the killer heuristic. pages 466–473, 1977.
2. F. Escolano, M.Á. Cazorla, M.I. Alfonso, O. Colomina, and M.A. Lozano. *Inteligencia Artificial: Modelos, Técnicas y Áreas de Aplicación*. Thomson, 2003.
3. J. Gillogly. *Performance Analysis of the Technology Chess Program*. PhD thesis, Department of Computer Science, Carnegie-Mellon University, 1978.
4. M.L. Ginsberg and A. Jaffray. Alpha-beta pruning under partial orders. *MSRI Publications*, 42:37–48, 2002.
5. J. Pearl. Scout: A simple game-searching algorithm with proven optimal properties. Stanford, 1980. First Annual National Conference on Artificial Intelligence.
6. R.L. Rivest. Game tree searching by min/max approximation. *Artificial Intelligence*, 34:77–96, December 1987.

7. J. Schaeffer. The history heuristic and alpha-beta search enhancements in practice. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-11:1203–1212, November 1989.

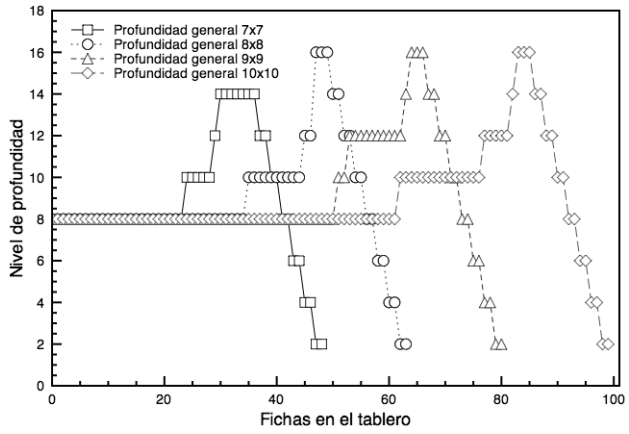



**Figura 1.** Evolución del nivel de profundidad usando el método general de $DDA\alpha\beta$.

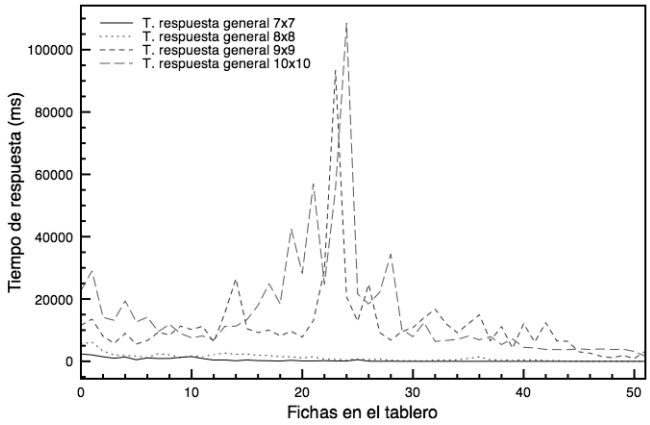



**Figura 2.** Evolución del tiempo de respuesta usando el método general de $DDA\alpha\beta$.

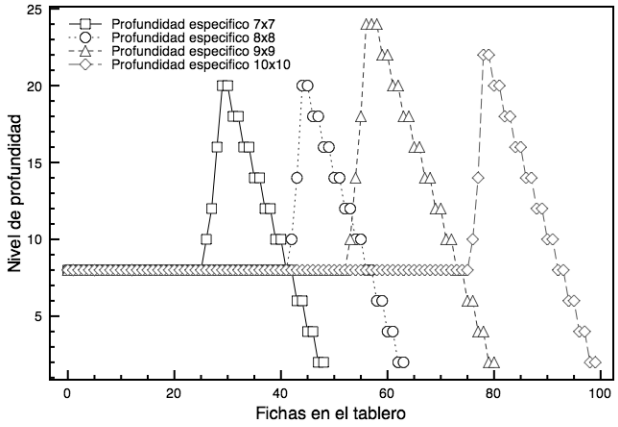



**Figura 3.** Evolución del nivel de profundidad usando el método específico de $DDA\alpha\beta$.

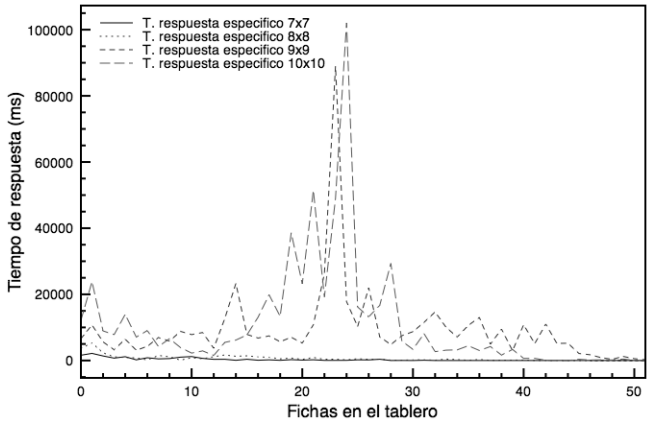



**Figura 4.** Evolución del tiempo de respuesta usando el método específico de $DDA\alpha\beta$.

# TRADINNOVA: Un algoritmo heurístico de compra-venta inteligente de acciones

I.J. Casanova y J.M. Cadenas

Dpto. Ingeniería de la Información y las Comunicaciones.
Facultad de Informática. Universidad de Murcia. Spain.
isidoroj@um.es, jcadenas@um.es

**Resumen** Existen diferentes técnicas que nos aconsejan como invertir en bolsa, desde la compra de una simple acción a la elección de una cartera de acciones que dé la máxima rentabilidad con el menor riesgo posible, aunque todas estás técnicas no nos dicen nada de cuando vender las acciones compradas para probablemente comprar otras acciones diferentes.
En este trabajo se propone y presenta un algoritmo heurístico (TRADINNOVA), que realiza la compra-venta inteligente de acciones durante un periodo de tiempo. La elección de la acción a comprar se podrá realizar mediante diversas técnicas heurísticas según la preferencia del inversor. Se ha realizado una simulación en el Mercado Continuo Español ejecutando TRADINNOVA entre los años 2001 y 2004.

*Palabras clave:* Inversión Bursátil, Heurísticas, Toma de Decisiones, Sistemas Inteligentes, Resolución de Problemas.

## 1. Introducción

En un día cualquiera en un mercado bursátil se pueden seleccionar un conjunto de acciones de las que se espera obtener una buena rentabilidad. Las técnicas para seleccionar estas acciones pueden ser muy diversas, desde el análisis técnico o el análisis fundamental hasta técnicas de inteligencia artificial como las redes neuronales, redes bayesianas, análisis de noticias, análisis del clima del mercado, etc.

Una vez que se han seleccionado estas acciones tendríamos que aplicar una política de compra-venta de acciones para obtener la mayor rentabilidad posible. Así, cada día de cotización se tendrá que escoger la mejor acción a comprar, su precio de compra, cuando se va a vender y su precio de venta. También se tendrá que realizar un seguimiento de las órdenes de compra o de venta que no se ejecuten para actuar sobre ellas. Para realizar todo esto, proponemos un algoritmo heurístico, que llamaremos TRADINNOVA, con el objetivo de simular el comportamiento de una forma inteligente de un inversor en el mercado continuo aplicando unas reglas para realizar la compra-venta de las acciones. Se ha probado el algoritmo y hemos realizado una simulación en la que se tiene

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 655-666, 2006.

en cuenta el precio histórico de la acciones, y se puede comprobar como en la mayoría de los casos se obtiene una revalorización mucho mayor que el índice de referencia (IBEX35).

En este trabajo pretendemos presentar este algoritmo heurístico para la compra-venta de acciones. Para ello, el trabajo lo hemos organizado de la siguiente manera: primero, haremos una introducción al campo de la inversión bursátil, comentando las técnicas que se suelen utilizar. En la sección 3, presentamos un sistema para la toma de decisiones para la compra-venta de acciones que nos ha llevado a definir el algoritmo TRADINNOVA. En la sección 4 se definen los parámetros y se presenta la simulación realizada con el algoritmo y los resultados obtenidos. Por último, acabamos con las conclusiones y vías futuras de trabajo.

## 2. La inversión bursátil

### 2.1. Técnicas de predicción

Lo que se persigue con la inversión bursátil es obtener el mayor beneficio posible. Para ello existen muchas técnicas que nos intentan decir si una acción esta valorada cara o barata (análisis fundamental) o si su precio esta en una tendencia alcista o bajista (análisis técnico).

Los precios de una acción se pueden representar en el tiempo. Lo ideal sería comprar al comienzo de una tendencia alcista, para vender en lo más alto posible, justo antes de que empiece una tendencia bajista. Estos momentos de compra (c) y venta (v), se pueden ver en la Fig. 1.

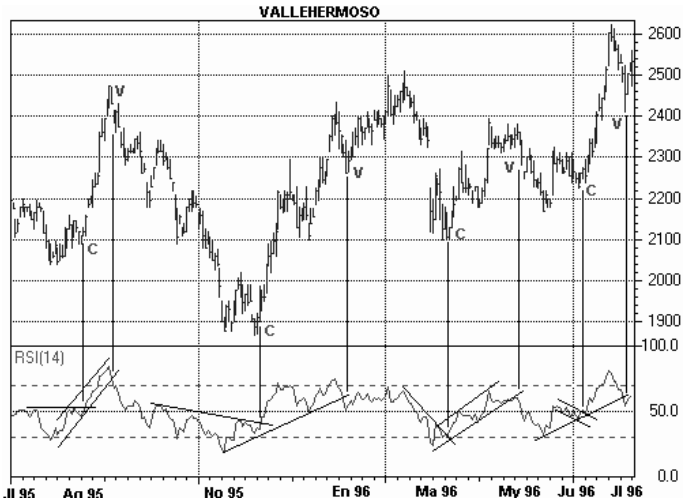



**Figura 1.** Señales de compra y venta del chart de Vallehermoso

Existen múltiples estudios de inteligencia artificial para predecir el precio de una determinada acción, siendo las redes neuronales una de las técnicas más utilizadas ([1,2,3]). También se utilizan los algoritmos genéticos para optimizar los parámetros de estas redes neuronales o para optimizar el análisis técnico de una acción ([4,5]).

Asimismo existen estudios de como predecir el comportamiento de una acción a partir de las noticias publicadas en Internet ([7,8]).

Si nos atenemos a [6], las técnicas para aplicar en finanzas se pueden dividir en 5 categorías (Fig. 2).

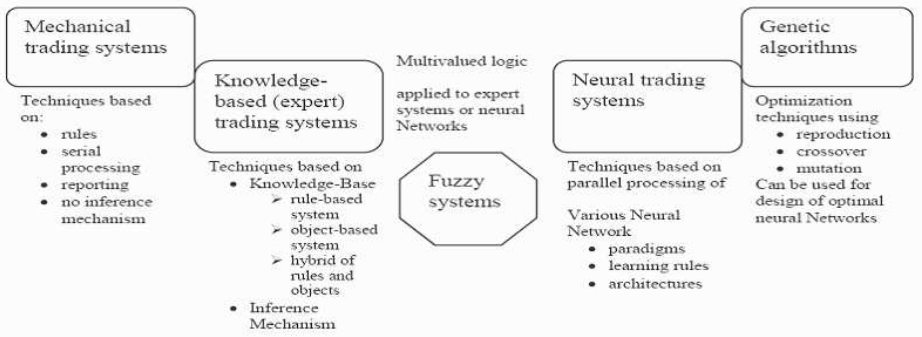



**Figura 2.** Clasificación de las técnicas para aplicar en finanzas

Todas las técnicas vistas hasta ahora lo que hacen es aconsejarnos sobre la compra de una acción. Pero una vez que se predice el comportamiento de una acción, habría que aplicar una política de compras y ventas sobre esa acción para maximizar el beneficio, siendo esto último un tema que hasta ahora no se ha estudiado a fondo.

### 2.2. Cartera de valores

Lo normal para realizar una predicción es fijarse en una sola acción, pero realmente un inversor no se debe fijar solamente en una acción, sino en todas las acciones que conforman un mercado concreto (que en nuestro caso sería el Mercado Continuo Español), o incluso fijarse en las acciones de otros países, para así formar una cartera de valores.

Al formar una cartera de acciones, se puede tener en cuenta la aversión al riesgo que tiene el inversor, así siguiendo la teoría de Markowitz, el inversor querrá obtener la máxima rentabilidad con el menor riesgo posible ([14,15,16]).

**Selección de la cartera de valores.** Se pueden utilizar diferentes técnicas para seleccionar las acciones (redes neuronales, análisis técnico, análisis fundamental, reglas, redes bayesianas, análisis de las noticias, análisis del clima del

mercado, etc) que van a formar la cartera de valores. Se puede ver una pequeña comparación en [11].

Una vez que se han seleccionado las acciones, el momento de su compra-venta no esta claramente especificado en ninguno de los trabajos realizados hasta ahora, así existen trabajos en los que se muestra en tiempo real un aviso en pantalla sobre que acción o grupo de acciones se recomiendan ([12]), en lo que se compran y venden las acciones anualmente ([10]), en los que se estima si va a ser una compra a largo o corto plazo ([9]), o en los que simplemente no se dice nada de que metodología se ha aplicado para estimar cuando se deben realizar estas compra-ventas.

En [17] se propone un algoritmo bastante interesante de rebalanceo de una cartera de acciones mediante compras y ventas, llamado *ANTICOR*, en el que se calcula para cada día el peso que deberían tener las acciones en la cartera. Este algoritmo tiene el inconveniente de que no se hace una verdadera simulación de las compras o de las ventas, ya que se rebalancea sin dar órdenes de compra o venta o sin tener en cuenta las comisiones aplicadas, y no se deja muy claro de que forma se seleccionan las acciones o en que momento y por qué hay que realizar su venta.

## 3. Política de compra-venta de acciones: Algoritmo TRADINNOVA

En este trabajo se propone un sistema para la toma de decisiones en la compra-venta (Fig. 3) a aplicar a las acciones que conforman cualquier mercado bursátil durante un periodo de tiempo.

Si suponemos un mercado de valores que cotiza desde una fecha inicial ($f_{ini}$) hasta una fecha final ($f_{fin}$), cada uno de esos días, todas sus acciones habrán tenido un precio de apertura ($p_{ape}$), un precio máximo ($p_{max}$), un precio mínimo ($p_{min}$) y un precio de cierre ($p_{cie}$).

Si cogemos un día $d$ cualquiera entre $f_{ini}$ y $f_{fin}$ en el que no se tiene ninguna acción comprada, entonces podremos seleccionar un conjunto de $n$ acciones ($S_d$), que utilizando cualquiera de las técnicas comentadas anteriormente (redes neuronales, análisis técnico, análisis fundamental, reglas, ...) serán las más recomendables comprar, porque se espera de ellas que den una buena rentabilidad.

Dependiendo de la técnica heurística elegida se seleccionarán las acciones de acuerdo a los objetivos particulares de cada inversor. Así, el funcionamiento de cada una de estas técnicas se podrá categorizar atendiendo a diferentes criterios, tales como rentabilidad, volatilidad, riesgo, etc.

La técnica heurística utilizada deberá calcular un valor para cada una de las acciones que conforman el mercado en ese día $d$, cuantificando lo recomendable que sería comprar o mantener en cartera esas acciones. Las acciones se ordenaran de mayor a menor de acuerdo a este valor y se tendrá que elegir el conjunto de las "mejores acciones", debiéndose definir hasta que valor una acción se consideraría que pertenece a este conjunto de las mejores, pudiéndose dar el caso que algún

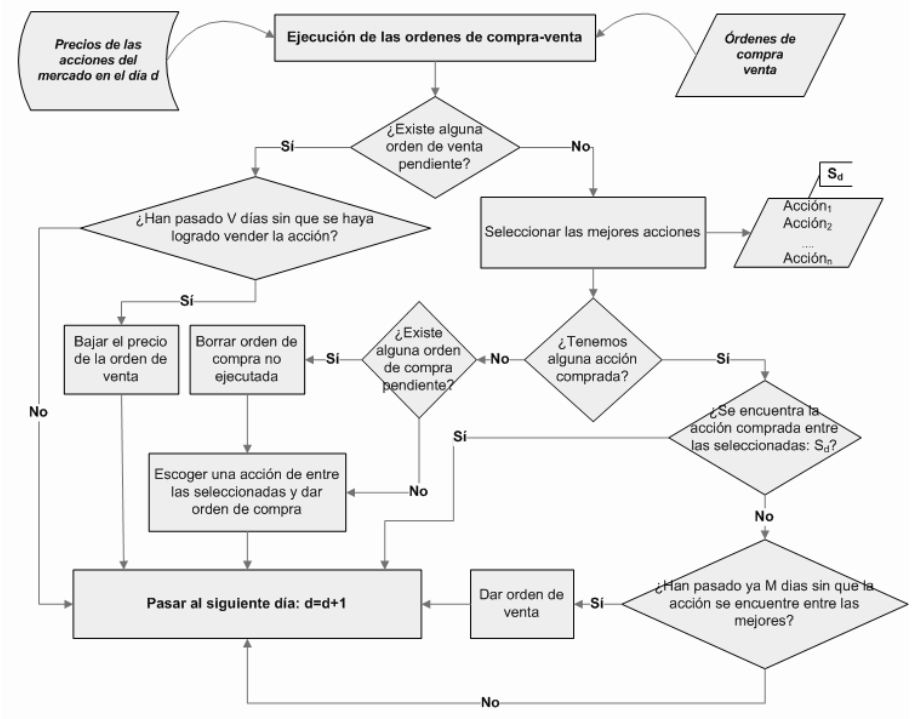



**Figura 3.** Toma de Decisiones para la compra-venta para un día d cualquiera

día no se recomiende ninguna acción ($n = 0$), o bien que se recomienden muchas porque su análisis ha sido lo suficientemente satisfactorio para todas ellas.

Una vez tenemos este conjunto $S_d$ con la selección de las mejores acciones para un día $d$, se podrían intentar comprar todas o algunas de estas acciones. Para simplificar el algoritmo se va a intentar comprar solamente una de estas acciones seleccionadas. Para escoger la acción a comprar tendríamos dos posibilidades, seleccionar la mejor o bien seleccionar cualquiera aleatoriamente.

Una vez que hemos escogido una acción, tendremos que dar una orden de compra para el siguiente día. El precio de compra lo limitaremos a cualquiera de los precios que ha tenido la acción durante ese mismo día (entre $p_{min}$ y $p_{max}$), o bien se podrá comprar al precio que marque la acción en la apertura del día siguiente ($p_{ape}$).

Si ponemos un precio de compra bajo, hay mas posibilidades de que no se llegue a ejecutar la compra en los días siguientes, pero a cambio la acción se habrá comprado mas barata.

Al día siguiente ($d + 1$) se ve si se puede ejecutar la compra de la acción, siempre y cuando el precio de compra se encuentre entre el rango de precios que tiene la acción en ese día (entre $p_{min}$ y $p_{max}$). Para este nuevo día se vuelven

a seleccionar las mejores acciones, y si ya tenemos una acción comprada, entonces comprobamos si la acción comprada se sigue encontrando entre las mejores ($S_{d+1}$), en cuyo caso no haríamos nada o en caso contrario dejaríamos que pasaran $M$ días sin que la acción se encontrara entre las mejores para dar una orden de venta.

Si no tenemos ninguna acción comprada porque no se ha podido ejecutar la orden de compra, entonces eliminamos esta orden de compra, y daríamos una nueva orden de compra, seleccionando una acción de entre las mejores ($S_{d+1}$) de ese día.

Una orden de venta se da cuando han pasado $M$ días sin que la acción se encuentre entre las mejores en esos días. Una orden de venta tendría también un precio limitado, al igual que la orden de compra. Cuanto mas alto sea este precio, mas difícil será que se ejecute. En caso de que durante $V$ días no se logre vender a este precio, porque seguramente la acción se encuentre en un periodo bajista, entonces se bajaría el precio de esta orden de venta.

A partir de la lógica de compra-venta explicada, proponemos el siguiente algoritmo heurístico, que denominaremos TRADINNOVA, que simula el comportamiento de un inversor en el mercado continuo:

```
FUNCTION TRADINNOVA (fecha_desde, fecha_hasta, tipo_seleccion,
                     dinero_inicial): dinero_final;
BEGIN
   fecha=fecha_desde;
   dinero=dinero_inicial);
   WHILE (fecha <= fecha_hasta)
   BEGIN
      dinero=dinero+Ejecutar_Ordenes_Venta_Pendientes( );
      dinero=dinero-Ejecutar_Ordenes_Compra_Pendientes( );
      /* Una orden de compra o venta sobre una acción será ejecutada
         siempre que el precio de la orden se encuentre entre el precio
         mínimo y el máximo que haya alcanzado la cotización de la acción.
         Pueden existir ordenes de compra o venta que no se ejecuten */
      IF  NOT EXISTS (Orden\_Venta\_Pendiente\_Ejecucion( ))
         Sd=Seleccionar_Mejores_Acciones(fecha, tipo\_seleccion);
         IF  NOT EXISTS (Orden_Compra_Pendiente_Ejecucion( ))
         /* No tenemos ninguna orden de venta o compra pendientes de
            ejecutarse */
            IF EXISTS (acción=Accion_Comprada_Sin_Orden_Venta( ))
               IF (acción NOT IN Sd DURANTE M días)
               /* Tenemos una acción comprada que no ha estado entre las
                  mejores durante M días */
                  Nueva_Orden_Venta(accion);
            ELSE
               /* No tenemos ninguna acción, por lo que compraremos la
                  mejor */
               Nueva_Orden_Compra(mejor(Sd))
         ELSE
            Cambiar_Orden_Compra(mejor(Sd));
```

```
      ELSE
         /* Existe una orden de venta pendiente */
         IF (EXISTS(Orden_Venta_Pendiente_Ejecucion( ) DURANTE V días))
            Cambiar_Orden_Venta_Pendiente_Ejecucion( );
      fecha=Siguiente_Dia(fecha);
   END;
   RETURN dinero;
END;
```

**Figura 4.** TRADINNOVA: Algoritmo heurístico para la compra-venta de acciones

Este algoritmo tiene como variables de entrada las fechas entre las que se va a ejecutar la simulación, la heurística que se utilizará para seleccionar las acciones y el dinero del que se dispone para invertir, y devuelve el dinero que habremos conseguido después de haber realizado diferentes inversiones en el mercado de acuerdo a unos parámetros que se habrán especificado previamente, como son los valores de $V$ y de $M$.

## 4. Parámetros de la simulación y resultados obtenidos

Se ha ejecutado TRADINNOVA en el Mercado Continuo Español entre los años 2001 y 2004, realizando una simulación en la que se ha tenido en cuenta la operatoria del mercado, aplicando comisiones y dando las órdenes de compra-venta limitadas a un precio, consiguiendo unos resultados esperanzadores.

Veamos a continuación, los parámetros definidos para la simulación y los resultados obtenidos para distintos periodos.

### 4.1. Parámetros utilizados para la simulación

Una vez implementado este algoritmo en JAVA, se ha realizado la simulación teniendo en cuenta las acciones más representativas del mercado continuo español que tuvieran cotización oficial desde el 02/01/2001 hasta el 30/12/2004.

Del IBEX35 se han escogido 28 acciones:

| *Acciones del IBEX35* | | |
|---|---|---|
| *Abentis (ABE)* | *ACS* | *Acerinox (ACX)* |
| *Altadis (ALT)* | *Acciona (ANA)* | *BBVA* |
| *Bankinter (BKT)* | *Endesa (ELE)* | *FCC* |
| *Ferrovial (FER)* | *Gás Natural (GAS)* | *Gamesa (GAM)* |
| *Iberdrola (IBE)* | *Indra (IDR)* | *Arcelor (LOR)* |
| *Mapfre (MAP)* | *Metrovacesa (MVC)* | *Popular (POP)* |
| *Prisa (PRS)* | *Red Eléctrica (REE)* | *Repsol (REP)* |
| *Santander (SAN)* | *Sogecable (SGC)* | *Sacyr Vallehermoso (SYV)* |
| *Telefónica (TEF)* | *Telefónica Móviles (TEM)* | *TPI* |
| *Unión FENOSA (UNF)* | | |

Y del resto del mercado 9 acciones:

| *Acciones del resto del mercado* | | |
|---|---|---|
| *Alba (ALB)* | *Banesto (BTO)* | *Cepsa (CEP)* |
| *Colonial (COL)* | *Ercros (ECR)* | *Ebro-Puleva (EVA)* |
| *NH Hoteles (NHH)* | *Jazztel (JAZ)* | *Zeltia (ZEL)* |

Mediante un fichero .INI se especifican los distintos parámetros de entrada al algoritmo (tabla 1).

| Parámetros de TRADINNOVA |
|---|
| DINERO-INICIAL= 100000 |
| NUM-INVERSORES = 1 |
| COMISION-COMPRA-VENTA = 0.2 |
| BOLSA-CANON = MADRID |
| |
| FECHA-DESDE = 02/01/2001 |
| FECHA-HASTA = 30/12/2004 |
| TIPO-SELECCIÓN = REVALORIZACION-AYER |
| VALOR-MIN-SELECCION = 0.1 |
| OPCION-SELECCION = MEJOR |
| VARIACIÓN-MEJOR-ACCION-SELECCION = 80 |
| |
| TIPO-PRECIO-COMPRA = APERTURA-DIA-SIGUIENTE |
| TIPO-PRECIO-VENTA = ALTO |
| |
| DIAS-SIN-EJECUTAR-VENTA = 10 |
| DIAS-SIN-SER-MEJOR-ACCION = 12 |

**Cuadro 1.** TRADINNOVA.INI: Parámetros de entrada al algoritmo

A continuación vamos a explicar detalladamente estos parámetros (Tabla 1) y qué otras posibles opciones se encuentran implementadas:

En este fichero se define el dinero que se dispondrá inicialmente para la inversión en bolsa (DINERO-INICIAL = 100000 Euros), las fechas que comprenden el periodo de estudio (FECHA-DESDE – FECHA-HASTA), el mercado donde se realiza la compra-venta de los valores (BOLSA-CANON = MADRID) para calcular el tipo de canon que cobrará (en la bolsa de Madrid se cargan cánones por

operación y liquidación de valores) y la comisión que cobra el intermediario financiero (COMISION-COMPRA-VENTA = 0.2 %) por realizar la compra-venta.

En TIPO-SELECCION se especifica la heurística que se utiliza para seleccionar las acciones, conforme a las preferencias y objetivos del inversor. Para este trabajo se ha escogido una sencilla de calcular, seleccionar las acciones que más se revalorizaron en el día anterior (TIPO-SELECCION = REVALORIZACION-AYER), de forma que para cada día se calculará la revalorización que han tenido cada una de las acciones respecto al día anterior, así una acción será mas apetecible cuanto mayor haya sido su revalorización. Esta técnica la elegiría un inversor arriesgado, ya que se van a seleccionar las acciones más volátiles sin tener en cuenta el riesgo asociado, por lo que se pretende conseguir el objetivo del inversor de alcanzar la máxima rentabilidad.

Se encuentran implementadas varias heurísticas más, como la de seleccionar las acciones aleatoriamente (RANDOM), ver cuantos días seguidos se encuentra subiendo cada acción (DIAS-SEGUIDOS-ANTERIORES-SUBIENDO), o aplicarles análisis técnico (DOUBLE-MOVING-AVERAGE).

Las heurísticas de selección de las acciones tendrán unos valores mínimos a partir de los cuales se recomiende la compra de una acción, así las revalorizaciones que tuvieron en el día anterior cada una de las acciones tendrán que tener como valor mínimo para que puedan ser seleccionadas (VALOR-MIN-SELECCION=0.1 %). Este valor mínimo será el máximo de entre VALOR-MIN-SELECCION y la variación permitida (VARIACION-MEJOR-ACCION=80 %) a la mejor revalorización encontrada. De las acciones seleccionadas escogeremos una para realizar su compra, se podría escoger una aleatoriamente (OPCION-SELECCION = RANDOM) o bien optar, tal y como se ha hecho, por la mejor (OPCIÓN-SELECCION = MEJOR).

El precio de compra que se dará, será el que marcará la acción en la apertura del día siguiente (TIPO-PRECIO-COMPRA = APERTURA-DÍA-SIGUIENTE) y el precio de venta será el más alto que hubiera alcanzado esa acción en ese día (TIPO-PRECIO-VENTA = ALTO). Existen más posibilidades para los distintos tipos de precios, ya que también se podría escoger entre el precio mas bajo (BAJO), la media entre el precio mas alto y el mas bajo (MEDIO) o escoger el precio de cierre de la acción (CIERRE).

Vamos a dejar como máximo 12 días (valor de $M$) el los que una acción no se encontrará entre las acciones seleccionadas en esos días para que se introduzca su orden de venta (DIAS-SIN-SER-MEJOR-ACCION = 12) y en caso de que exista una orden de venta introducida y no se logre ejecutar en 10 días (valor de $V$) se bajará automáticamente su precio de venta (DIAS-SIN-EJECUTAR-VENTA = 10).

### 4.2. Resultados

Se ha realizado la simulación con diferentes periodos, para tener más datos empíricos, obteniéndose en general unos resultados bastante buenos. En la Figura 5 se muestra la evolución del IBEX35 durante el periodo 2001-2004, y en la Figura 6 mostramos para los diferentes periodos en los que se ha realizado

la simulación la revalorización obtenida por el IBEX35 y la que se obtiene con TRADINNOVA.

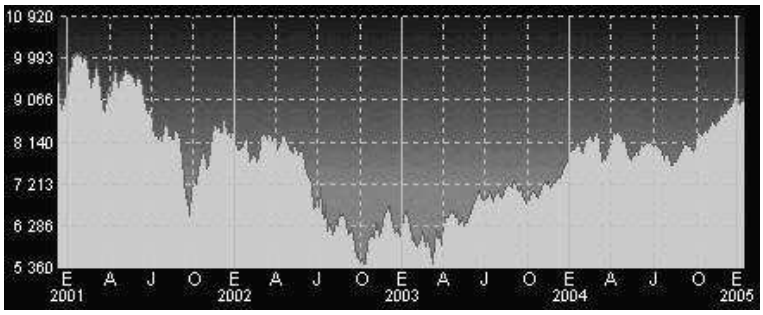



**Figura 5.** Evolución IBEX35 (2001-2004)

Tal y como podemos observar, en el periodo 2001 al 2004, el IBEX35 no ha obtenido ninguna revalorización, mientras que la simulación ha logrado una revalorización del 19 %, pero si utilizamos periodos más cortos con los últimos años, conseguimos unos resultados espectaculares, como una revalorización de la simulación para los dos últimos años del 358 % frente a un 44 % del IBEX35.

En concreto, en los periodos con pérdidas en el IBEX35 (los primeros años), la simulación consigue unos resultados regulares, así para el año 2001 el IBEX35 perdió un 7.51 % frente a una ganancia de la simulación del 0.35 % y para el año 2002 donde el IBEX35 perdió un 27.93 %, la simulación pierde un 49,85 %.

## 5. Conclusiones y vías futuras

Este trabajo se ha centrado en el campo bursátil y hemos presentado un sistema para la toma de decisiones en la compra-venta de acciones de la cual se ha derivado el diseño de un algoritmo heurístico (TRADINNOVA).

La aplicación de esta política rigurosa de compra-venta de acciones, junto a la elección en la simulación de una heurística tan sencilla como la revalorización que tuvieron las acciones en el día anterior, ha logrado superar la revalorización del IBEX en una amplia mayoría de casos, abriendo una nueva vía de investigación a la que todavía le quedan muchos interrogantes, como: ¿Cuáles serían los mejores parámetros para ejecutar el algoritmo? ¿Existe alguna otra heurística que funcione mejor para seleccionar las acciones? ¿Por qué da buenos resultados con los parámetros actuales? ¿Cómo funcionará con otros mercados?, etc

Como futuros trabajos se podrían realizar muchas mejoras, como por ejemplo, poder tener una cartera de acciones, pudiendo mantener compradas al mismo tiempo varias acciones y realizar la venta de las acciones que se consideren menos recomendables. Asimismo se podría mejorar la forma de seleccionar las mejores acciones, utilizando por ejemplo la optimización mediante colonia de hormigas, o también se le podrían hacer mejoras en la política de compra-venta, aplicando nueva reglas, como por ejemplo, dar una orden de venta si después de comprar

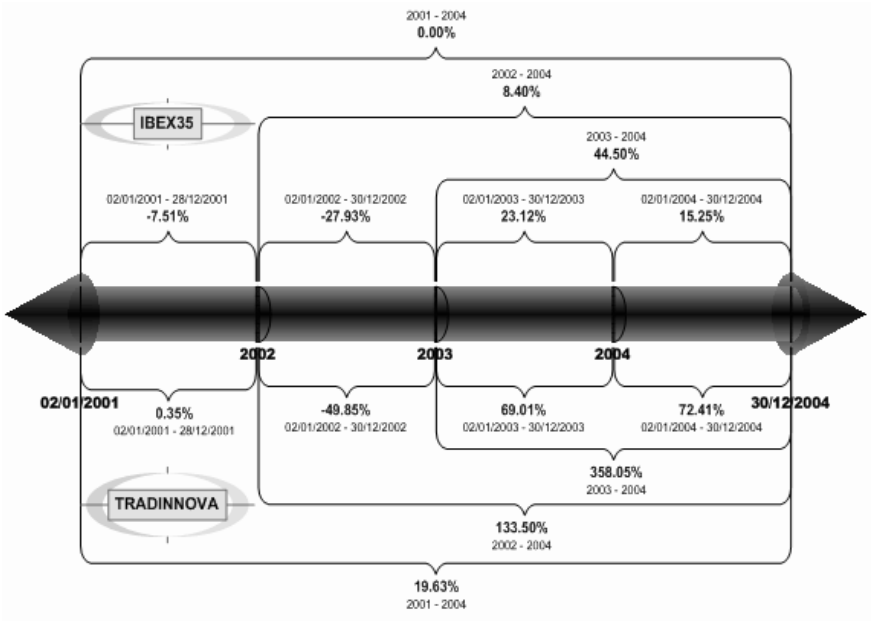



**Figura 6.** Comparación de las revalorizaciones del IBEX35 y TRADINNOVA

una acción esta acción baja durante B días seguidos, o incluso se podría hacer que el algoritmo fuera auto-adaptativo, seleccionando en cada momento la mejor heurística a utilizar y adaptando los parámetros del algoritmo conforme evoluciona el mercado.

## Agradecimientos

Los autores agradecen al Ministerio de Educación y Ciencia y al Fondo Europeo de Desarrollo Regional (FEDER) por su soporte parcial, bajo el proyecto TIN2005-08404-C04-02, para el desarrollo de este trabajo.

## Referencias

1. Swingler, K.: Financial Prediction, some pointers, pitfalls, and common errors. Neural Computing and Applications, Volume 4, Number 4 (1996) 192–197.
2. Zekic, M.: Neural Network Applications in Stock Market - A Methodology Analysis. 9th International Conference on Information and Intelligent Systems '98, Eds. Aurer, B., Logozar, R.,Varazdin (1998) 255–263.
3. Januskeviciuts, M.: Testing Stock Market Efficiency Using Neural Networks Case of Lithuania. Thesis. SSE Riga Working Papers 17 (52) (2003) http://www2.sseriga.edu.lv/library/working_papers/FT_2003_17.pdf

4. Tsang, E.P.K., Li, J.: Combining Ordinal Financial Predictions with Genetic Programming. Second International Conference on Intelligent Data Engineering and Automated Learning (IDEAL-2000), Hong Kong, December (2000) 13–15.
5. Drake, A.E.: Genetic Algorithms in Economics and Finance: Forecasting Stock Market Prices and Foreign Exchange - A Review. http://www.agsm.unsw.edu.au/˜bobm/papers/drake.pdf
6. Davis, L.: Trading on the Edge: Neural, Genetic, and Fuzzy Systems for Chaotic Financial Markets. PART II: 8. Genetic Algorithms and Financial Applications. Deboeck G.J. (ed.), Wiley Finance Edition. 1994.
7. Kroha, P., Baeza-Yates, R.: Classification of Stock Exchange News. Technical Report. Department of Computer Science, Engineering School, Universidad de Chile. 2004. http://www.tu-chemnitz.de/informatik/service/if-berichte/pdf/CSR-04-02.pdf
8. Mittermayer. M.A.: Forecasting Intraday Stock Price Trends with Text Mining Techniques. hicss, p. 30064b, Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 3, 2004.
9. Kührer, M., Loistl, O., Veverka, A.: A Holistic Approach to Portfolio Management. Working Paper (2001). http://ifm.wu-wien.ac.at/Forschung/portfolio.pdf
10. Zargham, M.R., Sayeh, M.R.: A Web-based Information System for Stock Selection and Evaluation, wecwis, p. 81, International Workshop on Advance Issues of E-Commerce and Web-Based Information Systems, 1999.
11. Tseng, C-C.: Comparing Artificial Intelligence Systems for Stock Portfolio Selection. The 9th International Conference of Computing in Economics and Finance, July 11-13 (2003) University of Washington, Seattle, Washington.
12. Tseng, C-C., Gmytrasiewicz, P.J.: Real Time Decision Support System for Portfolio Management. hicss, p. 79. 35th Annual Hawaii International Conference on System Sciences (HICSS'02)- Volume 3 (2002).
13. Forecast. Semanario Bursátil: Indicadores y osciladores técnicos. http://www.f-capital.com/
14. Rubinstein, M.: Markowitz's "Portfolio Selection":A Fifty-Year Retrospective. Journal of Finance 57, No. 3 (2002) 1041–1045.
15. León, T., Liern, V. and Vercher, E.: Viability of infeasible portfolio selection problems: A fuzzy approach. European Journal of Operational Research 139 (2002) 178–189.
16. Mitchell, J.E., Stephen Braun: Rebalancing an Investment Portfolio in the Presence of Transaction Costs. http://www.rpi.edu/˜mitchj/papers/transcosts.html
17. Borodin, A., El-Yaniv, R., Gogan, V.: Can We Learn to Beat the Best Stock. Journal of Artificial Intelligence Research Volume 21 (2004) 579–594.

# Hibridación entre filtros de partículas y metaheurísticas para resolver problemas dinámicos

Juan J. Pantrigo, Ángel Sánchez, Antonio S. Montemayor y Abraham Duarte

Universidad Rey Juan Carlos, Madrid, Spain
{juanjose.pantrigo, angel.sanchez, antonio.sanz, abraham.duarte}@urjc.es

**Resumen** Los problemas de optimización dinámica constituyen una generalización de los problemas de optimización en los cuales la descripción del problema y/o los datos relevantes cambian en el tiempo. Para estos problemas, es conveniente que los algoritmos de optimización utilizados dispongan de estrategias de adaptación a cambios. Los métodos metaheurísticos son procedimientos para explorar de forma efectiva y eficiente el espacio de búsqueda. Por otra parte, los filtros de partículas permiten describir la evolución temporal de un sistema. En este trabajo, se propone una nueva metodología para el desarrollo de algoritmos de optimización dinámica, denominada Filtro de Partículas Metaheurístico (MPF). MFP permite obtener algoritmos para la resolución de problemas de optimización dinámica a través de la hibridación de los métodos anteriores.

## 1. Introducción

Muchos problemas de optimización con aplicación práctica están definidos en entornos no estacionarios, es decir, el valor de las restricciones, variables relevantes del sistema, datos y/o estructuras del problema cambian con el tiempo [1]. Éstos reciben el nombre de problemas de optimización dinámica. Mientras que los métodos de optimización en problemas estáticos deben atender únicamente a encontrar una o un conjunto de soluciones de alta calidad, en los problemas dinámicos, además se debe seguir la evolución de éstas a lo largo del tiempo. En estas circunstancias, se necesitan métodos que permitan adaptar las soluciones continuamente a entornos en constante cambio [1][2].

La resolución de problemas dinámicos con metaheurísticas se aborda mediante dos estrategias posibles para obtener la solución a la instancia actual después de cambios [3]: restablecer el procedimiento de optimización (i) desde el inicio o (ii) desde las mejores soluciones encontradas a las instancias anteriores del problema. El primer enfoque, aborda el problema derivado suponiendo que no está relacionado con el anterior, lo que conlleva un desprecio de información que podría ser útil para reducir el tiempo de ejecución del algoritmo. La segunda estrategia es aplicable si el cambio en el entorno es poco importante,

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 667-678, 2006.

ya que implica una pérdida de diversidad drástica en el conjunto inicial de soluciones. Como ninguna de estas dos estrategias resuelve probemas dinámicos razonablemente, se han propuesto diversas modificaciones que las adaptan a condiciones no estacionarias. Entre ellas, destacan las que se basan en métodos evolutivos [3][4] y métodos constructivos [1]. Los métodos evolutivos son muy flexibles y permiten ajustar la diversidad de la población para adaptarla a los cambios en el entorno. El diseño de algoritmos de optimización dinámica asume frecuentemente que la diversidad es una propiedad deseable para conseguir adaptabilidad [3], ya que su pérdida reduce la capacidad de adaptación del algoritmo a los cambios del sistema. Sin embargo, Andrews [5] advierte que ésta puede ser una versión simplista del problema. Los métodos constructivos permiten ajustar una solución a los cambios del entorno mediante un proceso de *deconstrucción* y *reconstrucción* [6][1].

Los filtros de partículas resuelven problemas dinámicos introduciendo cierto conocimiento sobre la dinámica del sistema. Sin embargo, estos métodos no son adecuados para el tratamiento de problemas demasiado complejos. Por un lado, el cálculo de los pesos de las partículas, que implica la evaluación de la función de ponderación, es una tarea muy costosa para una gran parte de problemas con interés práctico y debe ser realizada una vez por cada partícula y en cada instante temporal. Además, se necesita un número impracticable de partículas para muestrear espacios de estados con un gran número de dimensiones [7][8]. Para abordar estos problemas se han propuesto diferentes procedimientos, que tienen el objetivo común de minimizar el número de partículas utilizadas sin perder calidad en las estimaciones.

La hipótesis central de este trabajo establece que, para resolver problemas de optimización dinámica, es conveniente el concurso de procedimientos de predicción, adaptación y optimización. Puesto que existen, por un lado, algoritmos de optimización (como las metaheurísticas) y, por otro, algoritmos de estimación secuencial (en particular, filtros de partículas), parece razonable abordar los problemas de optimización dinámica utilizando características de ambos métodos de manera sinérgica. En este trabajo se presenta el Filtro de Partículas Metaheurístico (*Metaheuristic Particle Filter* - MPF), una metodología de diseño de algoritmos híbridos orientados a la resolución más eficiente de problemas de optimización dinámica. Este método establece el marco conceptual de hibridación que permite obtener por instanciación diferentes algoritmos híbridos generales, que pueden ser posteriormente adaptados para resolver problemas concretos.

## 2. El problema de la estimación secuencial y los filtros de partículas

El problema de la estimación secuencial, desde un punto de vista bayesiano, consiste en el cálculo recursivo, con un cierto grado de confianza, del estado del sistema $\mathbf{x}_t$ en el instante $t$, dadas las observaciones $\mathbf{z}_{1:t} = \{\mathbf{z}_1, \ldots, \mathbf{z}_t\}$. Para ello,

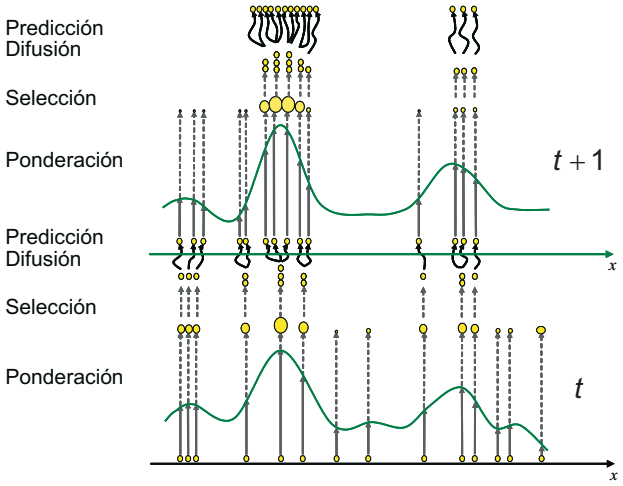



**Figura 1.** Esquema general del filtro de partículas.

es necesario calcular la *pdf* $p(\mathbf{x}_t|\mathbf{z}_{1:t})$. La *pdf* inicial $p(\mathbf{x}_0|\mathbf{z}_0) \equiv p(\mathbf{x}_0)$ se asume conocida. Así, la *pdf a posteriori* $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ se puede calcular en dos etapas:

1. **Predicción**: supóngase que se dispone de la *pdf* $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$ en el instante $t-1$. La etapa de predicción implica el uso del modelo del sistema para obtener de manera recursiva la *pdf a priori* $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ en el instante siguiente $t$, mediante la ecuación de Chapman-Kolmogorov:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1} \tag{1}$$

2. **Evaluación**: En el instante $t$, se dispone de una nueva medida $(\mathbf{z}_t)$ que se puede utilizar durante la actualización del estado del sistema mediante el uso del teorema de Bayes:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})} \tag{2}$$

Las ecuaciones (1) y (2) conforman la base de la solución bayesiana óptima. Esta propagación recursiva de la densidad *a posteriori* es únicamente un resultado conceptual, puesto que, en general, no se puede determinar analíticamente [9]. Por esta razón, se encuentra en la literatura un número significativo de trabajos que presentan modelos aproximados de esas funciones de distribución.

El algoritmo de filtro de partículas (PF) es una técnica para implementar filtros recursivos bayesianos [9], cuya idea clave consiste en representar la *pdf a posteriori* por un conjunto de muestras discretas con pesos asociados y calcular

los estimados basándose en esas muestras y pesos [10]. Cuando el número de muestras es lo suficientemente grande, esta representación se convierte en equivalente a la descripción funcional de la *pdf a posteriori* [10]. La figura 1 muestra una representación gráfica del funcionamiento de PF. Inicialmente, se genera una población de partículas utilizando una *pdf* conocida. En este momento se reciben nuevas medidas y se calcula el peso para cada partícula, utilizando la ecuación (2). El resultado es un conjunto de partículas con pesos asociados que constituyen una aproximación discreta de la *pdf a posteriori*. A continuación, se lleva a cabo el paso de remuestreo. En esta etapa, se seleccionan las partículas "más importantes" (es decir, con mayor peso) para aproximar la pdf a priori en el instante siguiente aplicando sobre este conjunto la regla de predicción dada por la ecuación (1).

## 3. Problemas de optimización y metaheurísticas

Un problema de optimización (OP) se puede definir como la mejora (minimización o maximización) de una función objetivo $f : S \rightarrow \mathbb{R}$ sujeta a una serie de restricciones [11]. Formalmente,

$$OP = \begin{cases} optimizar & f(x) \\ sujeto \quad a & x \in F \end{cases} \tag{3}$$

donde $S$ es con un conjunto de soluciones, en general: $S \subseteq \mathbb{R}^n$, $f : S \rightarrow \mathbb{R}$ es una función de coste que asigna a cada solución candidata ($x \in S$) un valor numérico ($f(x) \in \mathbb{R}$) y $F$ determina el conjunto de soluciones factibles $x \in F \subseteq S$.

Existe una colección importante de problemas interesantes en diferentes áreas de la ciencia y la ingeniería para la que no se dispone de algoritmos exactos que permitan encontrar la solución óptima en tiempos razonables. Una alternativa consiste en diseñar algoritmos aproximados que encuentren soluciones de alta calidad en un tiempo asumible. De entre todos los métodos aproximados, destacan las metaheurísticas por su eficiencia, efectividad y flexibilidad. Las metaheurísticas combinan métodos heurísticos básicos en un marco de trabajo de más alto nivel con el objetivo de mejorar la exploración sistemática del espacio de búsqueda del problema. Las metaheurísticas se han aplicado con éxito a una gran variedad de problemas de optimización. Este tipo de métodos combinan ideas que provienen de cuatro campos de investigación bien distintos: las técnicas de diseño de algoritmos (resuelven una colección de problemas), algoritmos específicos (dependientes del problema que se quiere resolver), fuentes de inspiración (del mundo real) y métodos estadísticos.

En una primera aproximación, se pueden clasificar las metaheurísticas en dos grandes bloques [12][13]: trayectoriales y poblacionales. Las metaheurísticas trayectoriales manejan en todo momento una sola solución y deben su nombre a que el proceso de búsqueda que desarrollan se caracteriza por una trayectoria en el espacio de soluciones. Es decir, partiendo de una solución inicial, generan un camino en el espacio de búsqueda mediante operaciones de movimiento. Dentro de estas metaheurísticas se pueden destacar por su interés la Búsqueda

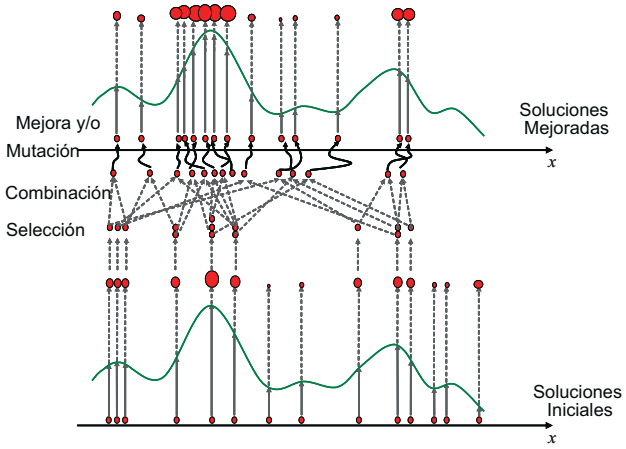



**Figura 2.** Esquema general de las metaheurísticas poblacionales.

Tabú (*Tabu Search* - TS), Procedimiento Aleatorio y Adaptativo de Búsqueda Miope (*Greedy Randomized Adaptive Search Procedure* - GRASP) o Recocido Simulado (*Simulated Annealing* - SA). Las metaheurísticas poblacionales implementan el proceso de búsqueda manteniendo simultáneamente un conjunto de soluciones. Ejemplos de este tipo de metaheurísticas son los Algoritmos Evolutivos (*Evolutionary Algorithms* - EA), la Búsqueda Dispersa (*Scatter Search* - SS), el Reencadenamiento de Trayectorias (*Path Relinking* - PR) y los Algoritmos de Estimación de la Distribución (*Estimation Distribution Algorithms* - EDA). Las metaheurísticas poblacionales son de especial interés para nuestra propuesta de hibridación, puesto que encajan perfectamente en el marco de trabajo del filtro de partículas, cuyo carácter es eminentemente poblacional.

En la figura 2 se representa gráficamente el esquema de funcionamiento general de una metaheurística poblacional. Inicialmente, se parte de soluciones generadas aleatoriamente. En una segunda etapa, se seleccionan aquellas que poseen una mayor calidad para generar otras mediante combinación. A las soluciones generadas, se les puede aplicar una etapa de mutación y otra de mejora, con el objetivo de dotar a la población de diversidad o calidad, respectivamente. Este proceso se repite hasta que se alcanza una condición de parada establecida previamente.

## 4. El Filtro de Partículas Metaheurístico

El Filtro de Partículas Metaheurístico (MPF) es una metodología de desarrollo de algoritmos híbridos entre filtros de partículas y metaheurísticas (o heurísticas), para la resolución de problemas de optimización dinámica. Se debe

distinguir entre el ámbito de la metodología del MPF y el de los algoritmos que se derivan de su aplicación.

La metodología MPF se aplica a algoritmos de estimación secuencial basados en filtros de partículas y en metaheurísticas. El propósito de esta metodología es el diseño de algoritmos de optimización dinámica mediante la hibridación de métodos de ambas familias. Es importante notar que los algoritmos resultantes son de propósito general en cuanto que los algoritmos de partida lo son, de modo que se pueden aplicar a un conjunto amplio de problemas dinámicos, pero son específicos en cuanto que están especializados en optimización dinámica.

### 4.1. Los elementos de la metodología MPF

En esta sección se introduce el conjunto de definiciones necesarias para describir los elementos que componen el marco de trabajo de MPF. Gran parte de estos elementos son heredados de los algoritmos de filtros de partículas y de las metaheurísticas.

**Definition 1.** *Una* ***solución*** *($\boldsymbol{x} \in S \equiv \mathbb{R}^D$) constituye la información necesaria y suficiente para representar una posible respuesta al problema de optimización.*

**Definition 2.** *Se entiende por* ***medida*** *($\boldsymbol{z} \in \mathbb{R}^M$) aquel conjunto de datos recogidos en un instante de tiempo, utilizando el proceso de medida (usualmente $M > D$).*

**Definition 3.** *Dado un espacio de soluciones $S$ y una medida $\boldsymbol{z}$, la* ***función de ponderación*** *($f : S \times \mathbb{R}^M \rightarrow \mathbb{R}$) es un operador que establece una relación entre el espacio de soluciones ($\boldsymbol{x} \in S$) y las medidas ($\boldsymbol{z}$).*

**Definition 4.** *Dadas una solución $\boldsymbol{x}$, una medida $\boldsymbol{z}$ y una función de ponderación $f$, se denomina* ***peso*** *de la solución $\boldsymbol{x}$, y se denota $\omega$, al resultado de aplicar $f$ sobre $\boldsymbol{x}$ y $\boldsymbol{z}$, es decir: $\omega = f(\boldsymbol{x}, \boldsymbol{z})$*

**Definition 5.** *Se define un* ***individuo*** *como la dupla formada por una solución $\boldsymbol{x}$ y su peso $\omega$, y se denota como: $(\boldsymbol{x}, \omega)$.*

En el marco PF, el par formado por una solución y un peso es conocido como *partícula*. Sin embargo, en el ámbito de las metaheurísticas, recibe diferentes nombres dependiendo del método. Por ejemplo, este elemento es conocido como *individuo* en GA, MA y EDA, mientras que en el contexto de SS y PR se llama *solución*.

**Definition 6.** *El* ***conjunto soporte*** *(SupportSet) es el formado por $N_s$ individuos que evolucionan en el tiempo guiados por un algoritmo de estimación secuencial.*

En PF, *SupportSet* equivale al conjunto de partículas. En el contexto de los algoritmos de optimización, *SupportSet* juega el papel del conjunto inicial de soluciones.

**Definition 7.** *El* ***conjunto mejorado*** *(ImprovedSet) es un subconjunto del conjunto soporte, formado por $N_s$ individuos que forman el conjunto inicial para la etapa de optimización.*

**Definition 8.** *Un* ***Algoritmo de Estimación Secuencial Montecarlo (PF)*** *es un procedimiento que determina la evolución temporal de SupportSet, entendiéndola como una representación discreta de una* pdf *utilizando las etapas generales de predicción y actualización.*

**Definition 9.** *Una* ***Metaheurística (M)*** *es un algoritmo aproximado de optimización basado en poblaciones que dirige la combinación de soluciones de ImprovedSet para obtener otras de mayor calidad, en el contexto MPF.*

**Definition 10.** *El procedimiento de* ***Selección del conjunto mejorado (Select)*** *permite seleccionar adecuadamente un conjunto de individuos de SupportSet para formar ImprovedSet, siguiendo un determinado criterio.*

**Definition 11.** *El procedimiento de* ***sustitución en el conjunto soporte (Replace)*** *permite que los individuos que forman parte del conjunto mejorado en el momento en el que finaliza el proceso de optimización, pasen a formar parte del conjunto soporte.*

**Definition 12.** *El procedimiento de* ***Estimación (Estimate)*** *establece una estimación de la mejor solución que describe al sistema en cada instante, basándose en una combinación adecuada de los individuos que forman parte del conjunto mejorado.*

### 4.2. Metodología de hibridación

MPF proporciona un método general de combinación de algoritmos de estimación secuencial (basados en filtros de partículas) con métodos metaheurísticos. Los algoritmos derivados de la aplicación de MPF están orientados a resolver problemas de optimización dinámica. MPF proporciona un método sistemático para la implementación de algoritmos híbridos que se puede resumir en la siguiente secuencia de pasos:

1. *Elegir un PF e identificar las etapas de inicialización, predicción y actualización.* La labor de las estrategias extraídas de PF consisten en inicializar y mantener el conjunto soporte de individuos *SupportSet* a lo largo del tiempo. Desde el punto de vista de M, el filtro de partículas juega el papel de inicializador de soluciones de la metaheurística (o generador de soluciones diversas), en cada instante temporal.
2. *Elegir una M e identificar las etapas de inicialización y optimización.* Esta etapa del algoritmo tiene como objetivo mejorar el conjunto de individuos *ImprovedSet* en cada instante temporal. Desde el punto de vista del PF, la metaheurística poblacional es un procedimiento de corrección de la estimación. Se debe recordar que PF posee sus propias estrategias para la corrección de la estimación.

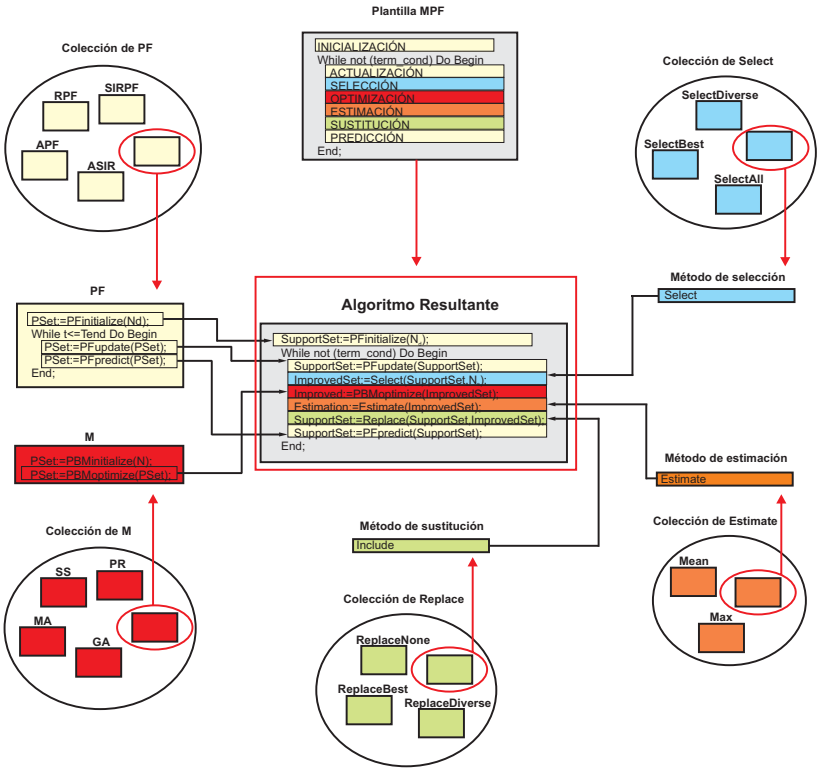



**Figura 3.** Esquema general del Filtro de Partículas Metaheurístico (MPF) y cómo se obtiene un algoritmo derivado de dicho esquema.

3. *Diseñar el procedimiento de selección.* El procedimiento de selección del conjunto mejorado debe elegir $N_i$ individuos de *SupportSet* para formar el conjunto *ImprovedSet*, siguiendo un determinado criterio.
4. *Diseñar un procedimiento de sustitución.* El procedimiento de sustitución se utiliza para añadir los individuos del conjunto mejorado *ImprovedSet* en *SupportSet*. Como el tamaño de *SupportSet* ($N_s$) debe permanecer constante (ya que el número de partículas en los algoritmos PF se mantiene constante a lo largo del proceso de estimación), la incorporación de nuevos individuos implica forzosamente que otros deben ser descartados.
5. *Elegir un procedimiento de estimación del estado del sistema.* El procedimiento de estimación calcula un estimado del estado del sistema en cada instante, que constituye la salida del algoritmo.
6. *Organizar los elementos utilizando la plantilla MPF.* Una vez que se han obtenido los diferentes elementos descritos, se organizan utilizando

la plantilla MPF, que se muestra en la parte central de la figura 3. Las etapas encajan como piezas independientes en la plantilla para diseñar un algoritmo híbrido. El color de las cajas contenedoras indica la procedencia del procedimiento.

### 4.3. Resolución de problemas dinámicos mediante la metodología MPF

Para resolver un problemas dinámicos mediante la metodología MPF hay que considerar los siguientes pasos:

1. Determinar si el problema estacionario existe. En este caso:
   *a*) Estudiar los métodos metaheurísticos que han sido aplicados con éxito a este problema
   *b*) Elegir uno de ellos para ser hibridado con un filtro de partículas

   si no
   *a*) Estudiar el problema dinámico desde el punto de vista de los filtros de partículas
   *b*) Elegir una metaheurística o procedimiento heurístico adecuado en función de la dificultad del problema
2. Hibridar ambos métodos utilizando la metodología MPF
3. Determinar los parámetros adecuados mediante experimentación

Por lo tanto, para obtener el máximo rendimiento de esta metodología, es necesario, en primer lugar, estudiar el problema dinámico y también su versión estacionaria si existe. El estudio del problema debe orientarse a responder a preguntas como si es un problema difícil, si es necesario el concurso de una metaheurística o basta con una heurística, si prima la calidad de las soluciones o tiene restricciones severas de tiempo, etc.

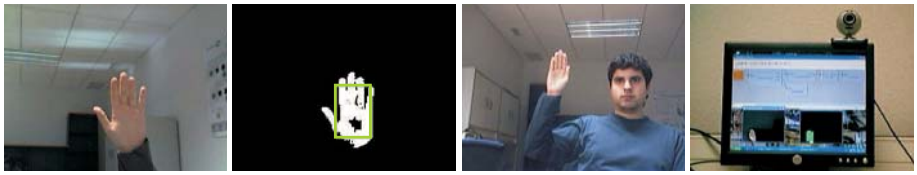



**Figura 4.** Sistema Visual Ratón: a través de la imagen de la mano (a) y de una detección de color (b) se consigue que un usuario (c) equipado con una cámara web pueda manejar un ratón del ordenador (d).

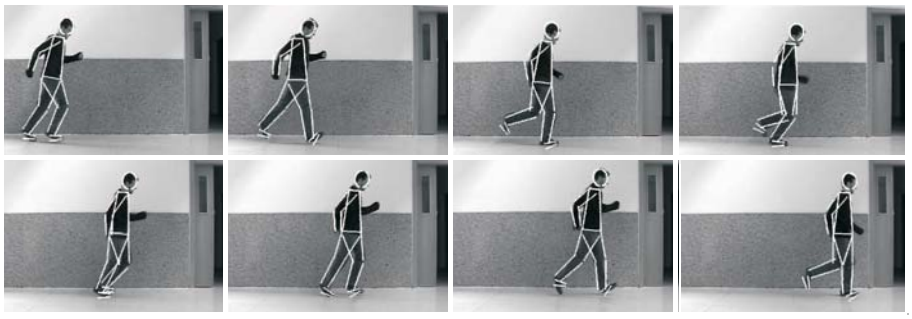


**Figura 5.** Seguimiento de un corredor utilizando el algoritmo SSPF.

## 5. Resultados

En esta sección, se presentan brevemente algunos de los problemas dinámicos que se han abordado desde el marco de MPF. Se han agrupado en dos grandes bloques, según la temática abordada.

### 5.1. Seguimiento visual

En este problema se han utilizado tres algoritmos MPF: Filtro de Partículas con Búsqueda Local (LSPF), Filtro de Partículas con Búsqueda Dispersa (SSPF) y Filtro de Partículas con Reencadenamiento de Trayectorias (PRPF). LSPF es un algoritmo que combina un filtro de partículas con una búsqueda local sobre la mejor solución encontrada en cada instante de tiempo. Este algoritmo es muy útil para el seguimiento preciso de un objeto. Se ha aplicado al seguimiento de la mano de un usuario para dirigir la trayectoria del puntero del ratón del ordenador [14] (ver figura 4).

SSPF y PRPF son hibridaciones de filtro de partículas con las metaheurísticas búsqueda dispersa (SS) y reencadenamiento de trayectorias (PR). Ambos algoritmos se han aplicado con éxito a problemas de seguimiento visual más complejos como el seguimiento de múltiples objetos [15] y seguimiento de objetos articulados [16] (ver figura 5). En ambos casos, por su interés, se han aplicado los algoritmos resultantes al seguimiento del cuerpo humano.

### 5.2. Problemas combinatorios dinámicos

El problema dinámico del viajante de comercio (*Dynamic Travelling Salesman Problem* - DTSP) es una generalización del Problema del Viajante de Comercio (TSP), que consiste en encontrar la ruta más corta de un viajero que comienza su viaje en una ciudad determinada, visita un conjunto prescrito de ciudades una y sólo una vez y vuelve a la ciudad de partida. El DTSP es una versión dependiente

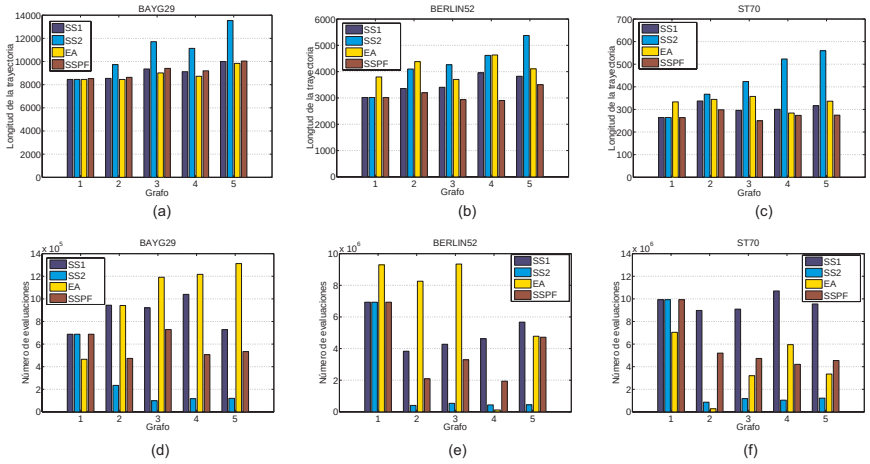



**Figura 6.** Longitud de las trayectorias en (a) $BAYG29$, (b) $BERLIN52$ y (c) $ST70$ y tiempo de ejecución requerido en (d) $BAYG29$, (e) $BERLIN52$ y (f) $ST70$ utilizando SS1, SS2, MA y SSPF.

del tiempo de este problema donde, en cada instancia, puede modificarse el grafo que lo define.

Se ha aplicado el Filtro de Partículas con Búsqueda Dispersa (SSPF) al DTSP [17]. Como resultado, se ha obtenido que el algoritmo SSPF presenta un mayor rendimiento que los algoritmos metaheurísticos con los que se ha comparado (Algoritmos meméticos y búsqueda dispersa). En la figura 6 se muestran los resultados obtenidos en calidad y tiempo de los diferentes algoritmos estudiados.

## 6. Conclusiones

En este trabajo, se ha perseguido identificar y extraer las mejores características de los filtros de partículas (PF) y de metaheurísticas, y utilizarlas en el diseño de algoritmos de optimización que puedan adaptarse a cambios. Como resultado, se ha propuesto un método de hibridación entre estos métodos, denominado *Filtro de Partículas Metaheurístico* (MPF). De su aplicación, se obtienen algoritmos que heredan las características más sobresalientes de cada método considerado: la capacidad de predicción y adaptación de PF y las estrategias de optimización de M. Se han aplicado los algoritmos MPF a diferentes problemas dinámicos complejos y se han comparado con otros métodos utilizados en la literatura para esos problemas, obteniendo los mejores resultados con los algoritmos propuestos. En definitiva, se ha podido comprobar, al menos sobre los experimentos analizados que, "*La combinación de estrategias de adaptación, predicción y optimización incrementa la eficiencia de la búsqueda de soluciones*

*de alta calidad para problemas de optimización dinámica*". Este método híbrido establece un prometedor enfoque en el ámbito de la optimización dinámica.

## Referencias


1. Randall, M.: Constructive Meta-heuristics for Dynamic Optimization Problems. Technical Report. School of Information Technology. Bond University (2002)
2. Saleem, S. , Reynolds, R.: Cultural Algorithms in Dynamic Environments. Proceedings of the Congress on Evolutionary Computation (2000) 1513–1520
3. Mori, N., Kita, H.: Genetic algorithms for adaptation to dynamic environments - a survey. 26th Annual Conf of the IEEE Industrial Electronics Society, Vol 4 (2000) 2947–2952
4. Branke, J.: Evolutionary Approaches to Dynamic Optimization Problems - Updated Survey. GECCO Workshop on Evolutionary Algorithms for Dynamic Optimization Problems (2001) 27–30
5. Andrews, M., Tuson, A.: Diversity does not Necessarily Imply Adaptability. Proceedings of the Workshop on Evolutionary Algorithms for Dynamic Optimization Problems at the Genetic and Evolutionary Computation Conference (GECCO) (2003) 24–28
6. Dorigo, M., Ganbardella, L.M.,: Ant Colony System: A Coopertaive Learning Approach to the Travelling Salesman Problem. IEEE Transaction on Evolutionary Computation, 1:1 (1997) 53–66
7. Lee, M.W., Cohen, I., Jung, S.K.: Particle Filter with analytical inference for human body tracking. IEEE Workshop on Motion and Video Computing (2002)
8. Deutscher, J., Reid, I., Davidson, A.: Automatic Partitioning of High Dimensional Search Spaces associated with Articulated Body Motion Capture. Proc. of the IEEE Conf. on CVPR, vol. 2 (2001) 8–14
9. Doucet, A., Godsill, S., Andrieu, C.: On sequential Monte Carlo sampling methods for Bayesian filtering. Statistics and Computing, 10:3 (2000) 197 - 208
10. Arulampalam, S., Maskell, S., Gordon, N., Clapp, T.: A Tutorial on Particle Filters for On-line Nonlinear/Non-Gaussian Bayesian Tracking. IEEE Trans. on Signal Processing 50:2 (2002) 174–188
11. Yaigura, M., Ibaraki, T.: On metaheuristic algorithms for combinatorial optimization problems. Systems and Computers in Japan 3 (2001) 33–55
12. Blum, C., Roli, A.: Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys 35:3 (2003) 268–308
13. Gendreau, M.: An Introduction to Tabu Search. In F. Glover, G. A. Kochenberger editors, Handbook of Metaheuristic. Kluwer Academic Publishers (2003) 37-54
14. Pantrigo, J.J., Montemayor, A.S., Sánchez, A.: Local Search Particle Filter applied to Human-Computer Interaction. 4th Int. Symposium on Image and Signal Proc. and Analysis (2005) 279–284
15. Pantrigo, J.J., Montemayor, A.S., Cabido, R.: Scatter Search Particle Filter to for 2D Real-Time Hands and Face Tracking. LNCS 3617 (2005) 953–960
16. Pantrigo, J.J., Sánchez, A., Gianikellis, K., Montemayor, A.S.: Combining Particle Filter and Population-based Metaheuristics for Visual Articulated Motion Tracking. Electronic Letters on Computer Vision and Image Analysis 5:3 (2005) 68–83
17. Pantrigo, J.J., Sánchez, A., Cabido, R., Duarte, A.: Scatter Search Particle Filter to Solve the Dynamic Travelling Salesman Problem. LNCS 3448 (2005) 177–189

# Modelado del coordinador de un sistema meta-heurístico cooperativo mediante SoftComputing

J.M. Cadenas, R.A. Díaz-Valladares, M.C. Garrido,
L.D. Hernández y E. Serrano

Dpto. Ingeniería de la Información y las Comunicaciones.
Facultad de Informática. Universidad de Murcia. Spain.
jcadenas@um.es, rdiaz98@gmail.com, cgarrido@um.es
ldaniel@dif.um.es, emilio.s.f@alu.um.es

**Resumen** Cuando se propone diseñar un sistema metaheurístico cooperativo centralizado para resolver problemas de optimización combinatoria, uno de los problemas que surge es la modelización del agente coordinador del sistema de cara a que haga lo más efectiva posible la coordinación entre las distintas metaheurísticas. Una de las formas de poder atacar este problema es mediante la extracción del conocimiento que se pueda obtener de distintas ejecuciones previas de las metaheurísticas a la hora de resolver instancias del problema o problemas planteados. En este trabajo se estudian las dos primeras fases que formarían el proceso de extracción del conocimiento a partir de datos aplicadas al problema de la descripción del coordinador de un sistema metaheurístico cooperativo.

*Palabras clave:* Minería de Datos, SoftComputing, Conjuntos Fuzzy, Sistemas Metaheurísticos cooperativos

## 1. Sistemas metaheurísticos cooperativos

Los algoritmos metaheurísticos brindan estrategias efectivas para encontrar soluciones aproximadas a problemas de optimización, aunque los tiempos de ejecución asociados con la exploración del espacio de búsqueda pueden resultar muy grandes. Además, existen dificultades debido a que dado un problema nuevo, es prácticamente imposible determinar cuál es el mejor algoritmo metaheurístico para resolverlo; aunque conozcamos un buen algoritmo, puede darse el caso de que el tipo de instancia que necesitamos resolver resulte compleja para el algoritmo; además no existe ninguna metodología práctica que permita encontrar el mejor conjunto de parámetros para resolver una instancia particular. Debido a esto, es difícil encontrar en la bibliografía algoritmos metaheurísticos para resolver problemas de optimización que posean características generales, que sean independientes del problema, robustos, fáciles de implementar y que produzcan resultados aceptables en tiempos razonables.

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 679-688, 2006.

Esto nos lleva a pensar en utilizar diferentes algoritmos metaheurísticos bajo un mismo esquema coordinado para resolver problemas de optimización combinatoria. La robustez se obtendrá por el uso de diferentes combinaciones de algoritmos que cooperan entre sí, alcanzando soluciones de muy alta calidad para diferentes clases de instancias. Esta ejecución cooperativa se puede realizar, entre otras formas, a través de un agente coordinador que controle y modifique el comportamiento de los agentes de manera coordinada. El coordinador tendrá, por tanto, dos tareas fundamentales: recopilar la información del rendimiento de cada uno de los algoritmos metaheurísticos y enviarles órdenes que afecten a sus comportamientos de búsqueda, [5].

Nosotros vamos a trabajar con este tipo de sistema cooperativo, en el cual, surge el problema de la descripción del agente coordinador, de manera que controle y modifique el comportamiento de los algoritmos metaheurísticos de manera eficiente y que suavice los problemas planteados anteriormente relacionados con el comportamiento de los algoritmos metaheurísticos de forma individual. En [4] se presenta una definición del proceso de Minería de Datos para resolver este problema. En este trabajo, se estudian con más profundidad, y proponiendo esquemas concretos, las dos primeras fases de este proceso. Para ello, el trabajo lo hemos organizado de la siguiente forma: en la sección 2, se presenta el proceso de minería de datos que queremos realizar. En ella se describe con mayor detalle la fase de preparación de los datos y la fase de utilización de técnicas de extracción de conocimiento. En esta segunda fase, describiremos el tipo de modelo que queremos obtener y presentaremos una de las técnicas que utilizaremos mostrando cómo podremos obtener el modelo a partir de ella. Por último acabaremos con la sección de conclusiones y trabajos en desarrollo.

## 2. Modelización del agente coordinador

En [4] se presentó el estudio sobre el proceso que deberíamos realizar para obtener la modelización del coordinador del sistema anteriormente descrito. Esta descripción será realizada mediante un conjunto de reglas fuzzy obtenidas a través de un proceso de minería de datos. La Fig. 1 ilustra el proceso completo propuesto para este propósito.

En dicha figura aparecen representadas las distintas fases, típicas en el proceso de extracción del conocimiento, aplicadas a este problema, [4], como son: la preparación de los datos, la minería de datos y la evaluación. A continuación vamos a presentar las dos primeras fases, indicando todas las decisiones necesarias para crear el proceso prototipo para resolver el problema planteado.

### 2.1. Preparación de los datos

En la fase de preparación de los datos queremos obtener una base de datos que contenga la información útil a la cual podamos aplicar las técnicas de minería de datos apropiadas para obtener el modelo. Para ello, el primer paso será la recopilación de información. Nuestras fuentes de información provienen de la

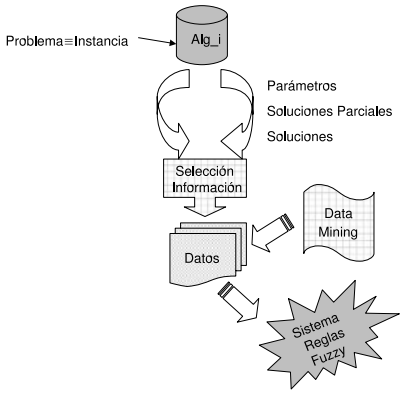



**Figura 1.** Proceso de Minería de Datos para el modelado del agente coordinador

aplicación de distintos algoritmos metaheurísticos a conjuntos de instancias de un problema o problemas. Como información interesante podemos encontrar:

- los parámetros usados en la ejecución de los algoritmos metaheurísticos,
- descripción de la instancia concreta de un problema,
- solución final obtenida,
- solución óptima,
- información obtenida durante la ejecución de un algoritmo.

Con esta información crearemos la estructura inicial de lo que denominaremos un ejemplo, formado por un vector definido por medio de un conjunto de atributos.

Para poder trabajar con estos datos necesitamos definir una estructura de datos que soporte todo tipo de información: características heterogéneas, numéricas y nominales, que puedan presentar valores con imperfección (incertidumbre e imprecisión) así como permitir que el valor de alguna característica de un ejemplo concreto sea desconocido, [1].

Una vez los datos están almacenados en dicha estructura le aplicaremos técnicas de selección de características que permitan establecer cuáles son las más relevantes para el objetivo marcado. Abordaremos esta etapa de preprocesamiento de los datos usando técnicas como redes bayesianas [7], integral difusa [6] y métodos heurísticos basados en "Fuzzy Pattern Matching" [3], entre otras.

Al aplicar esta etapa al conjunto de datos, obtendremos la base de datos sobre la cual aplicaremos las técnicas de minería de datos. En la Fig. 1 se muestra cómo a partir de toda la información inicial, y aplicada la etapa que hemos denominado selección de información, hemos creado el conjunto de datos al que aplicar las técnicas de minería de datos.

**Estructura del conjunto de datos.** Para la creación del conjunto de datos que contenga la información necesaria para la aplicación de las técnicas de extracción de conocimiento, necesitamos inicialmente realizar el siguiente proceso de experimentos:

- Disponer de un problema a resolver, con un conjunto de instancias, $z$, de distintas dificultades.
- Disponer de las soluciones y costos óptimos de todas las instancias seleccionadas anteriormente.
- Disponer de un conjunto de algoritmos metaheurísticos, $x$, dispuestos para ser aplicados a las instancias anteriores.
- Seleccionar el conjunto de los posibles valores para los parámetros de los algoritmos metaheurísticos (estos posibles valores serán los que se crean importantes o interesantes según las soluciones parciales y finales obtenidas). Supongamos que tenemos, por término medio, *par* parámetros para cada algoritmo y tomamos, por término medio, *pos* posibles valores.
- Tomando el conjunto de parámetros *par* para cada algoritmo, con sus *pos* posibles valores, tendremos, por término medio, un total de $y = pos^{par}$ posibles combinaciones de valores para los parámetros de cada algoritmo.
- Para una instancia dada, usaremos para resolverla cada uno de los algoritmos metaheurísticos con todos los posibles valores de sus parámetros. Al final tendremos, como término medio, un total de $x \cdot y \cdot z$ ejecuciones de los algoritmos a las distintas instancias.
- Para cada ejecución de un algoritmo, obtendremos un conjunto de posibles soluciones y costos parciales, $u$, en el cual supondremos que la última es la que propone el algoritmo como solución del problema.

Una vez realizado todo este proceso, tendremos $n$ vectores que contendrán la información obtenida al ejecutar los $x$ algoritmos heurísticos con $y$ posibles combinaciones de sus parámetros y a $z$ instancias del problema más la información relevante que obtengamos del problema-instancia.

Dado que cada vector, tal y como lo hemos descrito, corresponde a un par algoritmo-instancia podrían aparecer vectores de distinto tamaño debido a que distintos algoritmos pueden estar descritos por distintos números de parámetros. Ante esta situación, tomamos la siguiente opción a la hora de su almacenamiento en memoria: Fusionar la información de todos los algoritmos, relativos a una misma instancia, en un vector. Internamente, cada algoritmo, en este vector, quedará representado o identificado por sus parámetros. Es decir, cada vector corresponde a la información que nos proporcionan todos los algoritmos al intentar resolver una instancia dada. Por tanto, los datos de partida para el proceso de extracción de conocimiento serán el conjunto de vectores cuya estructura se muestra en la Fig. 2 (en el cuadro 1 se muestra la tabla de datos obtenidos con

todos sus atributos de los distintos algoritmos). Destacar que se utilizará de cada solución parcial y óptima el costo asociado, sin olvidar que cada uno de ellos lleva asociada la solución que lo obtiene.

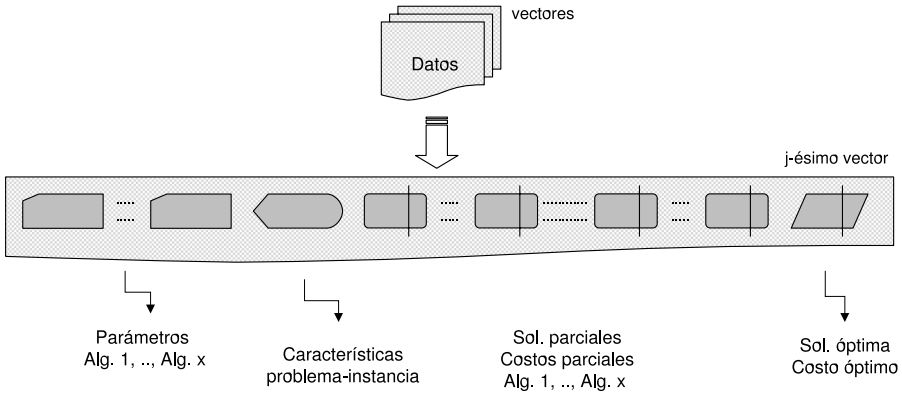



**Figura 2.** Estructura de los datos

| $p_{1,1}$ | .. | $p_{1,par_1}$ | ... | $p_{x,1}$ | .. | $p_{x,par_k}$ | $c_1$ | .. | $c_z$ | $cos_{1,1}$ | .. | $cos_{1,u}$ | ... | $cos_{x,1}$ | .. | $cos_{x,u}$ | $cos_{opt}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $val$ | .. | $val$ | ... | $val$ | .. | $val$ | $val$ | .. | $val$ | $val$ | .. | $val$ | ... | $val$ | .. | $val$ | $val$ |
| $val$ | .. | $val$ | ... | $val$ | .. | $val$ | $val$ | .. | $val$ | $val$ | .. | $val$ | ... | $val$ | .. | $val$ | $val$ |
| $val$ | .. | $val$ | ... | $val$ | .. | $val$ | $val$ | .. | $val$ | $val$ | .. | $val$ | ... | $val$ | .. | $val$ | $val$ |
| .. | .. | .. | ... | .. | .. | .. | .. | .. | .. | .. | .. | .. | ... | .. | .. | .. | .. |

**Cuadro 1.** Tabla de los datos obtenidos

### 2.2. Una técnica de minería de datos para el modelado

Como sabemos que los datos de partida con los que tenemos que trabajar contienen observaciones imperfectas (como por ejemplo, algunas características del problema-instancia estarán etiquetadas como “dificil”, “fácil”, etc) nos centramos en aquellas técnicas que permiten un mayor y más completo tratamiento de la información imperfecta y que pueda trabajar con atributos heterogéneos.

Una de las técnicas que permite el tratamiento de atributos desconocidos, atributos nominales que presentan valores inciertos desde el punto de vista probabilista, atributos continuos expresados mediante intervalos crisp (imprecisión crisp) y que incorpora el tratamiento de atributos expresados mediante valores

difusos es la técnica EMFGN basada en modelos de mezclas, [2,10]. Además, esta técnica resulta interesante debido a que obtiene modelos que capturan dependencias globales (entre todos los atributos), es decir, permite inferir los valores de cualquier subconjunto de atributos a partir de cualquier otro subconjunto de atributos con valores conocidos, usando el mismo modelo.

Por tanto, dadas las características comentadas anteriormente y debido a que impone menos limitaciones, nos proponemos abordar la fase de minería de datos mediante la técnica EMFGN, [2]. Esta técnica modelará el agente coordinador del sistema mediante un conjunto de reglas fuzzy.

**Generación de clusters.** La técnica EMFGN (Extended MFGN) toma como punto de partida la técnica denominada MFGN [10]. En MFGN se obtiene un modelo del sistema en base a encontrar la función de densidad conjunta que recoge las dependencias de los atributos. Esta función de densidad se obtiene encontrando una serie de componentes que capturan ejemplos con atributos independientes.

El modelo obtenido está expresado, desde el punto de vista más simple, de la siguiente forma ($C_1, C_2, ...C_l$ son los $l$ componentes (clusters) factorizados):

$$p(z) = p(z^1, z^2, ..., z^n) = \sum_{i=1}^{l} P\{C_i\} p(z/C_i) = \sum_{i=1}^{l} P\{C_i\} \prod_{j=1}^{n} p(z^j/C_i)$$

donde

- $z$ es un ejemplo representado con un vector de n atributos.
- $P\{C_i\}$ es el peso del i-ésimo componente.
- Para un atributo $z^j$ continuo, $p(z^j/C_i)$ es una Normal $N(z^j, \mu_i^j, \sigma_i^j)$.
- Para un atributo $z^j$ discreto o nominal, $p(z^j/C_i) = \sum_\omega t_{i\omega}^j N(z^j, \omega)$ donde $t_{i\omega} = P(z^j = \omega/C_i)$ y $N(z^j, \omega)$ es una delta de Dirac.

El aprendizaje del modelo se lleva a cabo maximizando la verosimilitud de los ejemplos disponibles como representativos del sistema. Es decir, si disponemos de una base de datos $S$ con $M$ ejemplos, la verosimilitud de los ejemplos está dada por:

$$p(S; \theta) = \prod_{k=1}^{M} p(S^{(k)}; \theta)$$

donde $\theta$ es una aproximación del vector de parámetros que maximiza la verosimilitud de los datos de $S$.

Este vector se obtiene mediante el algoritmo EM Extendido (EEM) [10] que se basa en el bien conocido algoritmo EM [9] (Expectation-Maximisation) que

iterativamente modifica los parámetros de la mezcla. Comienza con una inicialización aleatoria del conjunto de parámetros y los pasos E y M se repiten hasta que la verosimilitud de los ejemplos no mejora. En el paso E se obtiene la verosimilitud de que el j-ésimo ejemplo haya sido generado por el i-ésimo componente de la mezcla y en el paso M se vuelven a estimar los parámetros de la mezcla con los valores de verosimilitud obtenidos en el paso anterior. Este algoritmo básico fue modificado para incluir el aprendizaje de los parámetros de la mezcla a partir de datos con valores expresados con incertidumbre probabilística y con valores desconocidos, dando lugar al algoritmo EM Extendido [10].

En la técnica EMFGN, que será la que empleemos en la fase de minería de datos para modelar el coordinador, se sitúa el modelo de mezclas presentado anteriormente en una teoría más general como es la Teoría de Evidencias de Dempster-Shafer, con la idea de poder interpretar el modelo probabilístico anterior como una función de masas. Esto nos permitirá introducir en el aprendizaje de la técnica datos con valores expresados con incertidumbre subjetiva (probabilidades imprecisas), con imprecisión no difusa (intervalos clásicos) y con imprecisión difusa (etiquetas lingüísticas) [2]. Esta es una de las características principales de EMFGN, que permite el tratamiento de datos expresados con incertidumbre (tanto objetiva como subjetiva) e imprecisión (tanto difusa como no difusa). Además, permite trabajar con atributos discretos o nominales y con atributos continuos.

El aprendizaje en la técnica EMFGN se realiza utilizando el algoritmo EM Extendido Fuzzy (FEEM). Este algoritmo está basado en el algoritmo EM Extendido. Éste último algoritmo es modificado en [2] dando lugar al algoritmo EM Extendido Fuzzy. Básicamente este algoritmo modifica la forma de obtener la verosimilitud en el paso E y la de obtener el valor medio y la dispersión de cada atributo en cada componente en el paso M.

Con la expresión del modelo anteriormente comentada, podemos inferir el valor desconocido de cualquier atributo de un ejemplo, dado el valor conocido del resto de atributos. Esta es otra de las principales características del modelo, ya que la obtención de la densidad conjunta $p(z)$, captura la dependencia global entre los atributos por lo que no es necesario realizar un aprendizaje particular para cada dependencia parcial de un atributo o conjunto de atributos con respecto a los demás.

Podemos decir que en el marco de EMFGN, el sistema queda modelado mediante un conjunto de $l$ clusters, donde cada cluster agrupa aquellos ejemplos con independencia en sus atributos. De esta forma, el conjunto de clusters que se obtiene tienen la siguiente estructura:

$$P\{C_i\} \prod_{j=1}^{n} p(z^j/C_i)$$

Con EMFGN no se obtienen directamente reglas fuzzy que modelen el comportamiento de nuestro sistema, sino que vamos a obtener un conjunto de clusters fuzzy. Por lo tanto, es necesario hacer una interpretación de este conjunto de clusters para generar el conjunto de reglas fuzzy que modela el sistema.

**Obtención de reglas.** Las reglas utilizadas para describir el comportamiento del sistema cooperativo serán del tipo siguiente (las llamadas reglas del tipo Mamdani, Fig. 3):

"si <proposición fuzzy> entonces <proposición fuzzy>"

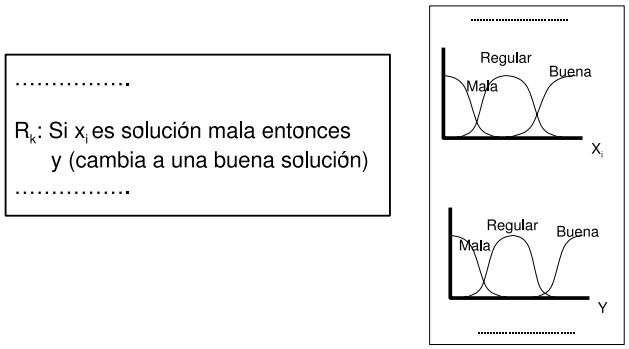



**Figura 3.** Tipos de reglas fuzzy utilizadas como modelo

De forma general, a partir de la descripción conjunta obtenida por la técnica EMFGN podemos inferir cualquier valor o conjunto de valores a partir de cierta información de entrada. Los valores inferidos y los valores de entrada serán el consecuente y antecedente respectivamente de una posible regla. Es decir, si dada la información de entrada $A$, usando EMFGN, inferimos $B$, construimos una regla de la forma si "modificador de $A$" entonces "modificador de $B$". Por ejemplo, se podría inferir el valor de los atributos "parámetros del algoritmo" dado que la "solución es buena" para la instancia del problema que estamos resolviendo. Es fácil, a partir de esta información, construir reglas para cada algoritmo del tipo:

"si solución=¬buena entonces parámetros_algoritmo= posibles valores"

donde posibles valores son etiquetas lingüísticas obtenidas mediante EMFGN.

Obtendremos, entre otras, reglas que modelan el siguiente conocimiento:

- Reglas que nos indiquen qué algoritmo debe ser reiniciado a una solución que parece ser la mejor en el momento actual.

- Reglas que nos indiquen en cuánto debe de modificarse el comportamiento de un algoritmo con respecto a su búsqueda hacia la solución.
- Reglas que nos indiquen qué algoritmo debe ser reiniciado con nuevos parámetros.
- Reglas que nos indiquen cuando debemos de parar; o porque se han excedido los recursos disponibles o porque se ha obtenido una solución satisfactoria.
- Reglas que nos indiquen, en el caso de poder aplicar más de una regla del sistema, qué regla debe de ser seleccionada.

## 3. Conclusiones y trabajos en desarrollo

Hemos presentado, la descripción de las dos primeras fases de un proceso de extracción de conocimiento para el modelado del coordinador de un sistema metaheurístico cooperativo. A partir de este esquema general, como trabajos en desarrollo está la construcción e implantación de un prototipo que lleve a cabo, de forma concreta, cada una de estas fases a partir del conjunto de datos obtenidos de distintos problemas y sus correspondientes instancias.

El problema tipo elegido es el problema de la mochila y un conjunto de instancias con distintos grados de dificultad [8]. Las metaheurísticas utilizadas para resolver las distintas instancias son: algoritmo genético, simulated annealing, búsqueda tabú y colonia de hormigas. Las base de datos en construcción consta de todos los resultados, comentados anteriormente, obtenidos de las resoluciones (para distintos parámetros) de las distintas instancias por cada una de las metaheurísticas. Una vez terminada la base de datos le aplicaremos la técnica EMFGN para obtener un conjunto de reglas que nos sirva para modelar el comportamiento cooperativo de las distintas metaheurísticas.

## Agradecimientos

Los autores agradecen al “Ministerio de Educación y Ciencia” y al “Fondo Europeo de Desarrollo Regional” (FEDER) por el soporte, bajo el proyecto TIN2005-08404-C04-02, para el desarrollo de este trabajo.

## Referencias

1. Garrido, M.C., Cadenas, J.M., Ruiz, A.: Una estructura de almacenamiento en memoria para datos imperfectos. Documento Técnico Dpto. Informática y Sistemas, Universidad de Murcia (Spain) TR DIS 3/98 (1998) 1–11
2. Cadenas, J.M., Garrido, M.C.: Imperfección Explícita versus adaptación de la información en MFGN extendido. In Proceedings XI Congreso Español sobre Tecnologías y Lógica Fuzzy, León (Spain) (2002) 547–552
3. Cadenas, J.M., Garrido, M.C., Hernández, J.J.: Improving fuzzy pattern matching techniques to deal with non discrimination ability features. In Proceedings of the IEEE Internacional Conference on System, Man and Cybernetics, Netherlands, vol. 6 (2004) 5708–5713

4. Cadenas, J.M., Garrido, M.C., Hernández, L.D., Muñoz, E.: Towards the definition of a Data Mining process based in Fuzzy Sets for Cooperative Metaheuristic systems. In Proceedings of the 11th Information Processing and Management of Uncertainty in Knowledge-Based systems Internacional Conference (IPMU 2006), Paris, Francía (2006) (por aparecer)
5. Cruz, C.A.: Estratégias coordinadas paralelas basadas en soft-computing para la solución de problemas de optimización. Tesis doctoral del Dpto. de Ciencias de la Computación e Inteligencia Artificial, Universidad de Granada (2005)
6. Grabisch, M.: Fuzzy integral for classification and feature extraction. In M. Grabisch, et al., eds., Fuzzy Measures and Integrals: Theory and Applications, Physica-Verlag (2000) 415–434
7. Inza, I., Larrañaga, P., Sierra, B.: Feature Subset Selection By Estimation of Distribution Algorithms. In P. Larrañaga et al., eds, Genetic Algorithms and Evolutionary Computation, Kluwer Academic Publishers (2002) chapter 13, 269–293
8. Kellerer, H., Pferschy, U., Pisinger, D.: Knapsack Problems. Springer Verlag (2004)
9. McLachlan, G.L., Krishnan, T.: The EM Algorithm and Extensions. Wiley Series in Probability and Statistics, (1997)
10. Ruiz, A., López de Teruel, P.E., Garrido, M.C.: Probabilistic Inference from Arbitrary Uncertainty using Mixtures of Factorized Generalized Gaussians. Journal of Artificial Intelligent Research **9** (1998) 167–217

# Localización en redes mediante heurísticas basadas en soft-computing

M. José Canós, Carlos Ivorra y Vicente Liern

Dep. Matemáticas para la Economía y la Empresa,
Universitat de València
maria.j.canos@uv.es, carlos.ivorra@uv.es, vicente.liern@uv.es

**Resumen.** Un problema de gran importancia para una empresa es localizar adecuadamente los centros desde donde deberá servir su producto a diferentes puntos de demanda. Este tipo de problemas han sido muy estudiados desde un punto de vista clásico, aunque los problemas reales obligan a introducir flexibilidad en los modelos. En otros trabajos hemos desarrollado el problema fuzzy de la *p*-mediana que flexibiliza de restricción sobre la demanda que debe ser atendida. Aquí presentamos y comparamos diversos procedimientos heurísticos para resolver este problema, teniendo en cuenta sus particularidades respecto al caso clásico.

## 1 Introducción

El problema clásico de la *p*-mediana consiste en localizar sobre una red *p* centros de servicio que satisfagan las demandas de los vértices minimizando el coste [8, 10, 11]. Una de las hipótesis es que el coste es directamente proporcional a las distancias recorridas y a la cantidad de producto transportada. Hakimi demostró [10, 11] que siempre existe una solución óptima en la que todos los centros de servicio están ubicados en vértices y la demanda de cada vértice es atendida por su centro de servicio más cercano.

En general, el objetivo del sector público suele ser maximizar los beneficios sociales, mientras que el objetivo del sector privado persigue maximizar los beneficios económicos. Por esta razón una empresa privada puede que prefiera dejar sin atender una pequeña parte de la demanda si esto le proporciona una reducción significativa en los costes. Para responder a esta necesidad de la empresa privada, planteamos el problema fuzzy de la *p*-mediana en el que suponemos que la demanda de cada vértice es una cantidad fuzzy, lo que implica que la restricción de la demanda es flexible (ver [3, 7] para más detalles). El decisor establece la máxima demanda $p_f$ que puede dejar de atenderse y la reducción del coste de transporte $p_g$ que consideraría totalmente satisfactoria. Estos parámetros determinan las funciones de pertenencia de dos conjuntos fuzzy. Si llamamos $\lambda$ al grado de pertenencia a la intersección de estos dos conjuntos, entonces $\lambda$ puede interpretarse como el grado global de satisfacción de la solución, y el objetivo del problema fuzzy de la *p*-mediana es maximizar $\lambda$. El planteamiento de este problema [3] es el siguiente:

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 689-698, 2006.

(FP) Encontrar $(x_{ij}, y_i)$

$$\text{s.a.} \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}x_{ij} \mathrel{\tilde{<}} z^* \quad (1)$$

$$\sum_{i=1}^{n} x_{ij} \le w_j \qquad 1 \le j \le n \quad (2)$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij} \mathrel{\tilde{>}} \sum_{j=1}^{n} w_j \quad 1 \le j \le n \quad (3)$$

$$0 \le x_{ij} \le w_j\, y_j \qquad 1 \le i, j \le n \quad (4)$$

$$\sum_{i=1}^{n} y_i = p \quad (5)$$

$$y_i \in \{0,1\}, \qquad 1 \le j \le n \quad (6)$$

donde $z^*$ es el óptimo crisp, $x_{ij}$ es la demanda del vértice $v_j$ atendida por un centro de servicio ubicado en el vértice $v_i$, $y_i$ es 1 si se instala un centro de servicio en vértice $v_i$ y 0 en otro caso, $w_j$ es la demanda del vértice $v_j$, y $d_{ij}$ es la distancia desde $v_i$ hasta $v_j$. La primera condición refleja el hecho de que la solución fuzzy debe ser mejor que la crisp y la tercera condición introduce en el modelo el concepto de factibilidad fuzzy.

Si suponemos funciones de pertenencia lineales $\mu_f$ y $\mu_g$, para las condiciones fuzzy, el modelo anterior puede expresarse mediante el programa siguiente:

(PF2) Max $\lambda$

s.a

$$\lambda + \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{d_{ij}}{p_g} x_{ij} \le \frac{z^*}{p_g}$$

$$\lambda - \sum_{i=1}^{n}\sum_{j=1}^{n} \frac{1}{p_f} x_{ij} \le 1 - \frac{1}{p_f}\sum_{j=1}^{n} w_j$$

$$0 \le x_{ij} \le w_j y_i \qquad 1 \le i, j \le n$$

$$\sum_{i=1}^{n} x_{ij} \le w_j \qquad 1 \le j \le n$$

$$\sum_{i=1}^{n} y_i = p,$$

$$y_i \in \{0,1\} \qquad 1 \le i \le n$$

donde $p_f$ y $p_g$ representan las tolerancias que el decisor está dispuesto a asumir para la demanda que deja de servir y la disminución de costes, respectivamente. Para resolverlo podemos construir varios algoritmos específicos (ver Fig 1).

En [3, 4] propusimos dos algoritmos exactos para resolver el problema: un modelo de programación matemática y un algoritmo de enumeración explícita. Sin embargo, como en el problema crisp, nuestros algoritmos sólo son adecuados para instancias de tamaño pequeño y medio. Para problemas reales se hacen necesarias técnicas más eficientes, y por ello hemos desarrollado varios métodos heurísticos. En [5] propusimos un heurístico de intercambio que funciona adecuadamente en instancias medias y grandes. No obstante, el coste óptimo crisp debía ser conocido, o al menos una aproximación. Esto implica resolver un problema NP-hard adicional, de modo que hemos diseñado algoritmos (de intercambio y genéticos) que permiten resolver ambos

problemas simultáneamente, lo que reduce el coste computacional total [6, 7]. En este trabajo presentamos el estado actual de nuestros algoritmos y comparamos su eficacia.

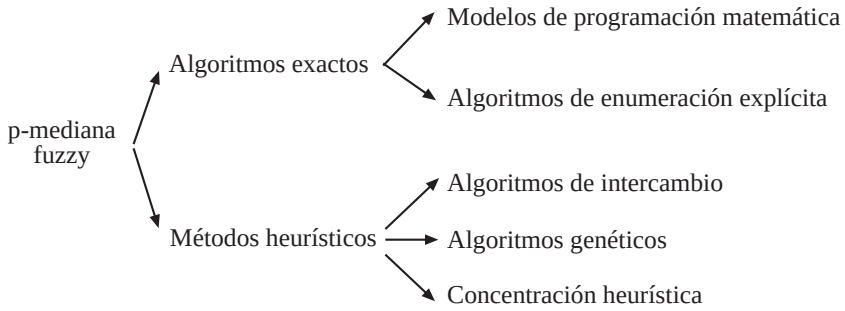



**Fig. 1.** Algoritmos para el problema fuzzy de la *p*-mediana

## 2 El problema fuzzy de la *p*-mediana

El problema crisp de la p-mediana se puede reducir a un problema combinatorio gracias al teorema de optimalidad en los vértices de Hakimi [11]. Sin embargo, esto no es suficiente en el caso fuzzy porque, aunque nos limitemos a buscar soluciones con los centros de servicio ubicados en los vértices, todavía quedan como variables continuas las cantidades de demanda servidas a cada vértice. En este epígrafe veremos que esta reducción también es posible en el caso fuzzy y es básica para todos nuestros algoritmos.

Si V es un conjunto de $p$ vértices, definimos $\lambda(V)$ como el mayor grado global de satisfacción que puede obtenerse con una solución cuyos centros de servicio estén ubicados en V. Así, calcular $\lambda$ (V) significa encontrar las reducciones de demanda más satisfactorias para una localización dada. Es fácil ver que la solución óptima del problema fuzzy de la *p*-mediana consiste en localizar los centros de servicio en el conjunto V que maximiza $\lambda(V)$. De este modo, la resolución del problema se descompone en dos partes:

- Encontrar un algoritmo para calcular $\lambda(V)$ a partir de V.
- Resolver el problema combinatorio para encontrar el conjunto V óptimo.

En [5] proponemos un algoritmo eficiente para calcular $\lambda(V)$ que puede incorporarse a cualquiera de nuestros métodos exactos y heurísticos. Su construcción se basa en razonamientos geométricos y tiene como punto de partida el teorema siguiente:

**Teorema 1** De entre todas las soluciones $x_{ij}$ *e* $y_j$ correspondientes a un mismo conjunto de vértices $V_0$ y una misma reducción total de demanda $x$,

1) La máxima satisfacción global se alcanza en las soluciones en las que el grado de mejora del objetivo es máximo (sin tener en cuenta el grado de factibilidad).

2) Estas soluciones son las que dejan por cubrir la demanda correspondiente a los vértices más alejados de los centros de servicio.

## 3 Búsqueda local con métodos de descenso

En [5] proponemos un algoritmo de intercambio para el problema fuzzy de la *p*-mediana basado en el algoritmo de Teitz y Bart [12]. Buscamos un conjunto de *p* nodos $V_0$ con la máxima satisfacción $\lambda(V_0)$. Para ello definimos el entorno de $V_0$ como la familia de todos los conjuntos de nodos obtenidos a partir de $V_0$ mediante un único intercambio. Este algoritmo funciona adecuadamente cuando el valor inicial de $z^*$ es el óptimo crisp o una buena aproximación. Sin embargo, si tratamos de aplicarlo usando como $z^*$ el mejor valor obtenido en cada paso, sin partir de una buena aproximación, las soluciones con las que termina el algoritmo tienen un alto nivel de satisfacción global $\lambda$ pero que resulta ser ficticio, porque la aproximación de $z^*$ no ha mejorado significativamente en el proceso. Pese a ello sería deseable no tener que resolver dos problemas NP-hard [9]. Aquí presentamos una modificación que busca simultáneamente buenas aproximaciones del coste óptimo crisp $z^*$ y de la máxima satisfacción global $\lambda$ La clave para ello es no fijar de antemano la máxima reducción de coste a la que se aspira, $p_g$, sino el mínimo coste reducido *DC* que se espera obtener, de modo que en el algoritmo se calcula $p_g = z^* - DC$ cada vez que se actualiza $z^*$.

Notemos que si no conocemos el coste crisp es más razonable pedir al decisor que fije un coste deseado final, *DC*, en lugar de una reducción de coste sin saber el punto de partida de la reducción. Técnicamente, el efecto es que las soluciones intermedias que obtiene el algoritmo no son sobrevaloradas y se evita así la convergencia prematura.

El algoritmo debe resolver además una dificultad adicional que no tiene equivalente en problemas de optimización combinatoria crisp. En este tipo de problemas es poco frecuente que la función objetivo tome el mismo valor en todos los elementos del entorno de una solución. Sin embargo, en nuestro problema fuzzy, si estamos considerando una localización $V_0$ cuyo coste de transporte $z$ sea mucho mayor que $z^*$, puede ocurrir que $\lambda$ tome el valor 0 en $V_0$ y en todas las localizaciones cercanas, lo que haría parar al heurístico. Para evitar que el algoritmo se detenga prematuramente por esta causa, establecemos que, mientras $\lambda$ tome el valor 0, la regla de cambio será saltar a la localización con menor coste $z$, en lugar de la que tenga mayor $\lambda$.

Un esquema del algoritmo es el siguiente:

**Inputs:** $D = [d_{ij}]$, $w_j$, $p_f$, $DC$.

PASO 0: Seleccionar una localización $V_0$ , calcular $z^* = z(V_0)$ , $p_g = z^* - DC$ y $\lambda(V_0)$.

PASO 1: Para cada localización $V$ en el entorno de $V_0$ :
Calcular $z(V)$. Si $z(V) < z^*$ asignar $z^* = z$ (V) y $p_g = z^* - DC$.
Calcular $\lambda(V)$.

PASO 2: Llamar $V_1$, $V_2$ a las localizaciones con mayor $\lambda(V_1)$ y menor $z(V_2)$ r spectivamente.

PASO 3: Si $\lambda(V_1) = 0$, hacer $V_0 = V_2$ e ir al paso 1.

PASO 4: Si $\lambda(V_1) \leq \lambda(V_0)$ parar.
PASO 5: Si $\lambda(V_1) > \lambda(V_0)$, hacer $V_0 = V_1$ e ir al paso 1.
**Outputs:** $V_0$, $\lambda^*$, $\lambda(V_0)$.

También hemos trabajado con una variante del algoritmo consistente en saltar a la primera localización $V_1$ encontrada en el entorno de $V_0$ que mejora la satisfacción global de $V_0$. Para distinguir ambas versiones llamaremos *estrategia 1* a la que se ha descrito en el algoritmo y *estrategia 2* a esta versión modificada.

## 4 Algoritmos genéticos

En este epígrafe describimos un algoritmo genético diseñado para nuestro problema [6]. Primero especificaremos las opciones que hemos elegido y luego presentaremos el algoritmo.

*a*) *Codificación*

Hemos tenido en cuenta los experimentos computacionales en los que la representación viene dada por cromosomas 0-1 de longitud *n* o cromosomas de longitud *p.* En esta última, los genes corresponden a los vértices en los que se localiza un centro de servicio abierto [2]. En este trabajo, Bozkaya *et al*. prueban que resulta más eficiente la representación con cromosomas de longitud *p*, y por tanto esta será la elección que realicemos en este trabajo. Es evidente que no se permite la repetición de genes dentro de una misma solución.

*b*) *Tamaño de la(s) población(es)*

Vamos a considerar y comparar dos variantes del algoritmo. Una de ellas empieza con una única población de *M* cromosomas, mientras que la otra trabaja simultáneamente con dos poblaciones de *M/2* cromosomas cada una. Esto permitirá una posterior comparación de los resultados. Elegimos el valor de *M* de forma que la probabilidad de que un vértice no aparezca en ningún cromosoma de la población sea suficientemente pequeña.

*c) Elección de los padres*

Cada cromosoma tiene una probabilidad asociada que utilizamos para elegir dos padres. Esta probabilidad es proporcional a una función que refleja la bondad del cromosoma (*fitness*). En el problema crisp, donde el objetivo es minimizar el coste *z*, la función utilizada es 1/*z*. Análogamente, en el problema fuzzy, donde el objetivo es maximizar $\lambda$, hemos escogido la función $1/(1-\lambda)$.

*d*) *Crossover*

Una vez elegidos los dos padres, el cromosoma resultante se genera mediante un operador de cruce (*crossover*). Hemos considerado dos posibilidades. En el primer operador generamos un punto de corte aleatorio *c* entre 1 y *p*-1, y copiamos

los primeros *c* genes del primer padre en el cromosoma resultante. Los restantes *p-c* genes se eligen de entre los genes del segundo padre que todavía no están en el hijo. El segundo operador consiste en tomar los genes comunes a ambos padres y completar el cromosoma resultante con genes elegidos aleatoriamente de entre los genes no comunes a ambos progenitores.

*e*) *Sustitución generacional*

El tipo de sustitución depende del número de soluciones comunes en dos generaciones sucesivas *OL*. Así, cuando *OL*=1, generamos *M*-1 cromosomas para reemplazar toda la población excepto el mejor cromosoma., y cuando *OL*=*M*-1 cada cromosoma resultante sustituye a uno de los padres tan pronto es generado. Hemos probado nuestro algoritmo para *OL*=1,…, *M-1*, manteniendo siempre el mejor cromosoma de la población.

*f*) *Invasión y mutación*

Cada cromosoma puede mutar aleatoriamente con una cierta probabilidad *R*. Asimismo las poblaciones sufren invasiones periódicas. En el caso de una población, la invasión consiste en sustituir una parte de la población por cromosomas generados aleatoriamente. En el caso de dos poblaciones, la primera población, cuyo objetivo es la maximización del nivel de satisfacción global, es invadida por cromosomas de la segunda población (ver Fig. 2), mientras que la segunda población sufre invasiones como en el caso de una población.

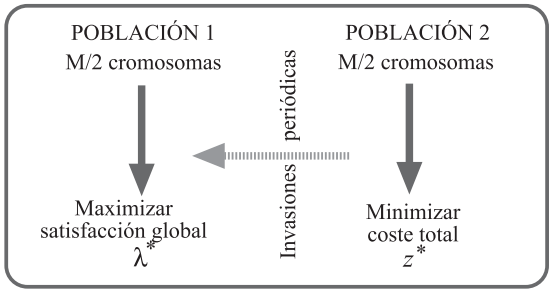



**Fig. 2**. Esquema de las invasiones en el algoritmo genético con dos poblaciones

Si suponemos que se produce una invasión cada I generaciones, el número de invasores vendrá dado por:

$$Invasores = \frac{M \cdot RR \cdot I}{100}$$

donde *RR* es el porcentaje medio de invasores en cada generación.

*g*) *Criterio de parada*

Como criterio de parada fijamos el parámetro *UG* de forma que el algoritmo parará cuando el valor del nivel global de satisfacción (y del coste, en su caso) no cambie durante *UG* generaciones.

*h*) *Tolerancia*

Al igual que en algoritmo de intercambio calcularemos $p_g$ como $p_g = z^*\text{-}DC$.

Con todo lo anterior, nuestro algoritmo es el siguiente:

PASO 1. Inicializar *N, M*, *OL*, *R*, *UG*, *I*, *RR*.

PASO 2. Generar aleatoriamente *N* poblaciones de *M* cromosomas.

PASO 3. Repetir hasta que el mejor valor de la satisfacción global permanezca sin cambios durante *UG* generaciones.

a) Calcular la probabilidad de ser seleccionado para cada cromosoma.
b) Repetir *M-OL* veces: Seleccionar dos cromosomas de acuerdo con las probabilidades calculadas en el paso anterior. Utilizar un operador para generar un nuevo cromosoma. Aplicar una mutación al cromosoma resultante con probabilidad *R*.
c) Sustituir *M-OL* cromosomas originales por cromosomas generados.
d) Aplicar, si procede, el correspondiente procedimiento de invasión.

## 5 Comparación entre algoritmos genéticos y de intercambio

Para comparar los algoritmos genéticos y los de intercambio hemos utilizado la base de datos [1], que nos ha servido para testar los problemas de la *p*-mediana en todos nuestros trabajos. Se trata de una colección de problemas que contiene redes desde 100 hasta 900 nodos, y la cantidad de centros de servicios que deben localizarse oscila entre 5 y 90. Los resultados que presentamos se han obtenido repitiendo 50 veces cada problema en un ordenador Macintosh PowerPC G5 1.5 GHz con diferentes valores de los parámetros.

A continuación presentamos (Tablas 1 y 2) los resultados obtenidos para algunos de estos problemas: El problema pmed21 es un ejemplo de problema sencillo, puesto que requiere localizar sólo 5 centros; el problema pmed32 es de tamaño medio, por el número de vértices, y el problema pmed10 es más complejo porque el número de centros es grande sin estar próximo al número de vértices.

En cada fila de la Tabla 1 aparecen, en este orden, el problema a resolver, el número de vértices de la red, el número de centros de servicio predeterminado, el valor óptimo de la *p*-mediana crisp, el número de poblaciones, el tipo de cruce y los resultados del algoritmo genético: el valor medio del coste crisp para la localización y el valor medio del grado de satisfacción global.

**Tabla 1.** Resultados computacionales de los algoritmos genéticos

| Problema | n | p | óptimo crisp | Num. Pob. | cruce | z* | λ* |
|---|---|---|---|---|---|---|---|
| pmed21 | 500 | 5 | 9138 | 1 | 1 | 9138 | 33.89 |
| | | | | 1 | 2 | 9138 | 33.89 |
| | | | | 2 | 1 | 9138 | 33.89 |
| | | | | 2 | 2 | 9138 | 33.89 |
| pmed32 | 700 | 10 | 9297 | 1 | 1 | 9301 | 37.12 |
| | | | | 1 | 2 | 9301 | 37.01 |
| | | | | 2 | 1 | 9297 | 37.12 |
| | | | | 2 | 2 | 9301 | 37.12 |
| pmed10 | 200 | 67 | 1255 | 1 | 1 | 1284 | 64.23 |
| | | | | 1 | 2 | 1272 | 64.23 |
| | | | | 2 | 1 | 1261 | 64.37 |
| | | | | 2 | 2 | 1272 | 64.09 |

En cada fila de la Tabla 2 aparecen, de izquierda a derecha, el problema a resolver, el número de vértices, el número de centros de servicio, el valor óptimo de la *p*-mediana crisp, la estrategia utilizada en el algoritmo de intercambio, el valor medio del coste crisp para la localización y el valor medio del grado de satisfacción global.

**Tabla 2.** Resultados computacionales de los algoritmos de intercambio

| Problema | n | p | óptimo crisp | estrategia | z* | λ* |
|---|---|---|---|---|---|---|
| pmed21 | 500 | 5 | 9138 | 1 | 9177 | 33.89 |
| | | | | 2 | 9177 | 33.89 |
| pmed32 | 700 | 10 | 9297 | 1 | 9301 | 37.41 |
| | | | | 2 | 9301 | 37.21 |
| pmed10 | 200 | 67 | 1255 | 1 | 1272 | 61.05 |
| | | | | 2 | 1270 | 62.31 |

Para mostrar más claramente las diferencias entre los distintos algoritmos representamos en las gráficas 2 y 3 la desviación de cada una de las entradas de la tabla respecto del mayor valor de $\lambda^*$ y $z^*$, respectivamente. Las seis líneas corresponden a las cuatro variantes del algoritmo genético (combinando el número de poblaciones con los operadores de cruce) y los dos algoritmos de intercambio con las estrategias 1 y 2.

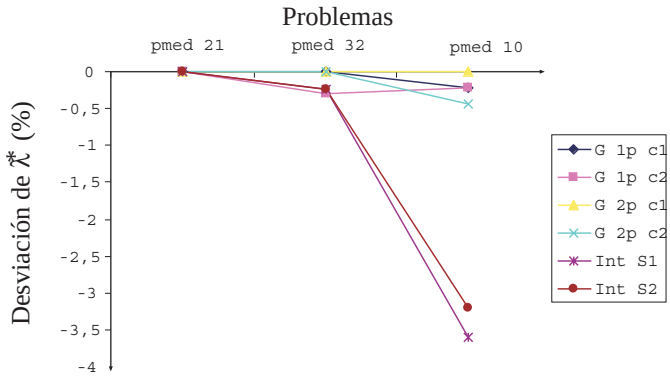



**Fig. 2.** Desviación respecto del mejor λ*

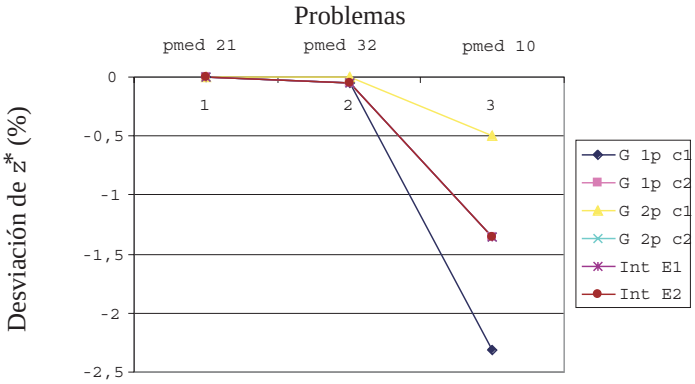



**Fig. 3.** Desviación respecto del mejor z*

Las conclusiones que hemos extraído de las pruebas realizadas con los 40 problemas citados anteriormente pueden resumirse en los cuatro puntos siguientes:

a) Los algoritmos genéticos son más estables, es decir no presentan diferencias significativas en cuanto a tiempos de CPU o número de veces que se alcanza el óptimo. De todos modos destacamos que el algoritmo con dos poblaciones encuentra más a menudo los óptimos crisp y fuzzy.
b) El número de centros de servicio afecta notablemente a la velocidad de los algoritmos de intercambio. Son bastante rápidos tanto para valores pequeños de p como para valores próximos al número de vértices de la red, mientras que para valores intermedios la velocidad desciende.
c) Para problemas pequeños no hay diferencias significativas entre los distintos algoritmos, mientras que para problemas mayores los algoritmos genéticos son más rápidos que los de intercambio.
d) Todas las versiones de los algoritmos genéticos obtienen el óptimo fuzzy con frecuencias similares, pero el de dos poblaciones alcanza el óptimo crisp más a menudo.

Más en general, este análisis puede considerarse como un estudio previo, ya que las ideas y estrategias empleadas en el diseño de estos algoritmos se pueden aplicar igualmente para tratar de forma eficiente versiones fuzzy de otros problemas clásicos o incluso problemas de naturaleza biobjetivo.

## Referencias


1. Beasley, J. E. (2004) 'OR-Library' http://people.brunel.ac.uk/~mastjjb/jeb/orlib/pmedinfo.html
2. Bozkaya, B, Zhang, J, Erkut, E. 'An Efficient Genetic Algorithm for the p-Median Problem', in Drezner, Z., Hamacher, H.W. (eds.) Facility Location. Applications and Theory, Springer (2002).
3. Canós, M.J., Ivorra, C., Liern, V. 'An exact algorithm for the fuzzy p-median problem', European Journal of Operational Research, Vol. 116 (1999) pp. 80-86.
4. Canós, M.J., Ivorra, C., Liern, V. 'The fuzzy p-median problem: A global analysis of the solutions', European Journal of Operational Research, Vol. 130 (2001) pp. 430-436.
5. Canós, M.J., Ivorra, C., Liern, V. 'Finding Satisfactory Near-Optimal Solutions in Location Problems', in Verdegay J. L. (editor), Fuzzy Sets Based Heuristics for Optimization, Springer-Verlag, (2003).
6. Canós, M.J., Ivorra, C., Liern, V. 'Genetic Algorithms for the Fuzzy p-Median Problem', Advances in Decision Technology and Intelligent Information Systems, Volume IV, (ed. K. Engemann and G. Lasker), The International Institute for Advanced Studies in Systems Research and Cybernetics, Windsor, Canada, pp. 56-60 (2003).
7. Canós, M.J., Ivorra, C., Liern, V. 'The fuzzy p-median problem', International Journal of Technology, Policy and Management, vol. 4, (2004) pp. 365-381.
8. Daskin, M.S. Network and Discrete Location: Models, Algorithms and Applications, Wiley (1995).
9. Garey, M.R., Johnson, D.S. Computers and Intractability: a Guide to the Theory of NP-Completeness, Freeman (1979).
10. Hakimi, S.L. 'Optimum Locations of Switching Centers and the Absolute Centers and Medians of a Graph', Operations Research, Vol. 12 (1964) pp. 450-459.
11. Hakimi, S.L. 'Optimum Distribution of Switching Centers en a Communication Network and Some Related Graph Theoretic Problems', Operations Research, Vol. 13 (1965) pp. 462-475.
12. Teitz, M.B., Bart, P. 'Heuristic Methods for Estimating the Generalized Vertex Median of a Weighted Graph', Operations Research , Vol. 16 (1968) pp. 955-951.

# Razonamiento abductivo en modelos finitos mediante $C$-tablas y $\delta$-resolución

Fernando Soler-Toscano[1], Ángel Nepomuceno-Fernández[1], Atocha Aliseda-Llera[2], and A. Liliana Reyes-Cabello[3]

[1] Departamento de Filosofía y Lógica, Universidad de Sevilla, C/ Camilo José Cela s/n, 41018 Sevilla, Spain
{fsoler, nepomuce}@us.es
[2] Instituto de Investigaciones Filosóficas, Universidad Nacional Autónoma de México, Circuito Mario de la Cueva s/n, 04510 México D.F., Mexico
atocha@filosoficas.unam.mx
[3] Departamento de Matemáticas, Universidad Nacional Autónoma de México, Circuito Exterior s/n, 04510 México D.F., Mexico
liliana@ciencias.unam.mx

**Resumen** Presentamos un acercamiento a la resolución de problemas abductivos en $C$-estructuras, estructuras que tienen un dominio de cardinalidad finita tal que para cada uno de sus elementos conocemos una constante del lenguaje cuya interpretación es dicho elemento. Empleando tablas semánticas con profundidad acotada (las $C$-tablas, una variante de las $N$-tablas introducidas en trabajos anteriores) y el cálculo de $\delta$-resolución, construimos un procedimiento efectivo para encontrar soluciones abductivas minimales dentro de la semántica propuesta.

## 1. Introducción

El razonamiento abductivo, o explicativo, se caracteriza por su proceder inverso a la deducción, desde la *conclusión* a las *premisas*. Dado cierto hecho sorprendente $\mathcal{A}$, para el que no tenemos explicación, si sabemos que desde $\mathcal{B}$ se obtiene $\mathcal{A}$, podemos suponer $\mathcal{B}$ como *hipótesis plausible*:

$$\frac{\mathcal{A}, \quad \mathcal{B} \to \mathcal{A}}{\mathcal{B}} \tag{1}$$

La regla (1), incorrecta para la lógica clásica (se trataría de la falacia de afirmación del consecuente), refleja el carácter *retroductivo* (deducción hacia atrás) de la abducción. Cada vez son más numerosas las aplicaciones que la abducción encuentra en Inteligencia Artificial: planificación, diagnosis, interpretación del discurso, asimilación de conocimientos, etc.

Cuando se intentan desarrollar formalismos para la resolución de problemas abductivos en lógica de primer orden, aparece siempre el problema de la indecidibilidad: en general, no es posible determinar si, para cierto *problema abductivo*,

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 699-711, 2006.

una hipótesis propuesta es o no una *solución abductiva* correcta. En el campo de la ASP (*Answer Set Programming*) es frecuente la reducción de ciertos problemas de satisfactibilidad en primer orden a dominios de cardinalidad finita. Una estrategia similar seguimos en este trabajo, con la particularidad de que el acercamiento a la abducción que presentamos integra dos sistemas ya acreditados dentro del panorama lógico del razonamiento explicativo, pero que hasta ahora sólo habían sido considerados por separado. Se trata de los sistemas basados en tablas semánticas y en resolución. Así, empleamos tablas semánticas para la reducción de los problemas abductivos a dominios de cardinalidad finita, y entonces, en un marco proposicional, usamos el cálculo de $\delta$-resolución, dual a la resolución estándar, para la búsqueda de soluciones. El resultado es un procedimiento abductivo que devuelve todas las soluciones abductivas *explicativas* y *minimales* (en el sentido de [1]) con una eficiencia mayor que el uso por separado tanto de las tablas semánticas como de la resolución. Estas soluciones, por lo general, no serán válidas para las versiones originales, en primer orden, de los problemas abductivos, pero sí para sus correspondientes en modelos finitos, con lo que nuestro acercamiento resulta especialmente adecuado para aplicaciones de la abducción en contextos modelables en dominios finitos.

Mediante $\mathscr{L}$ denotamos un lenguaje de primer orden sin identidad ni functores, definido con las convenciones habituales. En cuanto a la semántica, no contemplaremos asignaciones de valores a las variables libres, con lo que restringimos la interpretación a las sentencias. Dada la secuencia $t_1, \ldots, t_n$ de términos (constantes y variables) y el predicado $n$-ádico $P$, decimos que $P_{t_1,\ldots,t_n}$ y $\neg P_{t_1,\ldots,t_n}$ son literales complementarios (positivo y negativo, respectivamente). Dado cualquier literal $\lambda$, representamos mediante $\overline{\lambda}$ su complementario.

**Definición 1.** *Una* $\delta$-cláusula $\Sigma = \{\lambda_1, \ldots, \lambda_n\}$ *es un conjunto finito de literales de* $\mathscr{L}$ *tales que todos sus términos son constantes. Dada una estructura* $\mathcal{M}$, $\mathcal{M} \models \Sigma$ *syss* $\mathcal{M} \models \lambda_i$ *para todo literal* $\lambda_i$, $1 \leq i \leq n$. *Mediante* $\Diamond$ *representamos la* $\delta$*-cláusula vacía, universalmente válida.*

**Definición 2.** *Una* forma $\delta$-clausal $A = \{\Sigma_1, \ldots, \Sigma_n\}$ *es un conjunto finito de* $\delta$*-cláusulas. Dada una estructura* $\mathcal{M}$, $\mathcal{M} \models A$ *syss* $\mathcal{M} \models \Sigma_i$ *para al menos una* $\delta$*-cláusula* $\Sigma_i$, $1 \leq i \leq n$. *La forma* $\delta$*-clausal vacía es no satisfactible.*

Las definiciones anteriores son duales a las de cláusula y forma clausal del cálculo de resolución estándar. Ahora, una $\delta$-cláusula equivale a una conjunción (no disyunción) elemental de literales, y una forma $\delta$-clausal a una forma normal disyuntiva (no conjuntiva). Dada esta dualidad con las definiciones estándar, podremos obtener resultados como el teorema 2.

**Definición 3.** *Dado el conjunto* $C = \{c_1, \ldots, c_n\}$ *finito y no vacío de constantes de* $\mathscr{L}$, *definimos una* $C$-estructura *como una estructura* $\mathcal{M} = \langle \mathcal{D}, \mathcal{I} \rangle$, *con* $\mathcal{D}$ *como universo de discurso e* $\mathcal{I}$ *como función interpretación, tales que, además de los requisitos habituales, verifican*

- $|\mathcal{D}| = n$, *y*

- *Si* $i \neq j$, $1 \leq i, j \leq n$, *entonces* $\mathcal{I}(c_i) \neq \mathcal{I}(c_j)$.

**Definición 4.** *Dado cualquier conjunto de constantes* $C$ *y cualesquiera fórmulas* $\alpha, \beta \in \mathscr{L}$, *decimos que:*

- $\alpha$ *es* $C$-satisfactible *syss existe una* $C$*-estructura* $\mathcal{M}$ *tal que* $\mathcal{M} \models \alpha$.
- $\alpha$ *es* $C$-válida *syss para cualquier* $C$*-estructura* $\mathcal{M}$ *se verifica* $\mathcal{M} \models \alpha$; *en símbolos* $\models_C \alpha$.
- $\beta$ *es* $C$-consecuencia lógica *de* $\alpha$ *syss para cualquier* $C$*-estructura* $\mathcal{M}$ *se verifica que si* $\mathcal{M} \models \alpha$ *entonces* $\mathcal{M} \models \beta$; *en símbolos* $\alpha \models_C \beta$.
- $\alpha$ *y* $\beta$ *son* $C$-equivalentes *syss* $\alpha \models_C \beta$ *y* $\beta \models_C \alpha$.

*Las nociones anteriores se pueden extender, de manera natural, a conjuntos de fórmulas, así como a* $\delta$*-cláusulas y formas* $\delta$*-clausales, teniendo en cuenta las condiciones de satisfactibilidad que imponen las definiciones 1 y 2.*

**Definición 5.** *Dados el conjunto* $\Theta \subset \mathscr{L}$ *y* $\varphi \in \mathscr{L}$, *decimos que el par* $\langle \Theta, \varphi \rangle$ *es un* problema abductivo *si se verifican* $\Theta \not\models \varphi$ *y* $\Theta \not\models \neg\varphi$.

**Definición 6.** *Dado el problema abductivo* $\langle \Theta, \varphi \rangle$, *decimos que la* $\delta$*-cláusula* $\Sigma$ *es una* solución $C$-abductiva *al mismo si:*

1. $\Theta \cup \Sigma \models_C \varphi$.
2. $\Theta \cup \Sigma$ *es* $C$*-satisfactible.*
3. $\Sigma \not\models_C \varphi$.
4. *No existe ninguna* $\delta$*-cláusula* $\Sigma' \subset \Sigma$ *tal que* $\Theta, \Sigma' \models_C \varphi$.

*Mediante* $\mathcal{A}bd(\Theta, \varphi)_C$ *denotamos el conjunto de soluciones* $C$*-abductivas al problema abductivo* $\langle \Theta, \varphi \rangle$.

## 2. El cálculo de $C$-tablas

El cálculo de $C$-tablas parte de una modificación del método de las tablas semánticas [2] introducida paralelamente por [3] y [4], que ya ha sido usada con fines abductivos en [5,6], como extensión del procedimiento abductivo de [7,8]. En esta ocasión las definimos tomando como referencia, en vez de la cardinalidad de los modelos buscados, el conjunto $C$ de constantes que define la clase de $C$-estructuras en la que trataremos de comprobar si cierto conjunto de fórmulas es $C$-satisfactible.

**Definición 7.** *Dado un conjunto finito* $\Theta \subset \mathscr{L}$ *y un conjunto finito y no vacío de constantes* $C = \{c_1, \ldots, c_n\}$ *entre las que aparecen todas las de* $\Theta$, *una*

*$C$-tabla de $\Theta$, que denotamos mediante $\mathcal{T}(\Theta)_C$, es una tabla semántica que difiere de las tablas de Beth únicamente en las reglas $\gamma$ y $\delta$. Ahora:*

$$\frac{\forall x\varphi}{\begin{array}{c}\varphi(x/c_1)\\ \vdots \\ \varphi(x/c_n)\end{array}} \qquad \frac{\neg\exists x\varphi}{\begin{array}{c}\neg\varphi(x/c_1)\\ \vdots \\ \neg\varphi(x/c_n)\end{array}}$$

$$\frac{\exists x\varphi}{\varphi(x/c_1)|\ldots|\varphi(x/c_n)} \qquad \frac{\neg\forall x\varphi}{\neg\varphi(x/c_1)|\ldots|\neg\varphi(x/c_n)}$$

**Definición 8.** *La* forma $\delta$-clausal de la $C$-tabla $\mathcal{T}(\Theta)_C$, *llamada* $\mathcal{C}\delta(\mathcal{T}(\Theta)_C)$, *es la más pequeña que contiene, por cada rama abierta de $\mathcal{T}(\Theta)_C$, una $\delta$-cláusula con sus literales.*

**Lema 1.** *Dado un conjunto de literales $\Theta$ tal que no contiene ningún par de literales complementarios, si todos los términos que aparecen en $\Theta$ están contenidos en el conjunto de constantes $C$, entonces $\Theta$ es $C$-satisfactible.*

*Demostración.* Sea $\Theta$ un conjunto de literales tal como indica el enunciado del lema 1; supongamos que no es $C$-satisfactible. Entonces, por la definición 4 tenemos que no existe ninguna $C$-estructura que satisfaga simultáneamente todos los literales de $\Theta$. Sea $\mathcal{M} = \langle \mathcal{D}, \mathcal{I}\rangle$ cualquier $C$-estructura. Puesto que $\Theta$ no contiene literales complementarios, sólo puede ocurrir que haya en $\Theta$ dos literales $\varphi(c_1^1,\ldots,c_n^1)$ y $\neg\varphi(c_1^2,\ldots,c_n^2)$ tales que para algún valor de $i$ se verifique $c_i^1 \neq c_i^2$, $1 \leq i \leq n$ (ya que no pueden ser literales complementarios), pero $\mathcal{I}(c_j^1) = \mathcal{I}(c_j^2)$ para todo valor de $j$, $1 \leq j \leq n$. Sin embargo esto no puede ocurrir, puesto que conlleva que dos constantes diferentes, $c_i^1$ y $c_i^2$, tengan igual interpretación, lo cual es imposible, por ser $\mathcal{M}$ una $C$-estructura, y $c_i^1, c_i^2 \in C$. Por tanto, negamos nuestro supuesto y concluimos que $\Theta$ es $C$-satisfactible.

**Corolario 1.** *El conjunto de literales que pertenecen a cualquier rama abierta de una $C$-tabla es siempre $C$-satisfactible.*

*Demostración.* Sea $\Theta$ el conjunto de literales de una rama abierta de una $C$-tabla. Por la definición 7, todos los términos de $\Theta$ son constantes de $C$, y en $\Theta$ no hay literales complementarios. Entonces, por el lema 1, $\Theta$ es $C$-satisfactible.

**Lema 2.** *Si $\Sigma$ es el conjunto de literales que pertenecen a una rama abierta de $\mathcal{T}(\{\theta_1,\ldots,\theta_m\})_C$, entonces*

$$\Sigma \models_C \theta_1 \wedge \ldots \wedge \theta_m$$

*Demostración.* Sea $\mathcal{M} = \langle \mathcal{D}, \mathcal{I}\rangle$ una $C$-estructura que satisface $\Sigma$, conjunto de literales de una rama abierta de $\mathcal{T}(\{\theta_1,\ldots,\theta_m\})_C$. Tomemos $C = \{c_1,\ldots,c_n\}$. Probemos que $\mathcal{M}$ satisface todas las fórmulas de dicha rama, por inducción sobre su grado lógico. En el caso base son literales que, por hipótesis, son satisfechos

por $\mathcal{M}$. Supongamos que $\mathcal{M}$ satisface todas las fórmulas de la rama hasta las de grado $i$. Sea $\lambda$ una fórmula de grado $i+1$. No consideraremos el caso en que $\lambda$ es un literal negativo, pues ya sabemos que todos son satisfechos por $\mathcal{M}$. Por tanto, $\gamma$ sólo puede ser:

- Una doble negación $\neg\neg\epsilon$. Entonces, como para completar la construcción de la $C$-tabla se debió aplicar la regla de doble negación a $\lambda$, tenemos que $\epsilon$ está en la rama, y por ser su grado lógico menor o igual a $i$, $\mathcal{M} \models \epsilon$, y por evaluación de $\neg$, $\mathcal{M} \models \gamma$.
- Una fórmula de tipo $\alpha$. Entonces, sus dos componentes deben encontrarse en la rama, y por ser ambas de grado lógico menor o igual a $i$, ambas deben ser satisfechas por $\mathcal{M}$. Por evaluación de los signos lógicos de las fórmulas de tipo $\alpha$, tenemos que $\mathcal{M} \models \lambda$.
- Una fórmula de tipo $\beta$. Entonces, una de sus componentes debe estar en la rama, y por ser de grado lógico menor o igual a $i$, es satisfecha por $\mathcal{M}$. Por evaluación de los signos lógicos de las fórmulas de tipo $\beta$, $\mathcal{M} \models \lambda$.
- Una fórmula de tipo $\gamma$, como $\forall x\varphi$ (el caso en que $\lambda$ es $\neg\exists\varphi$ es similar). Entonces, como se debió aplicar la regla $\gamma$ durante la construcción de la $C$-tabla, todas las subfórmulas tipo $\varphi(x/c_j)$, $1 \leq j \leq n$, deben estar en la rama, y por ser de grado menor o igual a $i$ son satisfechas por $\mathcal{M}$. Además, como el valor de la función $\mathcal{I}$ para las constantes $c_j$ recorre todo el dominio $\mathcal{D}$ se verifica, por evaluación de $\forall$, $\mathcal{M} \models \lambda$.
- Una fórmula de tipo $\delta$, como $\exists x\varphi$ (es similar el caso en que $\lambda$ es como $\neg\forall x\varphi$). Entonces, por la aplicación de la regla $\delta$, la rama debe contener cierta subfórmula $\varphi(x/c_j)$, $1 \leq j \leq n$, que por hipótesis de inducción es satisfecha por $\mathcal{M}$. Por tanto, por evaluación de $\exists$, $\mathcal{M} \models \lambda$.

Por tanto, $\mathcal{M}$ satisface todas las fórmulas de la rama, y por ello también $\mathcal{M} \models \theta_k$, para cada valor de $k$ entre 1 y $m$. Por evaluación de $\wedge$, $\mathcal{M} \models \theta_1 \wedge \ldots \wedge \theta_m$.

**Lema 3.** *Para cualquier conjunto finito $C$-satisfactible $\Theta \subset \mathscr{L}$ se cumple:*

1. *Cualquier $C$-estructura que satisfaga $\Theta$ satisface todas las fórmulas de al menos una de las ramas de $\mathcal{T}(\Theta)_C$.*
2. *$\mathcal{T}(\Theta)_C$ es abierta.*

*Demostración.* Sea $\Theta$ un conjunto finito de fórmulas $C$-satisfactible, para cierto conjunto de constantes $C$, y $\mathcal{M} = \langle \mathcal{D}, \mathcal{I} \rangle$ una $C$-estructura que satisface $\Theta$. Demostraremos que $\mathcal{M}$ satisface todas las fórmulas de al menos una rama de $\mathcal{T}(\Theta)_C$, por inducción en el número de veces que se ha aplicado alguna regla de construcción de $C$-tablas. En el caso base, con 0 aplicaciones, la única rama de la $C$-tabla tiene sólo las fórmulas de $\Theta$, que son satisfechas por $\mathcal{M}$. Ahora supongamos que $\mathcal{M}$ satisface todas las fórmulas de cierta rama hasta la $i$-ésima aplicación de reglas. Consideremos la $(i+1)$-ésima regla que se aplica a cierta fórmula $\lambda$. En caso de que dicha regla no afecte a la rama que estamos estudiando, es trivial que la rama sigue siendo satisfecha por $\mathcal{M}$ tras su aplicación. En otro caso, veamos qué ocurre según dicha regla sea:

- La regla de doble negación. Entonces, $\lambda$ es $\neg\neg\lambda'$, y se añade a la rama $\lambda'$. Como por hipótesis $\mathcal{M} \models \lambda$ entonces, por evaluación de $\neg$, $\mathcal{M} \models \lambda'$, con lo que $\mathcal{M}$ sigue satisfaciendo todas las fórmulas de la rama.
- La regla $\alpha$. Entonces, $\lambda$ tiene la forma $\lambda_1 \wedge \lambda_2$, o bien $\neg(\lambda_1 \vee \lambda_2)$, etc., y se añaden a la rama las subfórmulas $\lambda_1$ y $\lambda_2$ o bien $\neg\lambda_1$ y $\neg\lambda_2$, etc. En cualquier caso, como por hipótesis $\mathcal{M}$ satisface $\lambda$, por evaluación de los signos lógicos, $\mathcal{M}$ satisface las dos nuevas subfórmulas, por lo que $\mathcal{M}$ sigue satisfaciendo todas las fórmulas de la rama tras la aplicación de la $(i+1)$-ésima regla.
- La regla $\beta$. Entonces $\lambda$ tiene la forma $\lambda_1 \vee \lambda_2$, o bien $\neg(\lambda_1 \wedge \lambda_2)$, etc., y entonces se divide la rema en dos y a cada una se añade una de las subfórmulas $\lambda_1$ y $\lambda_2$ o bien $\neg\lambda_1$ y $\neg\lambda_2$, etc. Como por hipótesis $\mathcal{M} \models \lambda$, por evaluación de los signos lógicos, $\mathcal{M}$ satisface al menos una de las dos nuevas subfórmulas, por lo que $\mathcal{M}$ satisface al menos una de las dos nuevas ramas tras la aplicación de la $(i+1)$-ésima regla.
- La regla $\gamma$. Entonces $\lambda$ es $\forall x\varphi$ (si $\lambda$ es $\neg\exists x\varphi$ la prueba es similar). Por tanto, al aplicar la regla $\gamma$ se añaden a la rama todas las subfórmulas $\varphi(x/c_j)$, $1 \leq j \leq n$. Pero como por hipótesis $\mathcal{M} \models \forall x\varphi$ entonces, por evaluación de $\forall$ se verifica que $\mathcal{M} \models \varphi(x/c_j)$ para cada constante $c_j$, $1 \leq j \leq n$. Por tanto, $\mathcal{M}$ satisface la rama tras la aplicación de la regla $\gamma$.
- La regla $\delta$. En este caso, $\lambda$ es $\exists x\varphi$ (si $\lambda$ es $\neg\forall x\varphi$ la prueba es similar), y por hipótesis de inducción $\mathcal{M} \models \lambda$, es decir, $\mathcal{M} \models \exists x\varphi$. Como $\mathcal{M}$ asigna a cada constante de $\mathcal{T}(\{\eta\})_C$ un elemento diferente de $\mathcal{D}$ se verifica, por evaluación de $\exists$, que para al menos una constante $c_j$, $1 \leq j \leq n$, $\mathcal{M} \models \varphi(x/c_j)$. Pero una de las nuevas ramas que surgen tras la aplicación de la regla $\delta$ contiene $\varphi(x/c_j)$, por lo que sigue habiendo una rama cuyas fórmulas son todas satisfechas por $\mathcal{M}$.

Al finalizar la construcción de la $C$-tabla habrá, por tanto, al menos una rama tal que todas sus fórmulas son satisfechas por $\mathcal{M}$, con lo que resulta probado el punto primero del lema 3. Pero además, entre las fórmulas de la rama satisfecha por $\mathcal{M}$ no puede haber literales complementarios. Por ello, la $C$-tabla es abierta, lo que prueba la segunda parte del lema.

**Teorema 1.** *Sean $\eta \in \mathscr{L}$ y $\mathcal{T}(\{\eta\})_C$ una $C$-tabla de $\{\eta\}$. Entonces,*

$$\eta \quad y \quad \mathcal{C}\delta(\mathcal{T}(\{\eta\})_C)$$

*son $C$-equivalentes.*

*Demostración.* Sea $\mathcal{M}$ una $C$-estructura que satisface $\eta$. Entonces, por el lema 3 se verifica que $\mathcal{M}$ satisface todas las fórmulas de al menos una rama de $\mathcal{T}(\{\eta\})_C$. Por tanto, $\mathcal{M}$ satisface todos los literales de dicha rama, entre los cuales no puede haber literales complementarios, con lo que son literales de una rama abierta de $\mathcal{T}(\{\eta\})_C$. Por la definición 8, tales literales constituyen una $\delta$-cláusula de $\mathcal{C}\delta(\mathcal{T}(\{\eta\})_C)$. Por las definiciones 1 y 2 tenemos que $\mathcal{M} \models \mathcal{C}\delta(\mathcal{T}(\{\eta\})_C)$.

Ahora, sea $\mathcal{M}$ una $C$-estructura que satisface $\mathcal{C}\delta(\mathcal{T}(\{\eta\})_C)$. Por las definiciones 1 y 2, $\mathcal{M}$ satisface todos los literales de al menos una rama abierta de $\mathcal{T}(\{\eta\})_C$. Por el lema 2, $\mathcal{M} \models \eta$.

## 3. Búsqueda mediante $\delta$-resolución

El cálculo de resolución [9] es posiblemente el procedimiento lógico más empleado en demostración automática de teoremas. En esta sección presentamos el cálculo de $\delta$-resolución [10] que nos servirá para obtener hipótesis explicativas para cierto conjunto de fórmulas. Comenzamos presentando la regla de $\delta$-resolución que, aunque sintácticamente idéntica, es semánticamente dual a la regla de resolución clásica, al ser las $\delta$-cláusulas equivalentes a la conjunción de sus literales. Puesto que trabajaremos con formas $\delta$-clausales sin variables (tras la reducción a modelos finitos que haremos con las $C$-tablas), no empleamos en las definiciones nociones como renombramiento de variables, unificación, etc.

**Definición 9.** *Dadas dos $\delta$-cláusulas $\Sigma_1 \cup \{\lambda\}$ y $\Sigma_2 \cup \{\neg\lambda\}$, la* regla de $\delta$-resolución *produce su* $\delta$-resolvente $\Sigma_1 \cup \Sigma_2$*:*

$$\frac{\Sigma_1 \cup \{\lambda\} \qquad \Sigma_2 \cup \{\neg\lambda\}}{\Sigma_1 \cup \Sigma_2}$$

**Definición 10.** *Una $\delta$-cláusula $\Sigma$ es* demostrable mediante $\delta$-resolución *a partir de la forma $\delta$-clausal $A$, lo que representamos como $A \vdash_\delta \Sigma$, si existe una secuencia de $\delta$-cláusulas tal que:*

- *Cada $\delta$-cláusula de la secuencia o bien pertenece a $A$ o es un $\delta$-resolvente de $\delta$-cláusulas anteriores.*
- *La última $\delta$-cláusula de la secuencia es $\Sigma$.*

**Teorema 2.** *Dadas la forma $\delta$-clausal $A$ y la $\delta$-cláusula $\Sigma$, si $A \vdash_\delta \Sigma$ entonces $\Sigma \models A$.*

*Demostración.* Hacemos la prueba por inducción sobre el número de aplicaciones de la regla de $\delta$-resolución. En el caso base, con 0 aplicaciones, $\Sigma \in A$, y por la definición 2, $\Sigma \models A$. Consideremos que hasta la $n$-ésima aplicación, $A$ es consecuencia lógica de cada $\delta$-cláusula resultante. Sea $\Sigma = \Sigma_1 \cup \Sigma_2$ la $n+1$-ésima $\delta$-cláusula obtenida, un $\delta$-resolvente de $\Sigma_1 \cup \{\lambda\}$ y $\Sigma_2 \cup \{\neg\lambda\}$. Entonces, cada estructura $\mathcal{M}$ que satisfaga $\Sigma$ satisface una de las $\delta$-cláusulas anteriores $\Sigma_1 \cup \{\lambda\}$ o $\Sigma_2 \cup \{\neg\lambda\}$, puesto que $\mathcal{M}$ satisface cada literal de $\Sigma_1 \cup \Sigma_2$ y $\mathcal{M} \models \lambda$ o $\mathcal{M} \models \neg\lambda$. Finalmente, por la hipótesis de inducción, $\mathcal{M}$ satisface $A$. Por tanto, $\Sigma \models A$.

**Teorema 3.** *Sea $A$ una forma $\delta$-clausal cuyas constantes pertenecen a $C$. Entonces, si $A$ es $C$-válida, $A \vdash_\delta \Diamond$.*

*Demostración.* Sea $A$ una forma $\delta$-clausal $C$-válida, $s$ la cardinalidad de $A$ y $t$ la suma de las cardinalidades de todas las $\delta$-cláusulas de $A$. Tomemos $k = t - s$. Procedemos por inducción sobre el valor de $k$, considerando que $\Diamond \notin A$ (para evitar el caso trivial, donde es obvio que $A \vdash_\delta \Diamond$). Si $k = 0$, la única posibilidad es que $A$ sea un conjunto de $\delta$-cláusulas unitarias (de un solo literal). Entonces, como la interpretación que toda $C$-estructura hace de cada constante de $A$ es

diferente (por ser todas constantes de $C$), si $A$ es $C$-válida sólo puede ocurrir que contenga dos $\delta$-cláusulas $\{\lambda\}$ y $\{\neg\lambda\}$. Con sólo una aplicación de la regla de $\delta$-resolución, obtenemos $\Diamond$.

Supongamos que el teorema se verifica para $k \leq n$, así que lo probaremos para $k = n + 1$. En este caso, debe haber en $A$ una $\delta$-cláusula $\Sigma = \{\lambda_1, \lambda_2, \ldots, \lambda_m\}$, donde $m \geq 2$. Entonces, definimos $\Sigma' = \{\lambda_1\}$, y $\Sigma'' = \{\lambda_2, \ldots, \lambda_m\}$. Pero si $A$ es $C$-válida, también lo son $(A - \Sigma) \cup \{\Sigma'\}$ y $(A - \Sigma) \cup \{\Sigma'\}$ (esto es una conclusión directa a partir de las definiciones 1 y 2), y para tales conjuntos, $k \leq n$, por lo que podemos obtener $\Diamond$ desde ellos, mediante las pruebas (secuencias de $\delta$-cláusulas) $\mathcal{D}em'$ y $\mathcal{D}em''$, respectivamente. Sea $\mathcal{D}em$ una prueba a partir de $A$, construida de forma parecida a $\mathcal{D}em''$, pero cada vez que $\Sigma''$ se usa en $\mathcal{D}em''$, se usa $\Sigma$ en $\mathcal{D}em$. Entonces, como la única diferencia entre $\Sigma''$ y $\Sigma$ es que $\Sigma$ contiene $\lambda_1$, la última $\delta$-cláusula en $\mathcal{D}em$ es o bien $\Diamond$ (en este caso $A \vdash_\delta \Diamond$, y la prueba termina) o $\{\lambda_1\}$. En el último caso, ya han aparecido en la demostración todas las $\delta$-cláusulas de $(A - \Sigma) \cup \{\Sigma'\}$ (puesto que $\Sigma'$ es $\{\lambda_1\}$), y podemos completarla como $\mathcal{D}em'$ para obtener $\Diamond$.

**Teorema 4.** *Sea $A$ una forma $\delta$-clausal $C$-equivalente a $\alpha$, que sólo contiene constantes de $C$. Entonces, $A \vdash_\delta \Sigma$ para cada $\delta$-cláusula $\Sigma$ tal que:*

1. *$\Sigma$ es $C$-satisfactible.*
2. *$\Sigma \models_C \alpha$.*
3. *No existe ninguna $\delta$-cláusula $\Sigma' \subset \Sigma$ tal que $\Sigma' \models_C \alpha$.*

*Demostración.* Sean $\alpha \in \mathscr{L}$, $A$ una forma $\delta$-clausal $C$-equivalente a $\alpha$ que sólo contiene constantes de $C$ y $\Sigma = \{\lambda_1, \ldots, \lambda_n\}$ una $\delta$-cláusula con las propiedades que indica la enumeración del enunciado del teorema 4. Probemos que $A \vdash_\delta \Sigma$. Como $\Sigma \models_C \alpha$, tenemos que $\models_C \lambda_1 \wedge \ldots \wedge \lambda_n \rightarrow \alpha$, ya que cada $C$-estructura o bien no satisface $\Sigma$ o satisface $\alpha$ (por ser $\alpha$ $C$-consecuencia lógica de $\Sigma$). Por evaluación de $\rightarrow$ y $\neg$, tenemos que $\models_C \overline{\lambda_1} \vee \ldots \vee \overline{\lambda_n} \vee \alpha$. Entonces, como $A \cup \{\{\overline{\lambda_1}\}, \ldots, \{\overline{\lambda_n}\}\}$ es una forma $\delta$-clausal $C$-equivalente a la fórmula anterior, por ser $C$-válida es posible obtener $\Diamond$ desde ella (teorema 3), mediante una prueba, que llamamos $\mathcal{D}em$. De modo similar a como hicimos al probar el teorema 3, construimos una prueba paralela a $\mathcal{D}em$, que llamamos $\mathcal{D}em'$, que sólo emplea $\delta$-cláusulas de $A$. Ahora, cuando una $\delta$-cláusula de tipo $\{\overline{\lambda_i}\}$, $1 \leq i \leq n$, se usa en $\mathcal{D}em$, no se hace nada en $\mathcal{D}em'$. Fácilmente observamos que la última $\delta$-cláusula de $\mathcal{D}em'$ debe ser una $\Sigma' \subseteq \Sigma$, pues cada $\delta$-cláusula $\{\overline{\lambda_i}\}$ sólo puede eliminar el literal $\lambda_i$ en la prueba. Pero si $\Sigma' \subset \Sigma$, como por el teorema 2, $\Sigma' \models A$, entonces $\Sigma' \models_C A$, y por tanto $\Sigma' \models_C \alpha$, lo cual es contrario a lo que sabemos de $\Sigma$. Por tanto, $\Sigma' = \Sigma$, de modo que $A \vdash_\delta \Sigma$.

**Definición 11.** *Decimos que la $\delta$-cláusula $\Sigma'$* subsume *la $\delta$-cláusula $\Sigma$ syss $\Sigma' \subset \Sigma$.*

**Corolario 2.** *Para todas las $\delta$-cláusulas $\Gamma$, $\Sigma$ y $\Lambda$, y cada forma $\delta$-clausal $A$, si $A \cup \{\Sigma\} \cup \{\Sigma \cup \Lambda\} \vdash_\delta \Gamma$ y $\Gamma$ es $C$-satisfactible, para cierto conjunto de constantes $C$, entonces existe una $\delta$-cláusula $\Gamma'$ tal que $A \cup \{\Sigma\} \vdash_\delta \Gamma'$ y $\Gamma' \subseteq \Gamma$.*

*Demostración.* Si eliminamos $\Sigma \cup \Lambda$ de $A \cup \{\Sigma\} \cup \{\Sigma \cup \Lambda\}$, la forma $\delta$-clausal resultante, $A \cup \{\Sigma\}$, es $C$-equivalente a la anterior, como puede observarse fácilmente mediante un simple razonamiento semántico basado en las definiciones 1 y 2. Por el teorema 2, $\Gamma \models A \cup \{\Sigma\} \cup \{\Sigma \cup \Lambda\}$, con lo que $\Gamma \models_C A \cup \{\Sigma\} \cup \{\Sigma \cup \Lambda\}$, y por la $C$-equivalencia mencionada, $\Gamma \models_C A \cup \{\Sigma\}$. Pero sea $\alpha \in \mathscr{L}$ una fórmula $C$-equivalente a $A \cup \{\Sigma\}$ (basta con que $\alpha$ sea la disyunción de las conjunciones elementales formadas a partir de cada $\delta$-cláusula de $A \cup \{\Sigma\}$). Entonces, $\Gamma \models_C \alpha$, por lo que debe existir una $\delta$-cláusula satisfactible $\Gamma' \subseteq \Gamma$ tal que $\Gamma' \models_C \alpha$ y para cualquier $\Gamma'' \subset \Gamma'$, $\Gamma'' \not\models_C \alpha$. Entonces, por el teorema 4, $A \cup \{\Sigma\} \vdash_\delta \Gamma'$.

**Corolario 3.** *Para toda $\delta$-cláusula $C$-satisfactible $\Sigma$, y cada forma $\delta$-clausal $A$ y literales $\lambda$, $\neg\lambda$, $\gamma_1$, ...,$\gamma_n$, si $A \cup \{\{\lambda, \neg\lambda, \gamma_1, \ldots, \gamma_n\}\} \vdash_\delta \Sigma$, entonces hay una $\delta$-cláusula $\Sigma' \subseteq \Sigma$ tal que $A \vdash_\delta \Sigma'$.*

*Demostración.* Por el teorema 2, $\Sigma \models A \cup \{\{\lambda, \neg\lambda, \gamma_1, \ldots, \gamma_n\}\}$, y por tanto $\Sigma \models_C A \cup \{\{\lambda, \neg\lambda, \gamma_1, \ldots, \gamma_n\}\}$, pero como $\{\lambda, \neg\lambda, \gamma_1, \ldots, \gamma_n\}$ no es $C$-satisfactible, por las definiciones 1 y 2, $A \cup \{\{\lambda, \neg\lambda, \gamma_1, \ldots, \gamma_n\}\}$ es $C$-equivalente a $A$, con lo que $\Sigma \models_C A$. Sea $\alpha$ una fórmula $C$-equivalente a $A$. Entonces, $\Sigma \models_C \alpha$. Sea $\Sigma' \subseteq \Sigma$ una $\delta$-cláusula tal que $\Sigma' \models_C \alpha$ pero para toda $\Sigma'' \subset \Sigma'$, $\Sigma'' \not\models_C \alpha$. Entonces, por el teorema 4, $A \vdash_\delta \Sigma'$.

**Definición 12.** *Dada la forma $\delta$-clausal $A$, el conjunto* saturación por $\delta$-resolución de $A$, *que llamamos $A^\delta$, es el más pequeño que contiene cada $\delta$-cláusula $\Sigma$ tal que:*

- *$A \vdash_\delta \Sigma$.*
- *$\Sigma$ es satisfactible.*
- *No existe ninguna $\Sigma' \subset \Sigma$ tal que $A \vdash_\delta \Sigma'$.*

Dado un conjunto finito de $\delta$-cláusulas $A$, $A^\delta$ puede obtenerse en un número finito de pasos. A pesar de ello, dada la complejidad del proceso, cobra sentido el empleo de estrategias que hagan la búsqueda más eficiente, como es habitual en resolución estándar. Como el cálculo de $\delta$-resolución es sintácticamente equivalente al de resolución, es posible adaptar las mismas estrategias de búsqueda. Por ejemplo, los corolarios 2 y 3 nos permiten eliminar las $\delta$-cláusulas subsumidas o contradictorias en el momento en que aparecen, sin esperar al final del proceso.

**Corolario 4.** *Sea $A$ una forma $\delta$-clausal $C$-equivalente a $\alpha$, para cierto conjunto $C$ de constantes que contiene todos los términos de $A$ y de $\alpha$. Entonces, $A^\delta$ es el conjunto de $\delta$-cláusulas $\Sigma$ tales que*

- *$\Sigma \models_C \alpha$.*
- *$\Sigma$ es $C$-satisfactible.*
- *Para toda $\Sigma' \subset \Sigma$, $\Sigma' \not\models_C \alpha$.*

*Demostración.* El teorema 4 asegura que cada $\Sigma$ que tenga las propiedades del enunciado de este corolario es demostrable mediante $\delta$-resolución a partir de $A$. Por tanto, por la definición 12, $\Sigma$ debe pertenecer a $A^\delta$, puesto que si $\Sigma$

es $C$-satisfactible también es satisfactible y no hay ninguna $\Sigma' \subset \Sigma$ tal que $A \vdash_\delta \Sigma'$ (pues entonces, por el teorema 2, $\Sigma' \models A$, y también $\Sigma' \models_C A$, lo que contradice las propiedades de $\Sigma$). Por tanto, cada $\delta$-cláusula con las propiedades de $\Sigma$ pertenece a $A^\delta$. Probemos ahora que toda $\delta$-cláusula $\Sigma$ de $A^\delta$ tiene las propiedades que enuncia este corolario. En primer lugar, como $A \vdash_\delta \Sigma$, entonces $\Sigma \models A$ (teorema 2) y por tanto $\Sigma \models_C A$ y $\Sigma \models_C \alpha$. Además, por la definición 12 tenemos que $\Sigma$ es satisfactible, lo que implica que no tiene literales complementarios. Como además todos los términos de $\Sigma$ son constantes de $C$, por el lema 1 sabemos que $\Sigma$ es $C$-satisfactible. Por último, si fuera el caso de que existiera una $\delta$-cláusula $\Sigma' \subset \Sigma$ tal que $\Sigma' \models_C \alpha$, entonces habría una $\delta$-cláusula $\Sigma'' \subset \Sigma$ tal que $\Sigma'' \models_C \alpha$ y para toda $\Sigma^* \subset \Sigma''$, $\Sigma^* \not\models_C \alpha$. Pero entonces, por el teorema 4, $A \vdash_\delta \Sigma''$, lo que contradice lo que la definición 12 afirma sobre $\Sigma$. Por tanto, $\Sigma$ tiene todas las propiedades que indica el teorema.

**Teorema 5.** *Dado el problema abductivo $\langle \{\theta_1, \ldots, \theta_n\}, \varphi \rangle$, si $N_\Theta$ y $O$ son respectivamente formas $\delta$-clausales $C$-equivalentes a $\neg(\theta_1 \wedge \ldots \wedge \theta_n)$ y $\varphi$, y $C$ contiene todos los términos de $N_\Theta$, $O$, $\varphi$ y de cada $\theta_i$, $1 \leq i \leq n$, entonces:*

$$\mathcal{A}bd\,(\Theta, \varphi)_C = \left(N_\Theta^\delta \cup O^\delta\right)^\delta - \left(N_\Theta^\delta \cup O^\delta\right)$$

*Demostración.* Tomando $\Theta = \{\theta_1, \ldots, \theta_n\}$, $\Sigma \in \mathcal{A}bd(\Theta, \varphi)_C$ equivale, según la definición 6, a que se verifica:

1. $\Theta, \Sigma \models_C \varphi$. Pero esto equivale a $\Sigma \models_C \neg(\theta_1 \wedge \ldots \wedge \theta_n) \vee \varphi$ y a $\Sigma \models_C N_\Theta \cup O$, por ser $N_\Theta$ y $O$, respectivamente, $C$-equivalentes a $\neg(\theta_1 \wedge \ldots \wedge \theta_n)$ y $\varphi$.
2. $\Theta \cup \Sigma$ es $C$-satisfactible, que equivale a $\Sigma \not\models_C \neg(\theta_1 \wedge \ldots \wedge \theta_n)$. Desde aquí también se infiere que $\Sigma$ es $C$-satisfactible.
3. $\Sigma \not\models_C \varphi$.
4. No existe ninguna $\Sigma' \subset \Sigma$ tal que $\Theta, \Sigma' \models_C \varphi$. Esto equivale a que para toda $\Sigma' \subset \Sigma$, $\Sigma' \not\models_C \neg(\theta_1 \wedge \ldots \wedge \theta_n) \vee \varphi$.

En primer lugar, consideremos que $\Sigma \in \mathcal{A}bd(\Theta, \varphi)_C$. Entonces:

- Desde 1, 2 y 4 (enumeración anterior), teniendo en cuenta que $N_\Theta \cup O$ es $C$-equivalente a $\neg(\theta_1 \wedge \ldots \wedge \theta_n) \vee \varphi$, obtenemos, por el corolario 4, $\Sigma \in (N_\Theta \cup O)^\delta$, y por tanto $\Sigma \in (N_\Theta^\delta \cup O^\delta)^\delta$ (el orden de aplicación de la regla de $\delta$-resolución en un conjunto $A$ no altera el resultado $A^\delta$).
- Desde 2 y 3 llegamos, por el corolario 4, a $\Sigma \notin N_\Theta^\delta$ y $\Sigma \notin O^\delta$, respectivamente, con lo que $\Sigma \notin (N_\Theta^\delta \cup O^\delta)$.

De los resultados obtenidos concluimos $\Sigma \in (N_\Theta^\delta \cup O^\delta)^\delta - (N_\Theta^\delta \cup O^\delta)$. Ahora, supongamos esto último, para probar $\Sigma \in \mathcal{A}bd(\Theta, \varphi)_C$ Tenemos:

- $\Sigma \in (N_\Theta^\delta \cup O^\delta)^\delta$. Como se ha observado, esto es equivalente a $\Sigma \in (N_\Theta \cup O)^\delta$, y por ser $C$-equivalentes $N_\Theta \cup O$ y $\neg(\theta_1 \wedge \ldots \wedge \theta_n) \vee \varphi$, por el corolario 4, $\Sigma \models_C \neg(\theta_1 \wedge \ldots \wedge \theta_n) \vee \varphi$ y para toda $\Sigma' \subset \Sigma$, $\Sigma' \not\models_C \neg(\theta_1 \wedge \ldots \wedge \theta_n) \vee \varphi$.

- $\Sigma \notin N^{\delta}_{\Theta}$. Supongamos que $\Sigma \models_C \neg(\theta_1 \wedge \ldots \wedge \theta_n)$. Entonces existirá una $\Sigma' \subseteq \Sigma$ tal que $\Sigma' \models_C \neg(\theta_1 \wedge \ldots \wedge \theta_n)$ y para toda $\Sigma^* \subset \Sigma'$, $\Sigma^* \not\models_C \neg(\theta_1 \wedge \ldots \wedge \theta_n)$. Pero entonces, puesto que $\Sigma$ es $C$-satisfactible (lema 1), por el corolario 4 tendríamos que $\Sigma' \in N^{\delta}_{\Theta}$, con lo que no puede ser que $\Sigma' = \Sigma$, pero tampoco que $\Sigma' \subset \Sigma$, ya que esto contradice $\Sigma \in (N^{\delta}_{\Theta} \cup O^{\delta})^{\delta}$, puesto que $\Sigma$ queda subsumida por $\Sigma'$. Por tanto, $\Sigma \not\models_C \neg(\theta_1 \wedge \ldots \wedge \theta_n)$.
- $\Sigma \notin O^{\delta}$. En este caso, mediante un razonamiento por reducción al absurdo, paralelo al del apartado anterior, concluimos que $\Sigma \not\models_C \varphi$.

Los resultados obtenidos son los mismos que al comienzo de la demostración comentamos que equivalen a $\Sigma \in \mathcal{A}bd(\Theta, \varphi)$.

**Corolario 5.** *Sean $\alpha, \beta \in \mathscr{L}$, $C$-equivalentes, respectivamente, a las formas $\delta$-clausales $A$ y $B$, de modo que cada término de $\alpha$, $\beta$, $A$ y $B$ pertenece a $C$. Entonces $\alpha \models_C \beta$ syss para cada $\Sigma \in A^{\delta}$ existe una $\Sigma' \in B^{\delta}$ tal que $\Sigma' \subseteq \Sigma$.*

*Demostración.* Supongamos que $\alpha \models_C \beta$ y $\Sigma \in A^{\delta}$. Entonces, por el corolario 4, $\Sigma$ es $C$-satisfactible y $\Sigma \models_C \alpha$. Pero $\alpha \models_C \beta$, y por tanto $\Sigma \models_C \beta$. Por ello, existe una $\delta$-cláusula $\Sigma' \subseteq \Sigma$ tal que $\Sigma' \models_C \beta$, $\Sigma'$ es satisfactible y para toda $\Sigma^* \subset \Sigma'$, $\Sigma^* \not\models_C \beta$. Entonces, por el corolario 4, $\Sigma' \in B^{\delta}$.

Ahora, supongamos que para cada $\Sigma \in A^{\delta}$ existe una $\Sigma' \in B^{\delta}$ tal que $\Sigma' \subseteq \Sigma$. Probaremos que $\alpha \models_C \beta$. Sea $\mathcal{M}$ una $C$-estructura tal que $\mathcal{M} \models \alpha$. Entonces, por el lema 3, $\mathcal{T}(\{\alpha\})_C$ tiene al menos una rama abierta cuyos literales son satisfechos por $\mathcal{M}$. Sea $\Sigma^{\star}$ la $\delta$-cláusula compuesta por tales literales. Por el lema 2, $\Sigma^{\star} \models_C \alpha$. Entonces existe una $\delta$-cláusula $\Sigma \subseteq \Sigma^{\star}$ tal que $\Sigma \models_C \alpha$ y para cada $\Sigma'' \subset \Sigma$, $\Sigma'' \not\models_C \alpha$. Por tanto, por el corolario 4, $\Sigma \in A^{\delta}$. Entonces, existe una $\Sigma' \subseteq \Sigma$ tal que $\Sigma' \in B^{\delta}$. Por el corolario 4, $\Sigma' \models_C \beta$. Entonces, como $\mathcal{M} \models \Sigma'$ (ya que $\Sigma' \subseteq \Sigma \subseteq \Sigma^*$ y $\mathcal{M} \models \Sigma^*$), $\mathcal{M} \models \beta$. Por tanto, $\alpha \models_C \beta$.

## 4. Proceso abductivo mediante $C$-tablas y $\delta$-resolución

**Definición 13.** *Definimos el* proceso de búsqueda de soluciones $C$-abductivas *de $\langle \Theta, \varphi \rangle$, con $\Theta = \{\theta_1, \ldots, \theta_n\}$, donde $\Theta \subset \mathscr{L}$, $\varphi \in \mathscr{L}$, y todos los términos de $\Theta$ y $\varphi$ están contenidos en el conjunto de constantes $C$, de la siguiente forma:*

**Paso 1: Análisis de la teoría.** *Sea $N_{\Theta} = \mathcal{C}\delta(\mathcal{T}(\{\neg(\theta_1 \wedge \ldots \wedge \theta_n)\})_C)$:*
- *Si $N_{\Theta} = \emptyset$, entonces $\Theta$ es $C$-válida, y el proceso acaba.*
- *Si no, se obtiene $N^{\delta}_{\Theta}$. Si $\Diamond \in N^{\delta}_{\Theta}$, entonces $\Theta$ no es $C$-satisfactible, y el proceso acaba. En otro caso,*

**Paso 2: Análisis de la observación.** *Sea $O = \mathcal{C}\delta(\mathcal{T}(\{\varphi\})_C)$. Entonces:*
- *Si $O = \emptyset$, entonces $\varphi$ no es $C$-satisfactible, y el proceso acaba.*
- *Si no, se obtiene $O^{\delta}$. Si $\Diamond \in O^{\delta}$, entonces $\varphi$ es $C$-válida, y el proceso acaba. En otro caso,*

**Paso 3: Búsqueda de refutaciones.** *Si para cada $\delta$-cláusula $\Sigma \in O^{\delta}$ existe una $\Sigma' \in N^{\delta}_{\Theta}$ tal que $\Sigma' \subseteq \Sigma$, entonces $\Theta \models_C \neg\varphi$, y el proceso termina. En otro caso,*

**Paso 4: Búsqueda de explicaciones.** *Desde $N^{\delta}_{\Theta}$ y $O^{\delta}$, se obtienen $(N^{\delta}_{\Theta} \cup O^{\delta})$ y luego $(N^{\delta}_{\Theta} \cup O^{\delta})^{\delta}$. Entonces, si $\Diamond \in (N^{\delta}_{\Theta} \cup O^{\delta})^{\delta}$, $\Theta \models_C \varphi$ y el proceso acaba. En otro caso, $\langle \Theta, \varphi \rangle$ es un problema abductivo, y el proceso devuelve:*

$$\mathcal{A}bd\,(\Theta, \varphi)_C = \left(N^{\delta}_{\Theta} \cup O^{\delta}\right)^{\delta} - \left(N^{\delta}_{\Theta} \cup O^{\delta}\right)$$

**Corolario 6.** *Para cualesquiera $\Theta \subset \mathscr{L}$ y $\varphi \in \mathscr{L}$, el proceso de búsqueda de soluciones $C$-abductivas (para cualquier conjunto $C$ de constantes que contenga todos los términos de $\Theta$ y $\varphi$) que sigue la definición 13 para $\langle \Theta, \varphi \rangle$, es correcto.*

*Demostración.* Dada una fórmula $\alpha \in \mathscr{L}$ que sea $C$-equivalente a la forma $\delta$-clausal $A$, $\Diamond \in A^{\delta}$ syss $\models_C \alpha$ (por los teoremas 2 y 3). Igualmente, $\alpha$ no es $C$-satisfactible syss $\mathcal{C}\delta(\mathcal{T}(\{\alpha\})_C)$ tampoco lo es (teorema 1), es decir, syss $\mathcal{C}\delta(\mathcal{T}(\{\alpha\})_C) = \emptyset$ (puesto que, por el corolario 1 y el lema 2, $\mathcal{T}(\{\alpha\})_C$ no tendrá ramas abiertas). Con estas observaciones, la prueba de corrección de los pasos 1 y 2 de la definición 13 es directa. El paso 3 es una consecuencia del corolario 5. Para el paso 4, si $\Diamond \in (N^{\delta}_{\Theta} \cup O^{\delta})^{\delta}$, entonces $N^{\delta}_{\Theta} \cup O^{\delta}$ es $C$-válida (teorema 2). Pero esta forma $\delta$-clausal es $C$-equivalente a $\neg(\theta_1 \wedge \ldots \wedge \theta_n) \vee \varphi$, por lo que $\models_C \neg(\theta_1 \wedge \ldots \wedge \theta_n) \vee \varphi$, lo cual equivale a $\Theta \models_C \varphi$. En otro caso, la forma en que el paso 4 construye $\mathcal{A}bd(\Theta, \varphi)_C$ es correcta, por el teorema 5.

## 5. Comentarios

Aunque el problema de la abducción en primer orden es, por lo general, indecidible, cobra sentido la búsqueda de soluciones abductivas en estructuras de cardinalidad finita, como son las $C$-estructuras, dado que buena parte de las aplicaciones que la abducción encuentra en Inteligencia Artificial se dan en dominios cerrados, con elementos conocidos.

El algoritmo presentado en la definición 13 devuelve, para cierto problema abductivo $\langle \Theta, \varphi \rangle$, todas las soluciones $C$-abductivas que verifican los requisitos de la definición 6. Son unos requisitos fuertes, pero resulta posible suavizarlos sin salir del marco introducido en este trabajo.

Disponemos de una implementación en Prolog de los algoritmos presentados que muestran mayor eficiencia en la búsqueda de soluciones abductivas que el uso por separado de los métodos tanto de tablas semánticas como de resolución.

## Referencias

1. Aliseda, A.: Abductive Reasoning: Logical Investigations into Discovery and Explanation. Volumen 330 de Synthese Library. Springer (2005)
2. Beth, E.W.: Semantic entailment and formal derivability. Koninklijke Nederlandse Akademie van Wentenschappen, Proceedings of the Section of Sciences **18** (1955) 309–342
3. Boolos, G.: Trees and finite satsifactibility. Notre Dame Journal of Formal Logic **25** (1984) 110–115

4. Díaz Estévez, E.: Árboles semánticos y modelos mínimos. En: Actas del I Congreso de la Sociedad de Lógica, Metodología y Filosofía de la Ciencia en España, Universidad Complutense de Madrid (1993) 40
5. Nepomuceno, A.: Scientific explanation and modified semantic tableaux. En Magnani, L., Nerssessian, N., Pizzi, C., eds.: Logical and Computational Aspects of Model-Based Reasoning. Applied Logic Series. Kluwer Academic Publishers (2002) 181–198
6. Reyes-Cabello, A.L., Aliseda-Llera, A., Nepomuceno-Fernández, A.: Abductive reasoning in first order logic. Logic Journal of the IGPL **14** (2) (2006)
7. Cialdea Mayer, M., Pirri, F.: First order abduction via tableau and sequent calculi. Bulletin of the IGPL **1** (1993) 99–117
8. Aliseda, A.: Seeking Explanations: Abduction in Logic, Philosophy of Science and Artificial Intelligence. Dissertation Series. Institute for Logic, Language, and Computation, Holland (1997)
9. Robinson, J.: A machine-oriented logic based on the resolution principle. Journal of the ACM **12** (1965) 23–41
10. Soler-Toscano, F., Nepomuceno-Fernández, A., Aliseda-Llera, A.: Model-based abduction via dual resolution. Logic Journal of the IGPL **14** (2) (2006)

# Evaluación parcial de programas lógicos multi-adjuntos y aplicaciones

Pascual Julian[1], Ginés Moreno[2] y Jaime Penabad[3]

[1] Dep. de Tecnologías y Sistemas de Información, UCLM, 13071 Ciudad Real.
[2] Dep. de Sistemas Informáticos, UCLM, 02071 Albacete.
[3] Dep. de Matemáticas, UCLM, 02071 Albacete.
{Pascual.Julian},{Gines.Moreno},{Jaime.Penabad}@uclm.es

**Resumen** Como su propio nombre indica, la *Evaluación Parcial* (EP) de un objetivo con respecto a un programa, consiste en evaluar parcialmente dicho objetivo mediante la generación de un árbol de desplegado (normalmente incompleto) para el mismo. La EP ofrece una amplia gama de aplicaciones como, por ejemplo, la transformación, optimización, depuración, corrección, aprendizaje y, muy especialmente, la *especialización* automática de programas declarativos con respecto a parte de sus datos de entrada mediante el cálculo de *resultantes*. En este artículo no sólo introducimos, por primera vez, los conceptos de EP y especialización de programas lógicos borrosos sino que, además, mostramos una aplicación original de estas técnicas que permite el cálculo eficiente de *reductantes*. Los reductantes son una interesante herramienta teórica introducida inicialmente en el contexto de la programación lógica generalizada con anotaciones para demostrar propiedades de corrección y completitud. Este concepto ha sido recientemente adaptado al entorno más rico y flexible de la programación lógica multi-adjunta, intentando resolver el problema de incompletitud que surge cuando se trabaja en algunos retículos. Aunque, en general, la noción de reductante no siempre es computable, en este trabajo desarrollamos un cálculo eficiente de reductantes basado en técnicas de EP.
*Palabras clave:* Prog. Lógica Borrosa, Evaluación Parcial, Reductante.

## 1. Introducción

Por su alto nivel de abstracción y su gran expresividad, los lenguajes declarativos son ampliamente reconocidos como una de las mejores herramientas para el desarrollo de aplicaciones con una fuerte componente simbólica, como ocurre a menudo en los campos de la inteligencia artificial, la ingeniería del conocimiento, etc. Más aún si se trata de lenguajes que, como el que nos ocupa en este trabajo, están basados en lógica difusa, lo que les aporta una habilidad natural para modelar escenarios de naturaleza vaga o imprecisa y razonamiento aproximado.

Por otro lado, entre la gran variedad de técnicas de transformación que se han propuesto en la bibliografía para mejorar el código de los programas, una

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 712-724 2006.

de las más estudiadas y mejor conocidas es la *Evaluación Parcial* (EP), que además ofrece un entorno unificado para el estudio de compiladores e intérpretes [13]. Informalmente, la *Evaluación Parcial* es una técnica de transformación automática de programas que, entre sus principales objetivos, persigue la optimización/especialización de un programa con el fin de dotarlo de un mejor comportamiento computacional. Más concretamente, se espera que el programa especializado (también llamado programa *residual* o *parcialmente evaluado*) pueda ejecutarse más eficientemente que el programa original. Esto es posible gracias a que el programa residual evita realizar, en tiempo de ejecución, algunos cálculos que ya fueron efectuadas sólo una vez en tiempo de EP. Para alcanzar este objetivo, la EP usa computación simbólica así como algunas técnicas que proporciona el campo de la transformación de programas [4], especialmente la llamada *transformación de desplegado*. Desplegar consiste esencialmente en reemplazar una llamada por su definición, usando substituciones apropiadas. En general, las técnicas de EP incluyen criterios de parada para garantizar la terminación del proceso[4]; por consiguiente, se trata de un tipo de trasformación automática (esto es, el proceso de EP puede ser completado sin intervención humana), lo que constituye un importante rasgo diferenciador de estas técnicas con respecto a otros métodos de transformación de programas [4, 23].

La EP ha sido ampliamente utilizada en diferentes paradigmas de programación declarativa, como es el caso de la programación funcional [5, 13, 24], la programación lógica [10, 16, 19, 23], donde se denomina habitualmente *deducción parcial* y la programación lógico–funcional [3, 1, 2]. También ha sido aplicada al área de lenguajes imperativos (por ejemplo, el lenguaje C [6]). Aunque los objetivos son similares, las técnicas son a menudo diferentes debido a los distintos modelos computacionales subyacentes. Las técnicas convencionales de deducción parcial de programas lógicos se apoyan frecuentemente en la propagación de parámetros vía unificación [12], lo que forma parte del principio mismo de resolución. En este contexto, los datos de entrada son proporcionados al sistema como argumentos parciales de un objetivo (en la mayoría de los casos, una fórmula atómica).

También, la EP se ha aplicado extensivamente a una gran variedad de problemas concretos entre los que pueden citarse [13]: la especialización de consultas en bases de datos; la demostración automática de teoremas; la optimización del procedimiento de trazado de rayos en el área de la generación de gráficos por computador; el mantenimiento del software y la comprensión de programas; el entrenamiento de una red neuronal; y la especialización de simuladores de circuitos.

Por su parte, la *Programación Lógica Multi-Adjunta* [20, 21, 22] es un entorno sumamente flexible que combina la lógica borrosa y la programación lógica. De manera informal, un programa lógico multi-adjunto puede verse como un conjunto de reglas con un grado de verdad asociado, y un objetivo es una pregunta al sistema más una substitución (inicialmente la substitución identidad, denotada

[4] El control del proceso de EP es una tarea importante. Sin embargo, está fuera del alcance de este trabajo preliminar.

por *id*). Dado un programa lógico multi-adjunto, los objetivos son evaluados en dos fases computacionales separadas. Durante la fase *operacional*, se aplican de manera sistemática *pasos admisibles* (una generalización de la regla de inferencia clásica *modus ponens*) mediante un proceso de razonamiento hacia a atrás, de manera análoga a la ejecución de pasos de resolución en la programación lógica pura. De modo más preciso, un paso admisible, para un átomo $A$ seleccionado en un objetivo y una regla $\langle H \leftarrow_i \mathcal{B}; v\rangle$ de un programa, tal que existe un unificador más general $\theta$ entre $A$ y $H$, se obtiene al substituir en el objetivo de partida el átomo $A$ por la expresión $(v \&_i \mathcal{B})\theta$, donde "$\&_i$" es una conjunción adjunta ligada a la evaluación de la regla de *modus ponens*. Finalmente, la fase operacional devuelve una substitución computada junto con una expresión donde todos los átomos han sido explotados. Esta última expresión es interpretada posteriormente en un retículo concreto en la fase que nosotros llamamos *interpretativa*, devolviendo un par $\langle$*grado de verdad*; *substitución*$\rangle$ que es el concepto borroso análogo a la noción clásica de respuesta computada usada tradicionalmente en la programación lógica.

Un programa lógico multi-adjunto, interpretado en un retículo completo, precisa contener un tipo especial de reglas llamadas reductantes a fin de garantizar su completitud (aproximada). Esto introduce severos inconvenientes prácticos a la hora de implementar un sistema eficiente y completo de programación lógica multi-adjunta, ya que pueden dispararse tanto el tamaño de los programas, como los tiempos de ejecución. En general, la noción de reductante no es siempre computable y, como hemos considerado, puede introducir una importante pérdida de eficiencia en las implementaciones prácticas. Por tanto, si deseamos desarrollar un sistema completo y eficiente para este entorno de programación, resulta crucial definir técnicas de optimización para el cálculo de reductantes.

Inspirados por nuestra experiencia previa en el desarrollo de técnicas de evaluación parcial para programas declarativos, así como en nuestros últimos avances en programación lógica borrosa, en este trabajo nos proponemos abordar las siguientes tareas:

- Definir el concepto de EP para programas y objetivos lógicos multi-adjuntos, adaptando las técnicas que surgieron en el campo de la dedución parcial de programas lógicos puros [10, 16, 19, 23], pero haciendo uso ahora de la regla de desplegado que desarrollamos en [14, 15] para esta clase de programas lógicos borrosos.
- Como aplicación immediata del nuevo marco de EP, ilustramos sus beneficios mediante algunos ejemplos sencillos de especialización, donde se obtienen programas que se ejecutan más eficientemente que los originales.
- A continuación procedemos a relacionar la noción de EP de programas lógicos multi-adjuntos y el cálculo de *reductantes*. Para ello proporcionamos una aproximación computable de este último concepto usando técnicas de EP, lo que convierte a la noción teórica de reductante en una herramienta útil que admite una implementación eficiente en la práctica.

## 2. Programación lógica multi-adjunta

Esta sección da un breve resumen de los principales rasgos de la programación lógica multi-adjunta (remitimos al lector interesado a [20, 21, 22] donde podrá encontrar una formulación más completa).

### 2.1. El lenguaje

Trabajamos con un lenguaje de primer orden, $\mathcal{L}$, conteniendo variables, símbolos de función, símbolos de predicado, constantes, cuantificadores $\forall$ y $\exists$, y conectivas varias (arbitrarias) que mejoran la expresividad del mismo:

| | | | |
|---|---|---|---|
| $\&_1, \&_2, \ldots, \&_k$ | `(conjunciones)` | $\vee_1, \vee_2, \ldots, \vee_l$ | `(disyunciones)` |
| $\leftarrow_1, \leftarrow_2, \ldots, \leftarrow_m$ | `(implicaciones)` | $@_1, @_2, \ldots, @_n$ | `(agregadores)` |

Aunque las conectivas $\&_i$, $\vee_i$ y $@_i$ son operadores binarios, frecuentemente las generalizamos y las tomamos como funciones con un número arbitrario (finito) de argumentos. En lo que sigue, escribiremos a menudo $@_i(x_1, \ldots, x_n)$ en vez de $@_i(x_1, @_i(x_2, \ldots, @_i(x_{n-1}, x_n)\ldots))$. Además, la función de verdad de cada conjunción verifica $\dot{\&}_i(\top, v) = \dot{\&}_i(v, \top) = v$, para todo $v \in L$, $i = 1, \ldots, n$; mientras que para cualquier operador de agregación $\dot{@} : L^2 \to L$ se requiere la monotononía y que satisfaga que $\dot{@}(\top, \top) = \top$ y $\dot{@}(\bot, \bot) = \bot$.

Además, nuestro lenguaje $\mathcal{L}$ contiene valores de un retículo multi-adjunto, $\langle L, \preceq, \leftarrow_1, \&_1, \ldots, \leftarrow_n, \&_n\rangle$, equipado con una colección de pares adjuntos $\langle \leftarrow_i, \&_i\rangle$, donde cada $\&_i$ es un conjuntor[5] ligado a la evaluación del *modus ponens*. En general, el conjunto de grados de verdad $L$ puede ser un retículo acotado completo aunque, a efectos de una mejor legibilidad, en los ejemplos tomaremos $L$ como el intervalo cerrado $[0, 1]$ (que es un retículo totalmente ordenado o cadena).

Una *regla* es una fórmula $H \leftarrow_i \mathcal{B}$, donde $H$ es una fórmula átomica (habitualmente llamada *cabeza*) y $\mathcal{B}$ (que se llama *cuerpo*) es una fórmula construida con fórmulas atómicas $B_1, \ldots, B_n$ ($n \geq 0$), grados de verdad de $L$ y conjunciones, disyunciones y agregadores. Las reglas cuyo cuerpo es $\top$ se llaman *hechos* (habitualmente, representaremos un hecho como una regla con cuerpo vacío). Un *objetivo* es un cuerpo planteado como una pregunta al sistema. Las variables de las reglas están universalmente cuantificadas.

A grandes rasgos, un programa lógico multi-adjunto es un conjunto de pares $\langle \mathcal{R}; \alpha\rangle$, donde $\mathcal{R}$ es una regla y $\alpha$ es el *grado de verdad* (un valor de $L$) que expresa la confianza del usuario del sistema acerca de dicha regla. Observemos que los grados de verdad están asignados axiomáticamente (por ejemplo) por un experto. Por abuso de lenguaje, en ocasiones nos referiremos al par $\langle \mathcal{R}; \alpha\rangle$ como a una "regla".

[5] Para observar una definición formal de retículo multi-adjunto y las propiedades semánticas de las conectivas de $\mathcal{L}$, véase [22]. Conviene destacar que un símbolo $\&_i$ de $\mathcal{L}$ no siempre forma parte de un par adjunto.

### 2.2. Semántica procedural

La semántica procedural del lenguaje lógico multi-adjunto $\mathcal{L}$ puede describirse contemplando dos fases: una fase operacional seguida de una fase interpretativa. De manera análoga a como consideramos en [15], en esta sección establecemos una separación clara entre ambas fases.

El mecanismo operacional usa una generalización del *modus ponens* tal que, dado un objetivo $A$ y una regla del programa $\langle H \leftarrow_i \mathcal{B}; v\rangle$, si hay una substitución $\theta = mgu(\{A = H\})$[6], sustituimos el átomo $A$ por la expresión $(v \&_i \mathcal{B})\theta$. En lo que sigue, escribiremos $\mathcal{C}[A]$ para denotar una fórmula donde $A$ es una subexpresión (frecuentemente un átomo) que aparece arbitrariamente en el contexto $\mathcal{C}[]$ (posiblemente vacío). Además, la expresión $\mathcal{C}[A/H]$ significa el reemplazamiento de $A$ por $H$ en el contexto $\mathcal{C}[]$. Usaremos también $\mathcal{V}ar(s)$ para referirnos al conjunto de variables distintas que aparecen en el objeto sintáctico $s$, mientras que $\theta[\mathcal{V}ar(s)]$ denota la substitución obtenida a partir de $\theta$ restringiendo su dominio, $Dom(\theta)$, a $\mathcal{V}ar(s)$.

**Definition 1 (Paso Admisible).** *Sea $\mathcal{Q}$ un objetivo y $\sigma$ una substitución. El par $\langle \mathcal{Q}; \sigma\rangle$ es un* estado *y denotamos por $\mathcal{E}$ el conjunto de estados. Dado un programa $\mathcal{P}$, una* computación admisible *se formaliza como un sistema de transición de estados, cuya relación de transición $\rightarrow_{AS} \subseteq (\mathcal{E} \times \mathcal{E})$ es la menor relación que satisface las siguientes* reglas admisibles[7] *(donde consideramos en todo caso que $A$ es el átomo seleccionado en $\mathcal{Q}$):*

1) $\langle \mathcal{Q}[A]; \sigma\rangle \rightarrow_{AS} \langle (\mathcal{Q}[A/v \&_i \mathcal{B}])\theta; \sigma\theta\rangle$ *si* $\theta = mgu(\{A' = A\})$, $\langle A' \leftarrow_i \mathcal{B}; v\rangle$ *en* $\mathcal{P}$

2) $\langle \mathcal{Q}[A]; \sigma\rangle \rightarrow_{AS} \langle (\mathcal{Q}[A/\bot]); \sigma\rangle$ *si no hay regla en $\mathcal{P}$ cuya cabeza unifique con $A$.*

Las fórmulas involucradas en pasos de computación admisibles son renombradas antes de usarse. Cuando sea necesario, usaremos los símbolos $\rightarrow_{AS1}$ y $\rightarrow_{AS2}$ para distinguir entre pasos admisibles específicos. Además, cuando sea preciso, indicaremos con un superíndice del símbolo $\rightarrow_{AS}$ la regla de programa usada en el paso de computación correspondiente.

**Definition 2.** *Sea $\mathcal{P}$ un programa y sea $\mathcal{Q}$ un objetivo. Una* derivación admisible *es una secuencia $\langle \mathcal{Q}; id\rangle \rightarrow^*_{AS} \langle \mathcal{Q}'; \theta\rangle$. Si $\mathcal{Q}'$ es una fórmula que no contiene átomos, el par $\langle \mathcal{Q}'; \sigma\rangle$, donde $\sigma = \theta[\mathcal{V}ar(\mathcal{Q})]$, se llama* respuesta computada admisible *(a.c.a., de admissible computed answer) para la derivación.*

[6] Denotemos por $mgu(E)$ el *unificador más general* de un conjunto de ecuaciones $E$ (véase [17] para una definición formal de este concepto).

[7] Obsérvese que el caso uno de esta definición subsume el caso dos de la definición original presentada en [22], puesto que cualquier hecho $A' \leftarrow$ puede expresarse como una regla $A' \leftarrow \top$. Sin embargo, desde el punto de vista práctico, cuando se ejecuta un paso admisible con un hecho, en lugar de "$\langle \mathcal{Q}[A]; \sigma\rangle \rightarrow_{AS} \langle (\mathcal{Q}[A/v \&_i \top])\theta; \sigma\theta\rangle$" escribiremos "$\langle \mathcal{Q}[A]; \sigma\rangle \rightarrow_{AS} \langle (\mathcal{Q}[A/v])\theta; \sigma\theta\rangle$", dado que $[\![\&_i]\!](v, \top) = v$.

Describimos ahora la fase interpretativa. Si explotamos todos los átomos del objetivo, aplicando tantos pasos admisibles como sean necesarios en la fase operacional, llegamos a una fórmula que no contiene átomos que puede interpretarse directamente en el retículo multi-adjunto $L$.

**Definition 3 (Paso Interpretativo).** *Sea $\mathcal{P}$ un programa, $\mathcal{Q}$ un objetivo y $\sigma$ una substitución. Formalizamos la noción de* computación interpretativa *como un sistema de transición de estados, cuya relación de transición $\rightarrow_{IS} \subseteq (\mathcal{E} \times \mathcal{E})$ se define como la más pequeña relación binaria que satisface: $\langle Q[@(r_1, r_2)]; \sigma\rangle \rightarrow_{IS} \langle Q[@(r_1,r_2)/\dot{@}(r_1,r_2)]; \sigma\rangle$, donde $\dot{@}$ es la función de verdad de la conectiva $@$ en el retículo $\langle L, \preceq\rangle$ asociado a $\mathcal{P}$.*

**Definition 4.** *Sea $\mathcal{P}$ un programa y $\langle Q; \sigma\rangle$ una a.c.a., es decir, $\mathcal{Q}$ es un objetivo que no contiene átomos. Una* derivación interpretativa *es una secuencia $\langle Q; \sigma\rangle \rightarrow_{IS}^{*} \langle Q'; \sigma\rangle$. Cuando $Q' = r \in L$, siendo $\langle L, \preceq\rangle$ el retículo asociado a $\mathcal{P}$, el estado $\langle r; \sigma\rangle$ se dice una* respuesta computada borrosa *(f.c.a., de fuzzy computed answer) para la derivación.*

Frecuentemente, llamaremos *derivación completa* a la secuencia de pasos admisibles/interpretativos de la forma $\langle Q; id\rangle \rightarrow_{AS}^{*} \langle \mathcal{Q}'; \sigma\rangle \rightarrow_{IS}^{*} \langle r; \sigma\rangle$ (a veces denotaremos también $\langle Q; id\rangle \rightarrow_{AS/IS}^{*} \langle r; \sigma\rangle$) donde $\langle \mathcal{Q}'; \sigma[\mathcal{V}ar(\mathcal{Q})]\rangle$ y $\langle r; \sigma[\mathcal{V}ar(\mathcal{Q})]\rangle$ son, respectivamente, la a.c.a. y la f.c.a. para la derivación.

## 3. Evaluación parcial y especialización de programas

En esta sección formalizamos las nociones básicas involucradas en la evaluación parcial de programas lógicos multi-adjuntos.

**Definition 5 (Resultante).** *Sea $\mathcal{P}$ un programa y sea $\mathcal{Q}$ un objetivo. Dada una secuencia de pasos admisibles e interpretativos $\langle \mathcal{Q}\ ; id\rangle \rightarrow^{+} \langle \mathcal{Q}'\ ; \sigma\rangle$, cuya longitud es estrictamente mayor que cero, definimos el* resultante *de esta derivación (posiblemente incompleta, si $\mathcal{Q}'$ es una agregación de fórmulas atómicas y valores del retículo) como: $\langle Q\sigma \leftarrow \mathcal{Q}'\ ; \top\rangle$, donde "$\leftarrow$" es cualquier implicación con un conjuntor adjunto.*

Observemos que, en contraste con lo que consideramos en la semántica procedural descrita en la Sección 2.2, los pasos admisibles e interpretativos pueden intercalarse en el cálculo de un resultante. Además, para la construcción de un resultante, pretendemos beneficiarnos de esta ventaja aplicando tantos pasos interpretativos como sea posible antes de ejecutar un paso admisible. Por tanto, en la práctica, concederemos preferencia a los pasos interpretativos sobre los pasos admisibles en el proceso de EP. Este proceso tiene semejanzas con la técnica de *normalización*[8] introducida en el contexto de la programación lógico funcional

[8] En una estrategia de *estrechamiento normalizante* un término se reescribe a su forma normal antes de que se aplique un paso de estrechamiento.

para reducir el indeterminismo de una computación [9]. En consecuencia, también llamaremos *normalización* a la secuencia de pasos interpretativos ejecutados antes de un paso de desplegado operacional.

Asimismo, observemos que la componente "regla" de un resultante no es en general una regla, porque el objetivo $\mathcal{Q}$ puede ser una agregación de fórmulas atómicas. Un resultante es particularmente significativo cuando el objetivo original $\mathcal{Q}$ es un átomo $A$, ya que en ese caso el resultante $\langle A\sigma \leftarrow \mathcal{Q}\ ; \top\rangle$ es un par $\langle regla; grado\ de\ verdad\rangle$ que forma parte del programa transformado. El siguiente ejemplo ilustra la noción de resultante.

*Example 1.* Para el programa $\mathcal{P}$ que sigue y el objetivo $p(X)$, tomando como retículo $([0,1],\leq)$ donde $\leq$ es el orden usual, y las etiquetas `G` y `L` significan lógica intuicionista de *Gödel* y lógica de *Łukasiewicz*, respectivamente,

$$\begin{array}{llll}
\mathcal{R}_1 : \langle p(a) \leftarrow_{\mathtt{L}} q(X,a); & 0{,}7\rangle & \mathcal{R}_5 : \langle s(a) \leftarrow_{\mathtt{G}} t(a); & 0{,}5\rangle \\
\mathcal{R}_2 : \langle p(a) \leftarrow_{\mathtt{G}} s(Y); & 0{,}5\rangle & \mathcal{R}_6 : \langle s(b) \leftarrow_{\mathtt{L}} t(b); & 0{,}8\rangle \\
\mathcal{R}_3 : \langle p(Y) \leftarrow_{\mathtt{G}} q(b,Y) \&_{\mathtt{L}} t(Y); & 0{,}8\rangle & \mathcal{R}_7 : \langle t(a) \leftarrow_{\mathtt{L}} p(X); & 0{,}9\rangle \\
\mathcal{R}_4 : \langle q(b,a) \leftarrow\ ; & 0{,}9\rangle & \mathcal{R}_8 : \langle t(b) \leftarrow_{\mathtt{G}} q(X,a); & 0{,}9\rangle
\end{array}$$

se obtiene el resultante $\langle p(a) \leftarrow 0{,}7 \&_{\mathtt{L}} 0{,}9; 1\rangle$ a partir de la siguiente derivación:

$$\langle p(X); id\rangle \rightarrow_{AS1}{}^{\mathcal{R}_1} \langle 0{,}7 \&_{\mathtt{L}} q(X_1,a); \{X/a\}\rangle \rightarrow_{AS1}{}^{\mathcal{R}_4} \langle 0{,}7 \&_{\mathtt{L}} 0{,}9; \{X/a, X_1/b\}\rangle.$$

Adviértase que, un paso admisible dado con el resultante y el objetivo $p(X)$ simula los efectos de los dos pasos admisibles dados en la anterior derivación, lo que evidencia la mejora introducida por el proceso de EP en el programa transformado final.

Por tanto, en general, el resultante condensa en un sólo paso toda la información de la derivación original correspondiente al objetivo $A$ (de ahí su nombre). También compila la información del grado de verdad en la primera componente del par. Así, para reproducir el efecto de la derivación original, es suficiente tomar como grado de verdad del resultante el supremo del retículo $L$.

Tradicionalmente, la evaluación parcial de un objetivo atómico se define construyendo árboles de búsqueda incompletos para el objetivo y extrayendo la definición especializada —los resultantes— a partir de las ramas (desde la raíz hasta la hoja) que no son de fallo. Para el lenguaje de programación lógica multi-adjunta no existen hojas de fallo, dado que si ninguna regla del programa unifica con el objetivo, se da un paso de computación con la regla $AS2$. En la definición que sigue, adaptamos al nuevo contexto la noción clásica de árbol de desplegado.

**Definition 6 (Árbol de desplegado).** *Sea $\mathcal{P}$ un programa y $\mathcal{Q}$ un objetivo. Un* árbol de desplegado *$\tau_\varphi$ para $\mathcal{P}$ y $\mathcal{Q}$ (usando la regla de computación $\varphi$) es un conjunto de pares de nodos $\langle objetivo; substitución\rangle$ verificando las siguientes condiciones:*

1. *El nodo raíz de $\tau_\varphi$ es $\langle \mathcal{Q}\ ; id\rangle$, donde id es la substitución identidad.*
2. *Si $\mathcal{N}_i \equiv \langle \mathcal{Q}[A]; \sigma\rangle$ es un nodo de $\tau_\varphi$ y suponemos que $\varphi(\mathcal{Q}) = A$ es el átomo seleccionado, entonces para cada regla $\mathcal{R} \equiv \langle H \leftarrow \mathcal{B}; v\rangle$ en $\mathcal{P}$, con $\theta = mgu(\{H = A\})$, $\mathcal{N}_{ij} \equiv \langle (\mathcal{Q}[A/v \& \mathcal{B}])\theta; \sigma\theta\rangle$ es un nodo de $\tau_\varphi$.*

3. *si* $\mathcal{N}_i \equiv \langle \mathcal{Q}[@(r, r')]; \sigma \rangle$ *es un nodo de* $\tau_\varphi$, $\mathcal{N}_{ij} \equiv \langle \mathcal{Q}[@(r, r')/\dot{@}(r, r')]); \sigma \rangle$ *es un nodo de* $\tau_\varphi$.

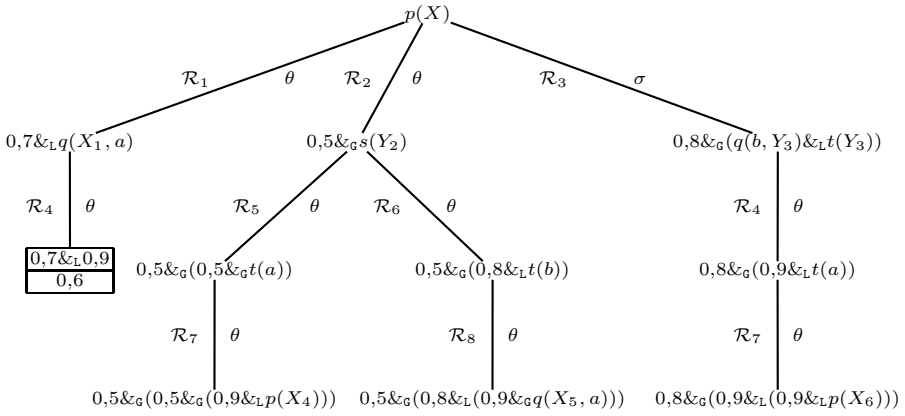
Observemos que el segundo y tercer caso están asociados, respectivamente, a la ejecución de un paso de desplegado operacional y de un paso de desplegado interpretativo, tal como definimos en [14, 15].

Un árbol de desplegado *incompleto* es un árbol de desplegado que puede contener también hojas donde ningún átomo (o expresión interpretable) ha sido seleccionada para un posterior paso de desplegado. Es decir, permitiremos terminar una derivación en cualquier punto adecuado.

**Definition 7 (Evaluación parcial de un átomo).** *Sea* $\mathcal{P}$ *un programa,* $A$ *un objetivo atómico, y* $\tau$ *un árbol de desplegado finito (posiblemente incompleto) para* $\mathcal{P}$ *y* $A$*, conteniendo al menos un nodo además de la raíz.*

*Sea* $\{\mathcal{Q}_i \mid i = 1, \ldots, k\}$ *el conjunto de hojas de* $\tau$*, y* $P' = \{\langle A\sigma_i \leftarrow \mathcal{Q}_i \; ; \top \rangle \mid i = 1, \ldots, k\}$ *el conjunto de resultantes asociado con el conjunto de derivaciones* $\{\langle A \; ; id \rangle \rightarrow^+ \langle \mathcal{Q}_i \; ; \sigma_i \rangle \mid i = 1, \ldots, k\}$*. Entonces, el conjunto* $P'$ *se llama* evaluación parcial *de* $A$ *en* $\mathcal{P}$ *(usando* $\tau$*).*

*Example 2.* Dado el programa $\mathcal{P}$ del Ejemplo 1, podemos construir el siguiente árbol de desplegado de nivel 3 para el programa $\mathcal{P}$ y el átomo $p(X)$:




donde las substituciones de cada estado se anotan en la rama que lleva al mismo, salvo para el nodo raíz que tiene asociada la substitución $id$; por simplicidad las restringimos a las variables del objetivo, de modo que $\theta = \{X/a\}$ y $\sigma = \{X/Y_3\}$. Cada vez que se usa una cláusula se renombran sus variables y, además, los nodos donde los pasos de normalización han originado nodos adicionales, se encierran en cajas. A partir de este árbol obtenemos el conjunto de resultantes

$\mathcal{R}'_1 : \langle p(a) \leftarrow 0{,}6; 1 \rangle$
$\mathcal{R}'_2 : \langle p(a) \leftarrow 0{,}5 \&_{\mathtt{G}} (0{,}5 \&_{\mathtt{G}} (0{,}9 \&_{\mathtt{L}} p(X_4))); 1 \rangle$
$\mathcal{R}'_3 : \langle p(a) \leftarrow 0{,}5 \&_{\mathtt{G}} (0{,}8 \&_{\mathtt{L}} (0{,}9 \&_{\mathtt{G}} q(X_5, a))); 1 \rangle$
$\mathcal{R}'_4 : \langle p(a) \leftarrow 0{,}8 \&_{\mathtt{G}} (0{,}9 \&_{\mathtt{L}} (0{,}9 \&_{\mathtt{L}} p(X_6))); 1 \rangle$

Es sencillo extender la Definición 7 a un conjunto de fórmulas atómicas. Si $S$ es un conjunto finito de átomos, entonces una evaluación parcial de $S$ en $\mathcal{P}$

(también llamada una *evaluación parcial de $\mathcal{P}$ con respecto a $S$*) es la unión de las evaluaciones parciales de los elementos de $S$ en $\mathcal{P}$.

La restricción de especializar objetivos atómicos no es una limitación severa que impida la especialización de objetivos más complejos, si convenimos en realizar la especialización de sus componentes por separado[9]. Dado un programa $\mathcal{P}$ y un objetivo compuesto $\mathcal{Q}$, suponiendo que $S = \{B_1, \ldots, B_n\}$ es el conjunto de constituyentes atómicos de $\mathcal{Q}$, la evaluación parcial de $\mathcal{Q}$ en $\mathcal{P}$ es la evaluación parcial de $\mathcal{P}$ con respecto a $S$.

## 4. Evaluación parcial y cálculo de reductantes

Los reductantes han sido introducidos en el contexto de la programación lógica multi-adjunta para afrontar el problema de incompletitud que se presenta en algunos retículos. Puede ocurrir que sea imposible computar la mayor respuesta correcta, si el retículo $(L, \preceq)$ no es totalmente ordenado [22]: sean $a, b$ elementos no comparables de $L$; supongamos que para un objetivo (básico) $A$ existen sólo dos reglas $\langle H_1 \leftarrow_1 \mathcal{B}_1; v_1\rangle$ y $\langle H_2 \leftarrow_2 \mathcal{B}_2; v_2\rangle$ cuyas cabezas pueden unificar con dicho objetivo; la primera regla contribuye con $(v_1 \&_1 \mathcal{B}_1)\theta_1$, (donde $\theta_1$ es un unificador de $A$ y $H_1$) y conduce a la respuesta computada borrosa $a$; análogamente, la segunda contribuye con $(v_2 \&_2 \mathcal{B}_2)\theta_2$, (donde $\theta_2$ es un unificador de $A$ y $H_2$) y deriva la respuesta computada borrosa $b$; en consecuencia, por la corrección de la semántica procedural, $a$ y $b$ son respuestas correctas y por tanto, $sup\{a, b\}$ es también una respuesta correcta; sin embargo, $sup\{a, b\}$ no es una respuesta computada, por lo que se pierde la completitud del sistema.

Este problema puede resolverse extendiendo el programa original con una regla especial $\langle A \leftarrow @_{sup}((v_1 \&_1 \mathcal{B}_1)\theta_1; (v_2 \&_2 \mathcal{B}_2)\theta_2); \top\rangle$, el llamado *reductante*, que nos permite obtener el supremo de todas las contribuciones para el objetivo $A$.

En esta sección establecemos una relación entre los reductantes y las técnicas clásicas del campo de la evaluación parcial. La siguiente definición formal de reductante es una ligera modificación de la que aparece en [22]. En ella, y en lo que sigue, consideraremos que el agregador $@_{sup}$ tiene función de verdad $\dot{@}_{sup}$ definida por $\dot{@}_{sup}(x_1, \ldots, x_n) = sup\{x_1, \ldots, x_n\}$.

**Definition 8 (Reductante).** *Sea $\mathcal{P}$ un programa y sea $A$ un átomo básico. Si $\{\langle H_i \leftarrow_i \mathcal{B}_i; v_i\rangle \in P \mid$ existe un $\theta_i, A = H_i\theta_i\}$ es el conjunto (no vacío) de reglas de $\mathcal{P}$ cuya cabeza unifica con $A$, entonces el* reductante *de $A$ en $\mathcal{P}$ es la regla $\langle A \leftarrow @_{sup}((v_1 \&_1 \mathcal{B}_1)\theta_1; \ldots; (v_n \&_n \mathcal{B}_n)\theta_n); \top\rangle$.*

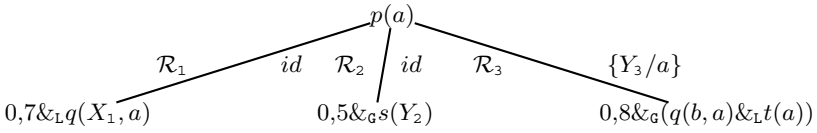
Es importante destacar que un reductante puede construirse usando técnicas propias del campo de la evaluación parcial, tal y como vemos en la siguiente definición:

[9] Sin embargo, es necesario admitir que se pierden algunas posibilidades de alcanzar una buena especialización. Esto puede evitarse introduciendo técnicas adecuadas para la reordenación y partición de objetivos complejos en porciones más pequeñas antes de pasar a su especialización, como se ha propuesto en el campo de la deducción parcial conjuntiva [11, 18].

**Definition 9 (Construcción de reductantes).** *Dado un programa $\mathcal{P}$ y un objetivo atómico básico $A$, enumeramos los siguientes pasos en la construcción de un reductante para $A$ en $\mathcal{P}$:*

1. *Construir el árbol de desplegado* de nivel 1*, $\tau$, para $\mathcal{P}$ y $A$, esto es, el árbol obtenido ejecutando un sólo paso de desplegado sobre el átomo $A$ en $\mathcal{P}$.*
2. *Recolectar el conjunto de hojas $S = \{\langle (v_1 \&_1 \mathcal{B}_1)\theta_1; \theta_1\rangle, \ldots, \langle (v_n \&_n \mathcal{B}_n)\theta_n; \theta_n\rangle\}$*
3. *Construir la regla $\langle A \leftarrow @_{sup}((v_1 \&_1 \mathcal{B}_1)\theta_1; \ldots; (v_n \&_n \mathcal{B}_n)\theta_n); \top\rangle$.*

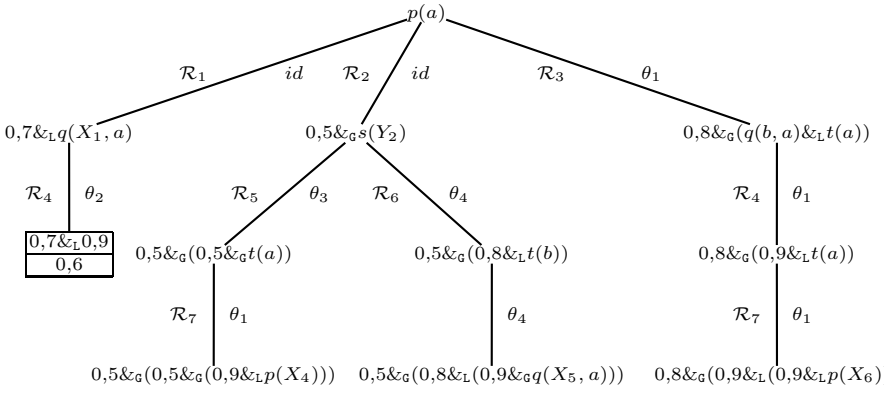
*Example 3.* Dado programa $\mathcal{P}$ del Ejemplo 1 el árbol de desplegado de nivel 1 para el programa $\mathcal{P}$ y el átomo $p(a)$ es el que muestra la figura, en la que mantenemos los convenios de notación expresados en el Ejemplo 2:




a partir del que tenemos el reductante: $\langle p(a) \leftarrow @_{sup}(0{,}7 \&_{\mathtt{L}} q(X_1, a); 0{,}5 \&_{\mathtt{G}} s(Y_2); 0{,}8 \&_{\mathtt{G}} (q(b, a) \&_{\mathtt{L}} t(a))); \ 1\rangle$.

La definición 9 que contempla la construcción de un reductante puede generalizarse, conduciendo a una aproximación más flexible de este concepto, si permitimos desplegar árboles de más de un nivel. Obviamente, usando un árbol de desplegado arbitrario, $\tau$, para un programa $\mathcal{P}$ y un átomo básico $A$, es posible construir una versión más refinada de la noción de reductante.

*Example 4.* La figura muestra un árbol de desplegado para el programa $\mathcal{P}$ del Ejemplo 1 y el átomo básico $p(a)$ de nivel 3 (esto es, todas sus ramas han sido desplegadas no más que 3 pasos).




con $\theta_1 = \{Y_3/a\}, \theta_2 = \{X_1/b\}, \theta_3 = \{Y_2/a\}, \theta_4 = \{Y_2/b\}$. Recogiendo las hojas del mismo obtenemos el reductante $\langle p(a) \leftarrow @_{sup}(0{,}6; 0{,}5 \&_{\mathtt{G}} (0{,}5 \&_{\mathtt{G}} (0{,}9 \&_{\mathtt{L}} p(X_4))); 0{,}5 \&_{\mathtt{G}} (0{,}8 \&_{\mathtt{L}} (0{,}9 \&_{\mathtt{G}} q(X_5, a))); 0{,}8 \&_{\mathtt{G}} (0{,}9 \&_{\mathtt{L}} (0{,}9 \&_{\mathtt{L}} p(X_6)))); \ 1\rangle$.

Dado que esta formulación está basada en técnicas de evaluación parcial, puede verse como un método que produce una especialización de un programa con respecto a un objetivo atómico, que permite computar la mayor respuesta correcta

para el objetivo $A$. Aunque para algún programa $\mathcal{P}$ y átomo básico $A$ es posible obtener distintos reductantes, dependiendo de la precisión del árbol de desplegado generado, afirmamos que todos ellos permiten computar la misma mayor respuesta correcta para el objetivo $A$.

## 5. Conclusiones y trabajo futuro

En este artículo hemos definido por primera vez un marco de evaluación parcial para programas lógicos multi-adjuntos, mostrando su habilidad para especializar programas tal y como se ha hecho tradicionalmente en otros contextos declarativos. Además hemos propuesto un método para calcular *reductantes* usando técnicas de EP, lo que supone una aplicación completamente novedosa en el nuevo contexto. La importancia de relacionar el concepto de EP de un átomo en un programa lógico multi-adjunto con la construcción de un reductante para dicho átomo, estriba en el hecho de que una aproximación computable para este último concepto resulta vital si realmente deseamos implementar sistemas completos para este tipo de lenguajes.

Si bien el método propuesto para el cálculo de reductantes aproximados tiene cierta similitud con las técnicas de tabulación desarrolladas en el marco de los programas (residuados y) multi-adjuntos [8, 7] —específicamente, en lo referente a la contrucción de los bosques de árboles—, las técnicas de tabulación proporcionan un procedimiento para la contestación de preguntas que se basa en el almacenamiento y reutilización de cómputos en tiempo de ejecución, mientras que nuestra técnica adelanta el máximo de cómputos posibles —sin incurrir en problemas de no terminación— en tiempo de evaluación parcial, previamente a la ejecución del programa.

Son varias las extensiones que pueden plantearse para mejorar las técnicas de EP, presentadas en este trabajo, y el cómputo de reductantes aproximados, como por ejemplo: considerar la especialización de conjunciones de átomos en lugar de átomos por separado; e introducir criterios más refinados (como los basados en técnicas de umbralización, órdenes bien fundados, etc) para decidir qué nodos del árbol de desplegado deben o no ser explotados.

## Agradecimientos

Este trabajo ha sido financiado parcialmente por la UE (FEDER) y el Ministerio de Educación y Ciencia, a través del proyecto TIN 2004-07943-C04-03.

## Referencias

[1] E. Albert, M. Alpuente, M. Falaschi, P. Julián, and G. Vidal. Improving Control in Functional Logic Program Specialization. In G. Levi, editor, *Proc. of SAS'98*, pp. 262–277. LNCS 1503, 1998.

[2] E. Albert, M. Alpuente, M. Hanus, and G. Vidal. A Partial Evaluation Framework for Curry Programs. In *Proc. of the LPAR'99*, pp. 376–395. LNAI 1705, 1999.

[3] M. Alpuente, M. Falaschi, and G. Vidal. Partial Evaluation of Functional Logic Programs. *ACM TOPLAS*, 20(4):768–844, 1998.
[4] R.M. Burstall and J. Darlington. A Transformation System for Developing Recursive Programs. *Journal of the ACM*, 24(1):44–67, 1977.
[5] C. Consel and O. Danvy. Tutorial notes on Partial Evaluation. In *Proc. of 20th Annual ACM Symp. on Principles of Programming Languages*, pp. 493–501. ACM, 1993.
[6] C. Consel, L. Hornof, F. Noël, J. Noyé, and E.N. Volanschi. A Uniform Approach for Compile-Time and Run-Time Specialisation. In *Proc. of the 1996 Dagstuhl Seminar on Partial Evaluation*, pp. 54–72. LNCS 1110, 1996.
[7] C.V. Damásio, J. Medina, and M. Ojeda-Aciego. Sorted multi-adjoint logic programs: termination results and applications. In *Proc. of JELIA'04*, pp. 260–273. Springer, LNAI 3229, 2004.
[8] C.V. Damásio, J. Medina, and M. Ojeda-Aciego. A tabulation proof procedure for residuated logic programming. *In Proc. of the European Conf. on AI, Frontiers in AI and Applications*, 110:808–812, 2004.
[9] M. Fay. First Order Unification in an Equational Theory. In *Proc of 4th Int'l Conf. on Automated Deduction*, pp. 161–167, 1979.
[10] J. Gallagher. Tutorial on Specialisation of Logic Programs. In *Proc. of PEPM'93*, pp. 88–98. ACM, 1993.
[11] R. Glück, J. Jørgensen, B. Martens, and M.H. Sørensen. Controlling Conjunctive Partial Deduction of Definite Logic Programs. In *Proc. Int'l Symp. on PLILP'96*, pp. 152–166. LNCS 1140, 1996.
[12] R. Glück and M.H. Sørensen. Partial Deduction and Driving are Equivalent. In *Proc. Int'l Symp. on PLILP'94*, pp. 165–181. LNCS 844, 1994.
[13] N.D. Jones, C.K. Gomard, and P. Sestoft. *Partial Evaluation and Automatic Program Generation*. Prentice-Hall, 1993.
[14] P. Julián, G. Moreno, and J. Penabad. On Fuzzy Unfolding. A Multi-Adjoint Approach. *Fuzzy Sets and Systems*, 154:16–33, 2005.
[15] P. Julián, G. Moreno, and J. Penabad. Operational/Interpretive Unfolding of Multi-adjoint Logic Programs. In F. López-Fraguas, editor, *Proc. of PROLE'2005*, pp. 239–248. University of Granada, 2005.
[16] H.J. Komorowski. Partial Evaluation as a Means for Inferencing Data Structures in an Applicative Language: A Theory and Implementation in the Case of Prolog. In *Proc. of 9th ACM Symp. on Principles of Programming Languages*, pp. 255–267, 1982.
[17] J.-L. Lassez, M. J. Maher, and K. Marriott. Unification Revisited. In J. Minker, editor, *Foundations of Deductive Databases and Logic Programming*, pp. 587–625. Morgan Kaufmann, 1988.
[18] M. Leuschel, D. De Schreye, and A. de Waal. A Conceptual Embedding of Folding into Partial Deduction: Towards a Maximal Integration. In M. Maher, editor, *Proc. of the JICSLP'96*, pp. 319–332. The MIT Press, 1996.
[19] J.W. Lloyd and J.C. Shepherdson. Partial Evaluation in Logic Programming. *Journal of Logic Programming*, 11:217–242, 1991.
[20] J. Medina, M. Ojeda-Aciego, and P. Vojtáš. Multi-adjoint logic programming with continuous semantics. *Proc of LPNMR'01, LNAI*, 2173:351–364, 2001.
[21] J. Medina, M. Ojeda-Aciego, and P. Vojtáš. A procedural semantics for multiadjoint logic programing. *Proc of EPIA'01, LNAI*, 2258(1):290–297, 2001.
[22] J. Medina, M. Ojeda-Aciego, and P. Vojtáš. Similarity-based Unification: a multi-adjoint approach. *Fuzzy Sets and Systems*, 146:43–62, 2004.

[23] A. Pettorossi and M. Proietti. Transformation of Logic Programs: Foundations and Techniques. *Journal of Logic Programming*, 19,20:261–320, 1994.
[24] V.F. Turchin. The Concept of a Supercompiler. *ACM TOPLAS*, 8(3):292–325, July 1986.

# Análisis del movimiento basado en valores de permanencia y lógica difusa

Juan Moreno-Garcia[1], Luis Rodríguez-Benítez[2], Antonio Fernández-Caballero[3] y María T. López[3]

[1] Escuela Universitaria de Ingeniería Técnica Industrial de Toledo
Universidad de Castilla-La Mancha, 45071-Toledo, Spain
Juan.Moreno@uclm.es
[2] Escuela Universitaria Politécnica de Almadén
Universidad de Castilla-La Mancha, 13400-Almadén, Spain
Luis.Rodriguez@uclm.es
[3] Instituto de Investigación en Informática de Albacete (I3A) y
Escuela Politécnica Superior de Albacete
Universidad de Castilla-La Mancha, 02071-Albacete, Spain
{caballer,mlopez}@info-ab.uclm.es

**Resumen** En este artículo se presenta un método de segmentación y de seguimiento de objetos móviles a partir de la llamada matriz de permanencia. En una secuencia de vídeo, nuestros algoritmos basados en movimiento permiten obtener la forma de los objetos en movimiento a partir de aquellos píxeles de la imagen donde se detecta un cambio en el nivel de gris entre dos tramas consecutivas. En la fase de segmentación se buscan las correspondencias entre los objetos a lo largo de la secuencia de imágenes. Con el fin de refinar los objetos se agrupan las regiones de un modo coherente. El artículo también presenta una primera aproximación a una fase de análisis basada en lógica difusa.

## 1. Introducción

La segmentación, el seguimiento y el posterior análisis del movimiento de objetos en secuencias de imágenes [18] es un tema de una importancia creciente en multitud de aplicaciones reales. De ahí que hasta el momento se haya invertido mucha investigación en temas de seguimiento [7],[1],[23]. También se han realizado muchos trabajos en la extracción de formas a partir de secuencias de imágenes ([9],[19],[20],[3],[24], entre otros). Generalmente, todos estos artículos se basan en que el flujo de imagen de un objeto en movimiento varía tanto espacial como temporalmente. Detrás de la mayoría de los enfoques (por ejemplo, [17]) puede verse una agrupación espacial y temporal de píxeles de imagen en regiones coherentes a partir de un conjunto de características comunes. En este artículo presentamos un método para construir espacio-temporalmente las figuras a partir del movimiento inherente presente en una escena [5].

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 725-738, 2006.

Este artículo presenta igualmente los primeros resultados de la aplicación de la lógica difusa a la mejora del análisis de los objetos en movimiento presentes en una secuencia de vídeo. La lógica binaria no permite expresar todas las restricciones que posibilitan componer un objeto a partir de otros de un modo eficiente [2]. Las reglas de decisión lógica permiten una gran reducción de la complejidad y el agrupamiento de los objetos espaciales en clases coherentes [4]. Así pues, la lógica difusa va a proveer el tratamiento de los datos imprecisos necesario para corregir los posibles errores aparecidos durante las fases de segmentación y seguimiento. Por otra parte, la lógica difusa aporta las características necesarias para la definición de la descripción lingüística del movimiento de los objetos [25]. Algunos autores de este trabajo ya tienen experiencia en el uso de etiquetas para describir cambios de un modo lingüístico [12],[13]. La idea principal, de modo similar a [21], consiste en usar etiquetas para indicar las posiciones horizontal y vertical position, así como el tamaño del objeto.

El método propuesto consta de tres fases que coinciden con las tres fases tradicionales utilizadas en visión artificial: la fase de segmentación, la fase de seguimiento (o tracking) y la fase de análisis. Las siguientes secciones describen nuestros algoritmos para cada una de ellas en profundidad, apoyados por un ejemplo práctico en el que una pelota va rodando desde la derecha a la izquierda de una escena (ver figura 1).

**Figura 1.** Una de las imágenes de entrada para el ejemplo usado

## 2. Fase de segmentación

La primera fase es la *fase de segmentación* de las imágenes de la escena. En ésta se hace uso de los vectores de movimiento de una imagen [10],[11], y de un posterior procesado de estos mismos vectores para obtener los objetos que componen cada una de las imágenes de una escena. Para cada imagen se obtiene una lista con $n$ componentes, donde $n$ es el número de objetos en la misma, aportándose, además, el área de los objetos que se encuentran en cada imagen de la escena. Los pasos para obtener los objetos de las imágenes se describen en las siguientes subsecciones.

### 2.1. Obtención de la matriz de permanencia.

Se obtiene una matriz de valores de permanencia, que denominaremos $P$, para cada una de las imágenes que forman la secuencia aplicando el método descrito en [5] y que reproduce en parte el algoritmo 1, donde $NRows$ y $NColumns$ es el número de filas y el número de columnas, respectivamente, de la imagen de

entrada. Debe tenerse en cuenta que la matriz de permanencia $P$ tiene inicializados todos sus valores al valor máximo permitido 255. El valor mínimo para un valor en la matriz de permanencia es 0.

**Algoritmo 1** Obtención de la matriz de permanencia.

```
for i = 1 to NRows do
  for j = 1 to NColumns do
    if M[i, j] = 0 then
      P[i, j] ← max(255, P[i, j] + SUM)
    else
      P[i, j] ← 0
    end if
  end for
end for
```

En el algoritmo ha de entenderse que si no se detecta movimiento en un píxel $[i, j]$ en dos imágenes de entrada consecutivas, es decir cuando $M[i, j] = 0$, la matriz de permanencia va cargándose gradualmente en un valor $SUM$ hasta alcanzar un máximo permitido de 255 en la posición $[i, j]$, mientras que si se detecta movimiento, el valor de permanencia baja al mínimo establecido en 0. En el primer caso, si el movimiento no persiste en el tiempo, la matriz de permanencia va incrementándose gradualmente. Cuando no detecta movimiento, hay una descarga instantánea [6]. La figura 2a muestra la matriz de permanencia $P$ resultado de la detección de todo el movimiento de la pelota en forma de estela trazada.

### 2.2. Umbralización de la matriz de permanencia

En esta fase se umbraliza la matriz de permanencia $P$ de cada una de la imágenes a un valor de umbralización que denominaremos *UMBRAL*. Los valores menores o iguales que *UMBRAL* mantienen su valor, mientras que los valores mayores a éste se cambian al máximo valor de permanencia, en nuestro caso 255. El objetivo de esta fase es eliminar de la matriz de permanencia los valores que reflejan un movimiento de "*hace demasiado tiempo*". Ello se refleja precisamente mediante el valor *UMBRAL*. El resultado de este proceso queda recogido en las matrices umbralizadas $U$. El algoritmo 2 describe este proceso.

La figura 2b muestra la matriz umbralizada $U$ resultado de quedarnos con el movimiento más reciente a partir de la estela completa.

### 2.3. Cálculo de la matriz de vecinos

En esta fase se obtiene una matriz que, para cada una de las celdas, recoge la media con los valores de sus *K-vecinos*, entendiendo *K-vecinos* como los valores que están a una distancia $K$ de una celda. Se calcula la matriz de vecinos $V$ para cada matriz umbralizada $U$, donde el valor de cada celda viene reflejado por la

**Algoritmo 2** Umbralización de la matriz de permanencia.

```
for i = 1 to NRows do
  for j = 1 to NColumns do
    if P[i,j] > UMBRAL then
      U[i,j] ← 255
    else
      U[i,j] ← P[i,j]
    end if
  end for
end for
```

ecuación 1 si el valor obtenido con dicha ecuación es inferior a $UMBRAL$, en otro caso se asignará el valor a 255. Como puede observarse, $K$ es el parámetro que medirá la distancia.

$$V[x,y] = \frac{\sum_{j=y-K}^{y+K} \sum_{i=x-K}^{x+K} U[i,j]}{(1+(2*K))^2} \tag{1}$$

El algoritmo 3 describe este proceso. Como resultado del mismo se obtiene una matriz de vecinos $V$ para cada una de las imágenes de vídeo de entrada.

**Algoritmo 3** Cálculo de las matrices de vecinos.

```
for i = 1 to NRows do
  for j = 1 to NColumns do
    a ← (Σ_{j=y-K}^{y+K} Σ_{i=x-K}^{x+K} U[i,j]) / ((1+(K*2))*(1+(K*2)))
    if a < UMBRAL then
      V[i,j] ← a
    else
      V[i,j] ← 255
    end if
  end for
end for
```

La figura 2c muestra la matriz de vecinos tal como se ha calculado a partir de la matriz umbralizada de la figura 2b. Como puede apreciarse, la matriz de vecinos elimina la sombra de la pelota.

### 2.4. Obtención de los objetos de la matriz de vecinos

A continuación, se utiliza la matriz de vecinos y los parámetros *UMBRAL* y $K$ para agrupar los valores vecinos. El objetivo de este agrupamiento es obtener las celdas que forman parte de cada objeto de la escena a partir de la *matriz de vecinos* $V$. Para ello se realiza un agrupamiento de las celdas vecinas con valor inferior a $UMBRAL$ y que son las que formarán un objeto. Se obtiene una lista de objetos para cada imagen de la secuencia que denominaremos *OBJECTS*.

Para realizar este proceso se utiliza el algoritmo 4. Dicho algoritmo inicializa $OBJECTS$ a $\emptyset$. A continuación recorre la matriz de vecinos $V$, y, si el valor de la celda $V[i,j]$ es inferior a $UMBRAL$, entonces se inserta el par $[i,j]$ en el objeto correspondiente de $OBJECTS$. En caso de que el par no pertenezca a ningún objeto se crea un nuevo objeto con este par únicamente (algoritmo 5).

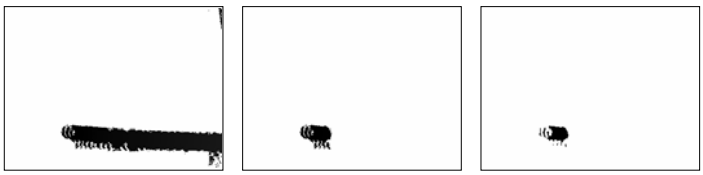



**Figura 2.** Las matrices. (a) Permanencia. (b) Umbralizada. (c) Vecinos

**Algoritmo 4** Obtención de los objetos de la matriz de vecinos.

$OBJECTS \leftarrow \emptyset$
**for** $i = 1$ to $NRows$ **do**
  **for** $j = 1$ to $NColumns$ **do**
    **if** $V[i,j] < UMBRAL$ **then**
      $Insertar([i,j], OBJECTS)$ {utilizando algoritmo 5}
    **end if**
  **end for**
**end for**

El algoritmo 5 recorre $OBJECTS$ buscando algún vecino en los objetos de $OBJECTS$. Si lo encuentra, entonces añade el par al objeto encontrado, si no crea un nuevo objeto con ese par.

### 2.5. Eliminación de los objetos pequeños

Se eliminan de *OBJECTS* los objetos que se consideran pequeños. Esto se realiza con la finalidad de eliminar el ruido de la matriz de valores de permanencia. Para realizar esta fase, se define un parámetro denominado *TAM* que indica el tamaño mínimo de los objetos, es decir, se eliminan de *OBJECTS* los objetos con un tamaño inferior a *TAM*. Este proceso se muestra en el algoritmo 6. Como puede apreciarse fácilmente, lo que hace es recorrer la lista de objetos y si el área en píxeles del objeto es menor que el parámetro $TAM$ entonces elimina el objeto de $OBJECTS$.

**Algoritmo 5** Insertar un par [x,y] en $OBJECTS$.

```
Inserted ← FALSE
while i < |OBJECTS| and Inserted = FALSE do
  if Tiene vecino [x, y] en OBJECTS[i] then
    Insertar([x, y], OBJECTS[i])
    Inserted ← TRUE
  end if
end while
if Inserted = FALSE then
  Crear nuevo objeto con [i, j] y añadirlo a OBJECTS
end if
```

**Algoritmo 6** Eliminación de $OBJECTS$ los objetos pequeños.

```
while i < |OBJECTS| do
  if Area en píxeles de OBJECTS[i] es menor que TAM then
    Eliminar el objeto OBJECTS[i] de OBJECTS
  end if
end while
```

### 2.6. Unión de objetos

En la fase anterior se han detectado varios objetos que en la imagen en realidad podrían ser el mismo; por ello se unen los "*objetos cercanos*.[en] base a un parámetro. Para calcular si se unen, se utiliza la distancia entre las áreas rectangulares de los diferentes objetos. Por ello, es necesario definir el parámetro llamado *distancia* que define la distancia en píxeles para unir los objetos que están *"cercanos"* (distancia inferior al parámetro *distancia*). El algoritmo 7 une los objetos cercanos en una imagen. La idea consiste en recorrer la lista de objetos y si dos objetos son cercanos se unen y se comienza el proceso de nuevo. Este proceso se repite hasta que se realiza una iteración en la que no se consigue realizar ninguna unión de objetos, lo que significa que ya no existen objetos cercanos.

## 3. Fase de seguimiento

En la *fase de seguimiento* se emparejará (proceso de matching) cada objeto de una imagen con su equivalente en los objetos de las imágenes siguientes. Son varias las consideraciones que se han tenido en cuenta:

1. En esta fase se ha considerado que, aunque la calidad del vídeo original de entrada no sea excesivamente buena (trabajamos generalmente con 25 muestras por segundo), un objeto en una imagen debe *"tocar"* al mismo objeto en la imagen siguiente. Esto se refleja en que tiene que existir intersección entre el área rectangular de un objeto de una imagen y el área rectangular del mismo objeto en las imágenes siguientes.

**Algoritmo 7** Unión de objetos cercanos de $OBJECTS$.

```
repetir ← TRUE
while repetir = TRUE do
  repetir ← FALSE
  i ← 0
  while repetir = FALSE and i < |OBJECTS| do
    j ← i + 1
    while repetir = FALSE and j < |OBJECTS| do
      if DistanciaAreas(OBJECTS[i], OBJECTS[j]) < distancia then
        repetir ← TRUE
        Unir(OBJECTS[i], OBJECTS[j])
        Eliminar(OBJECTS[i])
        Eliminar(OBJECTS[j])
      end if
    end while
    j ← j + 1
  end while
  i ← i + 1
end while
```

2. Además, en algunos casos, se detectan objetos que no tienen correspondencia a lo largo de las escenas provocados por el ruido de la matriz de valores de permanencia. En ese caso se desechan dichos objetos al no tener continuidad en las escenas.
3. Otro efecto que puede ocurrir es que haya objetos que no se han detectado en alguna de las imágenes. En ese caso no hay problema porque se tienen bastantes imágenes por segundo y, si en algún caso no se detecta un objeto, se puede recuperar en las imágenes siguientes. Por ello, se hará una selección de las áreas que son correctas (las que tienen correspondencia a lo largo del tiempo) y se eliminarán objetos que "creemos.$^{es}$tán mal detectados.

Para realizar este proceso se parte de una secuencia con los objetos representados en forma de áreas rectangulares, que aparecen para cada una de las imágenes del vídeo y que llamaremos *seq*. La idea del proceso consiste en elegir un objeto de la primera imagen de *seq*, y recorrer el resto de las imágenes para encontrarlo en las mismas, de tal forma que se realiza el emparejamiento del objeto a lo largo de la misma. Los objetos que se van emparejando en las imágenes se eliminan de las mismas. Este proceso se repite para cada uno de los objetos que se encuentran en la primera imagen.

El siguiente paso consiste en elegir un objeto de los que queden de la segunda imagen de *seq* y realizar el mismo proceso. Aquí se pueden dar dos casos: (1) que la segunda imagen ya no tenga objetos porque todos los objetos han sido emparejados con los objetos de la primera imagen; luego, se continúa el proceso con la tercera imagen de *seq*, o, (2) que sí exista algún objeto; luego se procede de la misma manera. Este proceso se repite para todas las imágenes de *seq*.

Como resultado se obtiene una lista con todos los objetos emparejados a lo largo de las imágenes, que están representados como áreas rectangulares.

**Algoritmo 8** Fase de seguimiento.

```
objetos ← ∅
while seq no vacía do
  if seq[1] está vacío then
    Eliminar seq[1] de seq
  else
    area ← ObtenerArea(seq[1]){obtiene un área de seq[1] mandándola a area y
    eliminándola de seq[1]}
    L ← ∅
    L ← L + area
    for i = 2 to |seq| do
      while (|seq| > j) and (no encontrado) do
        if área común de area y seq[i][j] mayor que COMUN then
          L ← L + seq[i][j]
          area ← seq[i][j]
          Eliminar seq[i][j] de seq
          encontrado ← TRUE
        else
          j ← j + 1
        end if
      end while
    end for
    objetos ← objetos + L
  end if
end while
for i = 1 to |objetos| do
  if duracion(objetos[i]) < duracion then
    Eliminar(objetos[i], objetos)
  end if
end for
```

El algoritmo 8 muestra con todo lujo de detalles cómo se ha realizado el proceso en la fase de seguimiento. La variable *objetos* es una lista que contendrá los objetos detectados. Inicialmente está vacía ($objetos \leftarrow \emptyset$). El primer bucle *while* se utiliza para recorrer *seq*, y con la primera sentencia $if$ se comprueba si hay algún objeto en la primera imagen de la secuencia ($seq[1]$) de *seq*. Si está vacía, se quita la primera secuencia y mediante el *while* anterior se continúa el proceso con la siguiente imagen. Si hay algún objeto, se obtiene de la primera imagen y se procede a emparejarlo con el resto de imágenes de la secuencia. $L$ es una lista que tendrá los objetos detectados a lo largo del tiempo; inicialmente está vacía y se añade el primer objeto, es decir, el de la variable *area*. A continuación se recorre *seq* mediante el bucle *for* para buscar el objeto equivalente al objeto de la variable *area* a lo largo de *seq*. El siguiente bucle *while* se utiliza para buscar

el objeto equivalente dentro de la imagen que toca en cada iteración del bucle *for* ($seq[i][j]$, que significa que dentro de la imagen $i$ se accede al objeto $j$). Si el área en común de los dos objetos, $area$ y $seq[i][j]$, es mayor que el parámetro $COMUN$ quiere decir que es el mismo objeto, entonces se añade el objeto a $L$ ($L \leftarrow L + seq[i][j]$). Entonces, $area$ es asignada a $seq[i][j]$ para que en la siguiente iteración se compare el área del último objeto detectado ($area \leftarrow seq[i][j]$), se quita $seq[i][j]$ de $seq$ puesto que ese objeto acaba de ser emparejado (eliminar $seq[i][j]$ de $seq$) y se pone a cierto la variable que termina las iteraciones del bucle *while* ($encontrado \leftarrow TRUE$). Si no se cumple la condición anterior se continúa buscando en los objetos de $seq$.

La última acción de esta fase consiste en eliminar de *objetos* los objetos que no han sido detectados durante un intervalo de tiempo mínimo. Esto se hace mediante un parámetro *duracion* que controla la duración de tiempo mínima del objeto en el vídeo. Si dura menos, se elimina, realizándose esto mediante el último bucle *for*.

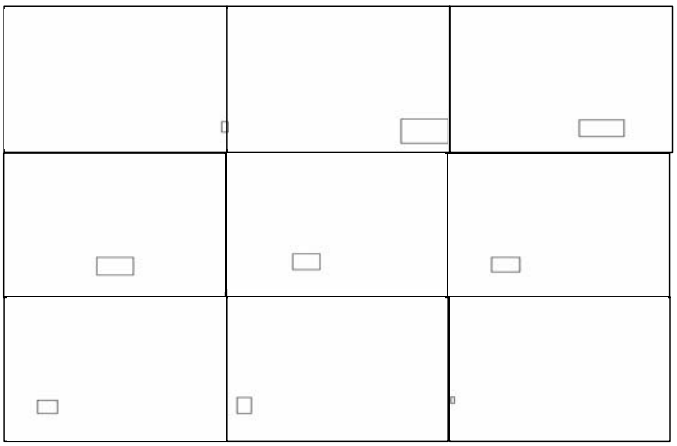


**Figura 3.** Resultado de la fase de seguimiento en distintos instantes de la secuencia

La figura 3 muestra el resultado de la fase de seguimiento en distintos instantes de tiempo de la secuencia ejemplo sobre la pelota. El primer cuadro muestra cómo la pelota va entrando en la escena. El segundo cuadro muestra un área mayor de lo debido, ya que el algoritmo ha incluido la sombra de la pelota. En el último cuadro, vemos cómo la pelota va saliendo de la escena.

## 4. Fase de análisis

En esta fase se pretende localizar la posición de cada objeto a lo largo de la secuencia. De la fase anterior se tiene el área donde está ubicado el objeto. Nosotros hemos supuesto que el objeto está localizado en el centro de dicha área.

Queremos hacer constar que esta posición no es muy precisa, ya que inicialmente se han tomado valores de permanencia distintos de cero, se han eliminado y unido áreas a lo largo del proceso, y en algunas imágenes no se ha detectado el objeto. Por todo esto se ha optado por la lógica difusa [22] para describir el movimiento del objeto a lo largo de la escena.

**Algoritmo 9** Fase de Análisis.

**1.** Cálculo de centros de objetos de las imágenes {Se obtiene una secuencia de listas con los centros de los objetos denominada $C$}
**2.** Cálculo las posiciones y tamaño lingüístico {Se obtiene una secuencia de listas con la etiqueta lingüística para las coordenadas $X$ e $Y$, así como el tamaño lingüístico}
**3.** Agrupación en estados lingüísticos de los objetos de las imágenes {Se obtiene una descripción lingüística temporal de los objetos de la imagen basado en una secuencia de estados con las coordenadas $X$, $Y$ y tamaño del objeto}

El algoritmo 9 muestra cómo se actua en la fase de análisis. En primer lugar, se calcula la posición del centro del área rectangular que contiene al objeto por medio de la ecuación 2, donde $(a, b)$ es el vértice izquierdo superior del área y $(c, d)$ es el vértice derecho inferior del mismo. El área se usará también para calcular el tamaño del objeto. Este proceso se realiza sobre todos los objetos obtenidos en todas las imágenes, obteniendo una secuencia de listas con los centros de los objetos detectados en cada una de las imágenes de la secuencia denominada $C$.

$$\left(\frac{c-a}{2}+a, \frac{d-b}{2}+b\right) \tag{2}$$

A continuación se realiza el cálculo de las etiquetas lingüísticas que reflejan el comportamiento de la posición del objeto y de su tamaño. Para ello se trabaja con variables lingüísticas que pueden tomar valores de un conjunto ordenado de etiquetas lingüísticas con algunas restricciones sobre su dominio. Estas variables se denominan *variables lingüísticas continuas* (desde ahora variables). Las etiquetas lingüísticas (desde ahora etiquetas) de las variables se definen antes de aplicar los algoritmos de esta sección. $X_j$ es una variable lingüística continua si tiene asociada una semántica a cada $SA_j^i$ definida sobre ella que verifica: (1) Cada etiqueta $SA_j^i$ se define para un dominio ordenado; (2) Las etiquetas definidas sobre $X_j$ están ordenadas mediante el MOM; (3) La suma del grado de pertenencia de un valor $x$ a todas las etiquetas definidas en $SA_j$ debe ser 1 ($\sum_{SA_j^i \in SA_j} \mu_{SA_j^i}(x) = 1$).

Se define un conjunto ordenado de etiquetas $SL_X$, $SL_Y$ y $SL_S$ para las variables $X$, $Y$ y $S$ respectivamente. La estructura general de estos conjuntos es $SL = \{SL_j^1 \dots SL_j^{i_j}\}$. En estos conjuntos el superíndice $i$ es la posición de la etiqueta; $j$ será uno de los subíndices $x$,$y$ o $s$; e $i_j = |SL|$. El orden de las etiquetas en los conjuntos de etiquetas se basa en el método de defuzzification llamado *media de máximos* (MOM) [8].

A continuación se calculan las posiciones lingüísticas horizontal $X$ y vertical $Y$, así como el tamaño lingüístico de los objetos $S$. Para ello se recorre la secuencia $C$ calculando la etiqueta lingüística de máxima pertenencia para cada uno de los componentes que describen el objeto (coordenada $X$, coordenada $Y$ y tamaño $S$) obteniendo una secuencia de listas de tuplas de tres componentes $E = \{l_1, l_2 \dots l_n\}$, donde cada componente $l_i$ es una lista de objetos $l_i =< o_1, o_2 \dots o_m >$ y cada objeto es representado mediante $o_j$ que tiene la estructura $o_j = \{ETI_x, ETI_y, ETI_s\}$ con $ETI_x \in SL_X$, $ETI_y \in SL_Y$ y $ETI_s \in SL_S$.

Finalmente se realiza la agrupación temporal de los objetos, para ello se recorre la lista $E$ comparando a lo largo del tiempo la representación de los objetos $o_j$, si son iguales entonces se agrupan, sino se añade el nuevo estado y se comienza de nuevo. La idea es agrupar la representación de los objetos consecutivamente iguales.

Para finalizar esta sección, mostramos cómo funcionaría la fase de análisis en el ejemplo de la pelota. La primera parte calcula los centros de las áreas que definen los objetos calculada en la fase de seguimiento. La tabla 1 muestra el centro de las áreas de algunas imágenes.

**Cuadro 1.** Centro y tamaño de la pelota a lo largo de las imágenes

| *image* | *Centro* | *Tamaño* |
|---|---|---|
| 35 | (200, 309) | 468 |
| 36 | (201, 303) | 990 |
| 37 | (204, 297) | 1680 |
| 38 | (204, 292) | 2080 |
| 39 | (204, 286) | 2520 |
| 40 | (204, 281) | 2960 |
| 41 | (203, 271) | 3075 |
| 42 | (197, 261) | 2072 |
| 43 | (197, 251) | 2016 |
| 44 | (196, 241) | 2088 |
| 45 | (195, 231) | 1944 |
| 46 | (195, 221) | 1863 |

La variable lingüística $X$ toma las etiquetas: *Muy Abajo*, *Abajo*, *Poco Abajo*, *Medio*, *Poco Arriba*, *Arriba* y *Muy Arriba*, mientras que la variable $Y$ puede valer *Muy Izquierda*, *Izquierda*, *Poco Izquierda*, *Medio*, *Poco Derecha*, *Derecha* y*Muy Derecha*. La variable lingüística de tamaño $S$ acepta las etiquetas *Muy Pequeño*, *Pequeño*, *Medio*, *Grande* y *Muy Grande*.

La tabla 2 muestra los resultados obtenidos para el ejemplo entre las imágenes 35 y 45 del vídeo. En la primera columna tenemos el número de la trama, las segunda y tercera columna son las etiquetas obtenidas para las variables $X$ e $Y$, respectivamente, y, por último, tenemos la etiqueta para la variable $S$. Como puede verse, la etiqueta lingüística para $X$ no varía, al no haber cambio alguno

**Cuadro 2.** Descripción lingüística del movimiento de la pelota

| *image* | *X* | *Y* | *S* |
|---|---|---|---|
| 35 | Abajo | Muy Derecha | Muy Pequeño |
| 36 | Abajo | Muy Derecha | Muy Pequeño |
| 37 | Abajo | Muy Derecha | Muy Pequeño |
| 38 | Abajo | Muy Derecha | Muy Pequeño |
| 39 | Abajo | Muy Derecha | Muy Pequeño |
| 40 | Abajo | Muy Derecha | Muy Pequeño |
| 41 | Abajo | Derecha | Muy Pequeño |
| 42 | Abajo | Derecha | Muy Pequeño |
| 43 | Abajo | Derecha | Muy Pequeño |
| 44 | Abajo | Derecha | Muy Pequeño |
| 45 | Abajo | Poco Derecha | Muy Pequeño |

de movimiento en el eje $x$. La etiqueta lingüística para la variable $Y$ evoluciona desde *Muy Derecha*, pasando por *Derecha*, hasta *Poco Derecha*; ello demuestra que la pelota va rodando de derecha a izquierda. En cuanto a $S$, se mantiene el valor de *Muy Pequeño* a lo largo de todas las imágenes.

**Cuadro 3.** Estados lingüísticos del movimiento de la pelota

| *images* | *X* | *Y* | *S* |
|---|---|---|---|
| 35 . . . 40 | Abajo | Muy Derecha | Muy Pequeño |
| 41 . . . 44 | Abajo | Derecha | Muy Pequeño |
| 45 . . . | Abajo | Poco Derecha | Muy Pequeño |

El último paso consiste en realizar la agrupación temporal de los objetos consecutivos definidos con la mismas etiquetas para cada una de las variables en lo que hemos denominado .estados lingüísticos". De tal forma que cada estado representa cómo durante una serie de instantes consecutivos se define el movimiento de un objeto de forma similar a [14]. Para el ejemplo de la tabla 2 se obtendrían los estados mostrados en la tabla 3. Con estos estados se tiene representado el posicionamiento y tamaño de un objeto dentro de una escena durante un intervalo de tiempo. Por ejemplo, del primer estado se puede concluir que de las imágenes 35 a 40 hay un objeto *Muy Pequeño* con centro posicionado *Abajo* y *Muy a la Derecha*. Estudiando el segundo estado podemos concluir que el objeto se desplaza a la izquierda, ya que en el primer estado la variable $Y$ tiene el valor *Muy Derecha* y en el siguiente tiene el valor *Derecha*, luego el movimiento es hacia la izquierda. Con estas estructuras la extracción de conocimiento de la escena se puede realizar de forma automática en una notación cercana al humano mediante el uso de etiquetas lingüísticas definidas a priori. Además, permite realizar la comparación de trayectorias de diferentes objetos

en las escenas mediante algoritmos similares a los presentados en [15] y [16]. Actualmente estamos trabajando en esta fase.

## 5. Conclusiones

En este artículo se han propuesto unos sencillos algoritmos capaces de segmentar y de seguir todos los objetos móviles de una secuencia de vídeo, de un modo robusto. Nuestro método puede compararse a los métodos de substracción del fondo o de diferencias de imágenes en el modo en que se detecta el movimiento. Después se aplica una técnica de crecimiento de regiones basada en valores de permanencia asignados a los píxeles. Para mejorar los resultados de la segmentación y del seguimiento se realizan algunas operaciones de refinamiento de los objetos inicialmente obtenidos. Cabe señalar que el modelo propuesto no tiene limitaciones en cuanto al número de objetos a diferenciar. Todos los objetos de una escena se segmentan de un modo claro.

La fase de análisis se está desarrollando todavía. Se usa lógica difusa debido a la inherente imprecisión de los procesos de segmentación y seguimiento, con el fin de obtener un método mucho más descriptivo. En esta fase se ha mostrado un ejemplo de cómo poder describir el movimiento de los objetos. Como trabajo futuro, tenemos la intención de mejorar el proceso de agrupamiento presentado en este trabajo y la eleboración de algoritmos para permitir la comparación de trayectorias de forma automática.

## Agradecimientos

Este trabajo está financiado en parte por los proyectos nacionales CICYT TIN2004-07661-C02-02 y CICYT TIC2003-08807-C02-02, así como el proyecto regional de la Junta de Comunidades de Castilla-La Mancha PBI06-0099.

## Referencias

1. Aggarwal, J.K., Nandhakumar, N.: On the computation of motion from sequences of images - A review. Proceedings of the IEEE **76**:8 (1988)
2. Bandemer, H., Gottwald, S.: Fuzzy logic, fuzzy methods with Applications. Wiley & Sons (1996)
3. Bobick, A.F., Davis, J.W.: An appearance-based representation of action. Proceedings of the 13th International Conference on Pattern Recognition (1996) 307–312
4. de Kok, R., Schneider, T., Ammer, U.: Object-based classification and applications in the Alpine forest environment. International Archives of Photogrammetry and Remote Sensing **32** Part 7-4-3 W6 (1999)
5. Fernández, M.A., Fernández-Caballero, A., López, M.T., Mira, J.: Length-speed ratio (LSR) as a characteristic for moving elements real-time classification. Real-Time Imaging **9** (2003) 49–59
6. Fernández-Caballero, A., Fernández, M.A., Mira, J., Delgado, A.E.: Spatio-temporal shape building from image sequences using lateral interaction in accumulative computation. Pattern Recognition **36**:5 (2003) 1131–1142

7. Huang, T.S.: Image Sequence Analysis. Springer-Verlag (1983)
8. Leekwijck, W. V. , Kerre, E. E.: Defuzzification: criteria and classification, Fuzzy Sets and Systems: 108, 2 (1999), 159-178.
9. Little, J.J., Boyd, J.E.: Recognizing people by their gait: The shape of motion. Videre: Journal of Computer Vision Research **1**:2 (1998) 2–32
10. López, M.T., Fernández-Caballero, A., Fernández, M.A., Mira, J., Delgado, A.E.: Motion features to enhance scene segmentation in active visual attention. Pattern Recognition Letters **27**:5 (2006) 429-438
11. Mira, J., Delgado, A.E., Fernández-Caballero, A., Fernández, M.A.: Knowledge modelling for the motion detection task: The algorithmic lateral inhibition method. Expert Systems with Applications **27**:2 (2004)169–185
12. Moreno-Garcia, J., Jimenez, L., Castro-Schez, J.J., Rodriguez, L.: A direct linguistic induction method for systems. Fuzzy Sets and Systems **146** (2004) 79–96
13. Moreno-Garcia, J., Jimenez, L. ,Castro-Schez, J.J., Rodriguez, L.: A linguistic modelling approach for dynamic systems by using temporal fuzzy models. Submitted to International Journal of Approximate Reasoning (2004)
14. Moreno-Garcia, J., Castro-Schez, J.J., Jimenez, L.: A fuzzy inductive algorithm for modeling dynamical systems in a comprehensible way. To be published in IEEE Transactions on Fuzzy Systems (2005)
15. Moreno-Garcia, J., Castro-Schez, J.J.: Comparing dynamical systems by using Temporal Fuzzy Modelss. Proceedings of Information Processing and Management of Uncertainty in Knowlegment-Based Systems 2006 (IPMU 2006).
16. Moreno-Garcia, J., Castro-Schez, J.J., Jiménez, L.: Comparación de sistemas dinámicos mediante Cadenas Difusas Temporales. Sometido a VIII Congreso Español sobre Tecnologías y Lógica Fuzzy (2006).
17. Olson, T., Brill, F.: Moving object detection and event recognition algorithms for smart cameras. Proceedings of the DARPA Image Understanding Workshop (1997) 159–175
18. Pérez de la Blanca, N., Fuertes, J.M., Lucena, M.: Deformable object matching based on multi-scale local histograms. Lecture Notes in Computer Science **3179** (2004) 154–162
19. Polana, R., Nelson, R.: Detecting activities. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (1993) 2–7
20. Polana, R., Nelson, R.: Recognition of nonrigid motion. Proceedings DARPA Image Understanding Workshop (1994) 1219–1224
21. Rodriguez-Benitez, L., Moreno-García, J., Castro-Schez, J.J, Jimenez, L, Garcia, S.: Linguistic motion description for an object on Mpeg compressed domain. Proceedings of the Eleventh International Fuzzy Systems Association World Congress **II** (2005) 1064–1070
22. Tanaka, K.: An Introduction to Fuzzy Logic for Practical Applications. Springer-Verlag (1998)
23. Wang, J., Huang, T.S., Ahuja, N.: Motion and Structure from Image Sequences. Springer-Verlag (1993)
24. Yang, M.-H., Ahuja, N.: Extracting gestural motion trajectories. Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition (1998) 10–15
25. Zadeh, L.A.: The concept of a linguistic variable and its applications to approximate reasoning, Part I, II and III. Information Science **8** (1975) 199–249, **8** (1975) 301–357, **9** (1975) 43–80

# Estrategias cooperativas paralelas con uso de memoria basadas en Soft Computing

Carlos Cruz, David Pelta, Alejandro Sancho Royo y José Luis Verdegay

Grupo de Modelos de Decisión y Optimización
Dpto. de Ciencias de la Computación e I.A
Universidad de Granada, España

**Resumen** En este trabajo se dispone de una serie de algoritmos de optimización controlados por un proceso coordinador y que se ejecutan de forma concurrente resolviendo la misma instancia del problema. Se modela una base de reglas difusas que hacen uso de una memoria que almacena el historial del comportamiento de los algoritmos, y que permiten controlar y modificar el comportamiento de cada uno de ellos. Los resultados alcanzados sobre el problema de la mochila y el problema de la p-mediana muestran los beneficios de esta propuesta.
*Palabras clave:* metaheurísticas, optimización, paralelismo, Soft Computing

## 1. Introducción

En la literatura aparecen varias propuestas que intentan la paralelización de metaheurísticas [1, 5, 13] con buenos resultados. En particular, se han incrementado los trabajos relacionados con los métodos de búsqueda múltiple.

De ellos, se destaca últimamente el diseño de implementaciones asíncronas cooperativas multihebras, o también llamadas búsquedas cooperativas multinivel [3, 4]. En estas se ha observado que la cooperación debida al intercambio de información no solo mejora el factor de aceleración sino que modifica los patrones de búsqueda de los programas obteniendo un alto rendimiento [2].

En este trabajo presentamos una estrategia cooperativa paralela multihebras que hace uso de la memoria para mejorar el proceso de búsqueda y cuyos mecanismos de cooperación se han modelados con técnicas de Soft Computing.

## 2. Estrategia propuesta

La idea de este trabajo puede resumirse imaginando un comité de expertos encargado de resolver algún problema concreto, y dirigido por un coordinador que conoce los aspectos generales relativos al problema en cuestión y las capacidades y características de cada uno de los miembros del equipo. Además, este coordinador sabe cuándo puede considerarse aceptable una solución obtenida.

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 739-750, 2006.

Cada miembro del comité es un experto en resolver el problema mediante un tratamiento específico, que puede ser compartido por más de uno. En este contexto, cada uno de los expertos del comité trabaja de forma independiente; pero todos al mismo tiempo ejecutando algoritmos de optimización. O sea, ellos poseen el conocimiento necesario para resolver un problema de optimización específico y a los cuales se les envía desde el coordinador directivas de ajuste de su comportamiento.

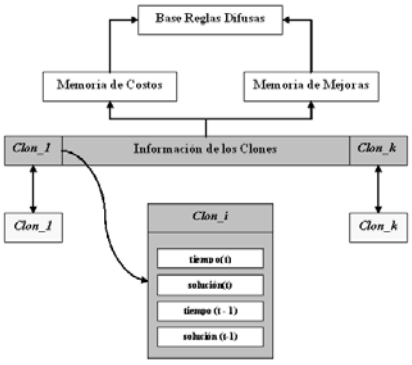



**Figura 1.** Esquema de la Estrategia

En el diseño presentado cada experto se modela a través de un algoritmo de búsqueda por entornos, adaptable y difuso, FANS [7], y su elección se debe a su facilidad para obtener comportamientos de diferentes heurísticas [8] al variar sus parámetros de ajuste, además debido a que es una técnica de búsqueda local, fácil de comprender, implementar y que no necesita muchos recursos computacionales.

Así, la idea del comité se diseña como una estrategia cooperativa multihebras, organizada jerárquicamente en dos niveles, bajo un esquema de coordinación de planificación centralizada (Ver Figura 1), en la que en el nivel superior actúa un agente Coordinador, y en el segundo nivel actúan agentes Resolvedores sin comunicación entre ellos y cuya función básica es ejecutar el algoritmo FANS para solucionar la instancia del problema en cuestión.

### 2.1. Implementación de la estrategia

A continuación, con la ayuda de los pseudocódigos, se describe con más detalles esta propuesta. En la Figura 2 se muestra el pseudocódigo para el Coordinador.

En un primer paso, el Coordinador determina el comportamiento inicial (conjunto de parámetros) para cada agente Resolvedor. El Coordinador pasa dichos comportamientos a cada agente y entonces estos se ejecutan.

La comunicación entre el Coordinador y los agentes Resolvedores se realiza a través de mensajes de intercambio donde los agentes envían la información sobre

```
Procedimiento Coordinador:
Begin
  /* Agente_i, agente resolvedor */
  /* Par_i, conjunto de parámetros */
  /* Coordinador */
  /* MC_i, memoria para almacenar la información de cada Agente_i */
  /* FRB, La Base de reglas difusas del Coordinador */
  For i := 1 To k Do
    obtener Par_i para Agente_i usando la FRB;
    ajustar Agente_i con Par_i;
    Agente_i comienza a ejecutar;
  Od
  Repeat Until ( no se alcance el criterio de parada ) Do
    MC_i es actualizada por Agente_i (asincrónicamente);
    For (cada Agente_i que haya informado nuevos resultados) Do
      Coordinador decide si Agente_i necesita ser cambiado;
      If (SI) Then
        obtener Par_i para Agente_i usando la FRB;
        Coordinador almacena Par_i en MC_i;
      Fi
    Od
  Od
  For i := 1 To k Do
    detener Agente_i;
  Od
End.
```

**Figura 2.** Pseudo código del Coordinador

```
Procedimiento Agente_i:
Begin
  Repeat Until ( no haya orden de parar ) Do
    verificar si hay información del Coordinador en MC_i;
    If (hay nueva información) Then
      adaptar los parámetros internos;
    Fi
    ;
    /* aquí se define la estrategia de Agente_i */
    ;
    If (se verifican las condiciones para comunicación) Then
      enviar información de su desempeño al Coordinador;
    Fi
  End.
```

**Figura 3.** Pseudo código del Agente Resolvedor.

su desempeño y reciben órdenes de adaptación; el Coordinador recibe y almacena comportamientos y recupera medidas de desempeño que usa para adaptar su Base de Reglas interna. Los agentes se encuentran en ejecución resolviendo el problema, enviando los resultados y esperando las directivas del Coordinador.

El Coordinador almacena los costos $f(s^t)$ producidos por cada algoritmo y un historial de las tasas de mejoras definido por la siguiente ecuación:

$$\Delta_f = \frac{(f(s^t) - f(s^{t-1}))}{(time^t - time^{t-1})}$$

de forma tal que, ante un nuevo informe, calcula la "calidad" de la solución reportada, en términos del historial almacenado y donde $time^t - time^{t-1}$ representa el tiempo transcurrido entre dos informes consecutivos.

Entonces, los valores $\Delta_f$ y $f(s^t)$ y son guardados en arreglos ordenados separados a longitud fija o "memorias" $M_{\Delta_f}, M_f$.

El Coordinador comprueba qué agente proporciona nueva información y decide si su comportamiento necesita ser adaptado. En este caso, usa su base de reglas para obtener un nuevo comportamiento que será enviado al agente. Cuando se cumple el criterio de parada, el Coordinador detiene la ejecución de la estrategia.

La operación de los agentes Resolvedores es sencilla (Ver Figura 3). Cada uno se ejecuta de forma asíncrona, alternando el envío del comportamiento de su desempeño (solución actual, su coste y tiempo) y la recepción de las directivas del Coordinador. Como FANS es una heurística basada en búsqueda local, se decidió que la información enviada por el Coordinador a cada algoritmo sea una nueva solución, o sea, un nuevo punto en el espacio de búsqueda o cambiarle su comportamiento mediante un ajuste de algún parámetro. Cuando el algoritmo lo recibe, vuelve a empezar la búsqueda a partir de ese nuevo punto o en el otro caso bajo un esquema de búsqueda modificado.

### 2.2. Base de Reglas Difusas

En la Base de Reglas, se implementan las siguientes:

1. **Regla SM, Sin Memoria:**

| |
|---|
| **IF** la solución del $agente_i$ es **peor** |
| que la mejor vista hasta el momento |
| **THEN** enviarle al $agente_i$ la mejor solución. |

   Bajo esta regla si el $agente_i$ reporta una solución *peor* respecto a la mejor de todas las que ya se han visto entonces se le indica que se "mueva" en el espacio de búsqueda asignándole como solución de reinicio esa mejor solución. Entiéndase como *peor* que el costo de la solución obtenida por el $agente_i$ sea mayor o menor a la mejor de todas las ya vistas, según se maximice o minimice en el problema. Esta es una reglas "crisp" y no se hace uso de la memoria, al no tener en cuenta el historial del comportamiento de los agentes.

2. **Regla MR1, con Memoria e independiente de los agentes:**

| |
|---|
| **IF** la calidad de la solución del $agente_i$ es **mala** |
| *AND* razón de mejora del $agente_i$ es **mala** |
| **THEN** enviarle al $agente_i$ la mejor solución. |

   La etiqueta difusa *mala* se define por la siguiente función de pertenencia:

$$\mu(x,\alpha,\beta) = \begin{cases} 0 & \text{si } x > \beta \\ (\beta - x)/(\beta - \alpha) & \text{if } \alpha \leq x \leq \beta \\ 1 & \text{si } x < \alpha \end{cases}$$

Donde $x$ es el rango percentil de la variable que se trate, ya sea el costo de la solución, $f(s^t)$, o la tasa de mejora, $\Delta_f$, de entre los valores almacenados en la memoria. Los otros parámetros fueron fijados empíricamente a los siguientes valores: $\alpha = 10$ y $\beta = 20$.
O sea, si el $agente_i$ reporta soluciones "malas" respecto a las anteriores y si su tasa de mejora entre dos reportes consecutivos es de las peores, lo que se hará es "moverlo" en el espacio de búsqueda asignándole la mejor solución vista por el Coordinador hasta ese momento.
De esta forma si denominamos por $\mu_{costo}$ y $\mu_{tasa}$ a los valores obtenidos en la función de pertenencia para $x = f(s^t)$ y $x = \Delta_f$, el antecedente de las reglas se calcula como:

$$\min\ (\mu_{costo},\ \mu_{tasa})$$

y el consecuente se produce si este valor anterior es mayor que un cierto umbral fijado empíricamente en 0.7. Esta regla es independiente del tipo de agentes resolvedores que use la estrategia.

3. **Regla MR2, con Memoria y dependiente de los agentes:**

| |
|---|
| **IF** la calidad de la solución informada por $agente_i$ es **mala** |
| *AND* razón de mejora del $agente_i$ es **mala** |
| **THEN** enviarle al $agente_i$ la mejor solución |
| *AND* ajustar el valor de $\lambda$ del $agente_i$. |

En esta regla también se lleva un historial de las tasas de mejoras y los costos producidos por los algoritmos, de forma tal que, ante un nuevo informe, se calcula la "calidad" de la solución reportada. Se usa la misma función difusa de pertenencia para definir la etiqueta *mala*.
La diferencia en esta Regla está en que además de lo que hace la regla MR1 se le añade *ajustar* el valor de uno de los parámetros de FANS que permite variar su comportamiento de búsqueda [8], o sea el valor de $\lambda$ del $Agente_i$. Se le enviará un nuevo valor de $\lambda$, denominado $\lambda_i^{'}$, que acerca el valor de $\lambda_i$ al del agente que haya obtenido la mejor solución vista hasta el momento, $\lambda^*$. Para ello, se define una etiqueta difusa $\Delta_\lambda$ de esta forma:

$$\Delta_\lambda = \mu_\delta^{-1}(min(\mu_{\cos to}, \mu_{tasa}))$$

donde, si $\left|\lambda^* - \lambda^i\right| \neq 0$ :

$$\mu_\delta(x) = \begin{cases} \frac{x}{|\lambda^* - \lambda_i|} & \text{si} \quad 0 \leq x \leq 1 \\ \\ 1 & \text{si} \quad x \geq 0 \end{cases}$$

si $\left|\lambda^* - \lambda^i\right| = 0$ : entonces $\mu_\delta(x) = 1, \forall x$, en este caso la expresión de $\Delta_\lambda$ carece de sentido, pero se hace innecesario su cálculo ya que $\lambda^* = \lambda^i$ y por lo tanto $\Delta_\lambda = 0$.
De esta forma $\lambda_i^{'}$ queda definido como:

$$\lambda_i^{'} = \begin{cases} min(1, \lambda_i + \Delta_\lambda) & \text{si} \quad \lambda_i \leq \lambda^* \\ \\ max(1, \lambda_i - \Delta_\lambda) & \text{si} \quad \lambda_i > \lambda^* \end{cases}$$

4. **Regla RR1, retrasa el uso de la regla MR1:**

| |
|---|
| **IF** tiempo de ejecución del $agente_i \leq \frac{1}{3}$ tiempo total de ejecución |
| **THEN** usar la Regla SM. |
| **ELSE IF** tiempo de ejecución del $agente_i > \frac{1}{3}$ tiempo total de ejecución |
| **THEN** usar la Regla MR1. |

5. **Regla RR2, retrasa el uso de la regla MR2:**

| |
|---|
| **IF** tiempo de ejecución del $agente_i \leq \frac{1}{3}$ tiempo total de ejecución |
| **THEN** usar la Regla SM. |
| **ELSE IF** tiempo de ejecución del $agente_i > \frac{1}{3}$ tiempo total de ejecución |
| **THEN** usar la Regla MR2. |

Estas dos reglas se definen para permitir a los agentes resolvedores que se estabilicen en su búsqueda evitando de este modo convergencias prematuras hacia las zonas de los mejores valores encontrados al inicio y por tanto permitiendo una mayor diversificación en la estrategia durante esta fase. Por tanto, lo que se hace es demorar el uso de la Regla **MR1** o **MR2** un tiempo determinado, en este caso un tercio del tiempo total de ejecución asignado, y mientras se usará la Regla **SM**.

## 3. Verificación experimental

Para comprobar la validez de la estrategia propuesta y analizar el impacto del uso de la memoria en la búsqueda de mejores soluciones se utilizarán los siguientes problemas: el problema de la mochila "clásico" y el problema de la p-mediana. Todos los experimentos se realizaron en un cluster de ordenadores bajo el paquete de software PVM (Parallel Virtual Machine). La configuración del cluster de computadores usado es la siguiente: 8 ordenadores AMD Athlon duales, con 2 GHz, 256 k en la memoria caché, y 2 GBytes de RAM, con el Sistema Operativo Linux Fedora Core release 3. Basado en resultados anteriores en cuanto al número de agentes [9] se determinó usar 6 clones de FANS, que se configuran bajo esquemas de búsqueda diferentes teniendo en cuenta que esto se logra variando uno de sus parámetros [8].

### 3.1. Problema de la mochila: Formulación del problema

El problema de la mochila [6] puede ser formalmente definido así: dado un conjunto $N$, compuesto de $n$ items $i$ con un beneficio o valor $p_i$ y peso $w_i$, y un valor $C$ de capacidad de la mochila. El objetivo es seleccionar un subconjunto de $N$ tal que el total de los beneficios de los items seleccionados sea máximo y el total de los pesos no exceda a $C$. Su formulación matemática es:

$$Max \sum_{i=1}^{n} p_i \times x_i$$

$$s.t. \sum_{i=1}^{n} w_i \times x_i \leq C, x_i \in \{0,1\}, i = 1, \ldots, n$$

donde
$n$ es el número de items,
$x_i$ indica si el item $i$ está incluido o no en la mochila,
$p_i$ es el beneficio asociado al $i$,
$w_i \in [0, .., r]$ es el peso del item $i$,
$C$ es la capacidad de la mochila.
Se asume que $w_i < C$, $\forall i$; y $\sum_{i=1}^{n} w_i > C$

En este problema se pueden generar instancias diferentes teniendo en cuenta: tamaño de las instancias, tipo de correlación entre los pesos y los beneficios y rango de valores disponibles para los pesos. Se pueden construir instancias muy duras de resolver y que constituyen todo un reto para la mayoría de los algoritmos exactos de la actualidad [6]. De ellas, las de tipo span$(v, m)$, son construidas de manera que todos sus items son múltiplos de un pequeño conjunto de items llamado conjunto llave y se caracterizan por los siguientes parámetros: $v$ es el tamaño del conjunto llave, $m$ es el limite multiplicador y en el conjunto llave se puede tener cualquier distribución de los items.

El conjunto $v$ de items se genera con los pesos con distribución uniforme en el intervalo [1, R] y los beneficios acordes a la distribución. Los items ($p_k$, $w_k$) en el conjunto llave se normalizan dividiendo los pesos y los beneficios entre $m + 1$. Los items se construyen tomando repetidamente un ítem ($p_k$, $w_k$) del conjunto llave y un multiplicador generado aleatoriamente en el intervalo [1, m]. El ítem construido queda así ($a \times p_k$, $a \times w_k$).

Para este experimento se consideraron tres distribuciones en las cuales el límite multiplicador se tomó $m = 10$ y tal como los experimentos computacionales lo demuestran [6] las instancias mas duras son para valores pequeños de $v$.

- Instancias No Correlacionadas span
  NC: $w_i = U(1, R), p_i = U(1, R), span(2, 10)$.
- Débilmente Correlacionadas span
  WC: $w_i = U(1, R), p_i = U(w_i - \frac{R}{10}, w_i + \frac{R}{10})$ con $p_i \geq 1$ y $span(2, 10)$
- Fuertemente Correlacionadas span
  SC: $w_i = U(1, R), p_i = w_i + \frac{R}{10}, span(2, 10)$.

La Capacidad de la Mochila se calcula de esta manera:

$$C = \frac{h}{1+H} \sum_{i=1}^{n} w_i,$$

donde $H = 100$ y $h$ es el número de instancia.

Se conoce el óptimo de cada instancia, calculado con el mejor algoritmo exacto desarrollado por Pisinger. Usando este valor, se calcula el *Error*:

$$Error = 100 * \frac{Optimo - Costo\,Obtenido}{Optimo}$$

### 3.2. Problema de la mochila: Experimentos

En este experimento se probarán cada una de las reglas definidas y se tomará la regla Sin Memoria, **SM**, como base de la comparación. Se toman 3 tamaños del problema, $n = (1000, 1500, 2000)$, y los pesos $w_i$ se distribuyen uniformemente con $R = \{10000\}$ que fueron los casos más difíciles de resolver en experimentos anteriores [10, 11]. Se ejecuta 10 veces la estrategia durante 15 segundos para cada instancia y regla.

El primer análisis se enfoca en las tres primeras reglas y como se muestra en la Tabla 1 el uso de la memoria, con las reglas **MR1** y **MR2**, muy pocas veces alcanzan o mejoran la calidad de las soluciones obtenidas con la regla **SM**. Con estas tres reglas no se alcanza a nivel global de la estrategia un equilibrio adecuado entre la exploración y la explotación del espacio de búsqueda.

**Cuadro 1.** Promedio de Error y Desviación Típica para cada regla y tamaño.

*NCspan*

| **Regla** | 1000 | 1500 | 2000 |
|---|---|---|---|
| SM | 3,73 ( 3,66 ) | 2,87 ( 3,18 ) | 3,61 ( 3,37 ) |
| MR1 | 3,92 ( 4,06 ) | 3,10 ( 3,39 ) | 3,68 ( 3,57 ) |
| MR2 | 3,00 ( 3,68 ) | 2,72 ( 3,41 ) | 3,57 ( 3,61 ) |
| RR1 | 3,85 ( 3,78 ) | 3,09 ( 3,64 ) | 3,61 ( 3,40 ) |
| RR2 | 3,33 ( 3,74 ) | 2,65 ( 3,22 ) | 3,25 ( 3,36 ) |

*WCspan*

| **Regla** | 1000 | 1500 | 2000 |
|---|---|---|---|
| SM | 2,19 (1,59) | 2,45 (1,59) | 3,04 (1,95) |
| MR1 | 2,37 (1,72) | 2,67 (1,77) | 3,20 (2,09) |
| MR2 | 2,18 (1,61) | 2,69 (2,10) | 3,24 (2,32) |
| RR1 | 2,18 (1,52) | 2,48 (1,59) | 3,00 (1,91) |
| RR2 | 2,07 (1,56) | 2,30 (1,57) | 2,88 (1,86) |

*SCspan*

| **Regla** | 1000 | 1500 | 2000 |
|---|---|---|---|
| SM | 1,96 ( 1,43 ) | 1,90 ( 1,32 ) | 2,32 ( 1,79 ) |
| MR1 | 2,10 ( 1,55 ) | 2,15 ( 1,58 ) | 2,65 ( 2,23 ) |
| MR2 | 2,43 ( 2,56 ) | 2,41 ( 2,48 ) | 2,96 ( 3,00 ) |
| RR1 | 1,92 ( 1,37 ) | 1,92 ( 1,31 ) | 2,35 ( 1,85 ) |
| RR2 | 1,86 ( 1,31 ) | 1,80 ( 1,20 ) | 2,23 ( 1,69 ) |

El hecho básico de tener varios agentes diferentes trabajando de forma paralela, que se iniciaron a partir de soluciones diferentes y sólo almacenando la mejor solución global encontrada por ellos en el tiempo de ejecución, como hace la regla **SM**, parece ser suficiente para alcanzar soluciones de calidad. Consideramos que esto puede mejorarse, ya que se nota que hay cierta convergencia de los agentes en aquella zona donde se alcanzan los primeros mejores valores; y por ello, se intentará resolver permitiendo que los agentes se "establezcan" y progresen por sí mismos antes de comenzar a tener en cuenta su historial. Esto se hace con las reglas **RR1** y **RR2**, en las que se desactiva el uso de la

memoria durante un tercio del tiempo total de la computación de la estrategia (en este caso los primeros 5 segundos), y luego ésta se activa. Precisando el funcionamiento de estas reglas, lo que se hace es activar la regla **SM** en los inicios de la búsqueda dejando que los agentes resolvedores se reorganicen en aquellas zonas del espacio de búsqueda potencialmente promisorias, y luego del tiempo fijado activar la memoria a través de las reglas **MR1** o **MR2** obligando a los agentes a ir moviéndose a las regiones donde se van obteniendo los mejores resultados. Se intenta lograr con esto que la exploración alcance un peso mayor respecto a la explotación durante los primeros momentos de la búsqueda.

En la Tabla 1 se ve que las reglas que aplican cierto retraso en el uso de la memoria siempre mejoran los promedios de Error respecto a las demás, en particular la regla **RR2** que obtiene los mejores resultados promedios. Los resultados también muestran que la información almacenada en la memoria es mucho más útil al combinar varias reglas. Se ve que siempre las reglas **MR2** y **RR2** se comportan mejor que la reglas **MR1** y **RR1** respectivamente. La regla **MR2** combina aspectos que son independientes de los agentes, como lo hace la regla **MR1**, con aspectos que son intrínsecos al agente como es actuar sobre algún parámetro que pueda cambiar dinámicamente su comportamiento de búsqueda, y mejor aún, como en la regla **RR2**, si a esto se le añade un retraso en la activación del mecanismo de la memoria.

Esto indica que las estrategias cooperativas junto al uso de memoria y reglas difusas tienen sentido al intentar resolverse instancias difíciles, sobre todo cuando estas reglas se aplican dejando libre al agente resolvedor durante un cierto tiempo para que realice la búsqueda sin tener en cuenta su historial precedente.

### 3.3. Problema de la p-mediana: Formulación del problema

Un problema clásico de localización es el problema de la p-mediana, *PM*, donde dado un número total de instalaciones o medianas que van a ser abiertas, hay que decidir en qué puntos éstas deben ser abiertas y asignar cada punto de demanda a una mediana abierta, de forma tal que la distancia total que se atraviesa para cubrir toda la demanda sea mínima.

Para ello, considere un conjunto $L$ de $m$ localizaciones potenciales para $p$ instalaciones o medianas y un conjunto $U$ de localizaciones para $n$ usuarios. Considere una matriz $D$, de tamaño $n \times m$, que contiene las distancias o costos para satisfacer la demanda desde la medianas $j$ hasta el usuario localizado en $i$, para todos los $j \in L$ e $i \in U$.

El problema de la p-mediana intenta localizar simultáneamente las $p$ medianas a fin de minimizar el costo total para satisfacer las demandas de los usuarios, estando cada uno de ellos suministrado por la mediana abierta mas cercana. Matemáticamente el objetivo es seleccionar $p$ columnas de $D$, tal que la suma de los coeficientes mínimos en cada línea dentro de esas columnas sea la menor posible.

$$\text{mín} \sum_{i \in U} \underset{j \in J}{\text{mín}}\, d_{ij}$$

donde, $J \subseteq L$ y $|J| = p$,

### 3.4. Problema de la p-mediana: Experimentos

En el presente trabajo para comprobar el desempeño de la estrategia ante el problema de la p-mediana se usarán las instancias de 1400 nodos tomadas de la biblioteca TSPLIB [12].

El tiempo de cálculo asignado para la búsqueda, $t_{max}$, es igual a 240 segundos. Se hacen 10 repeticiones para cada regla y mediana. El rendimiento de la estrategia se mide en términos de la calidad de la solución y se calcula por el error con respecto a la mejor solución conocida.

$$Error = 100 * \frac{MejorConocido - ValorObtenido}{MejorConocido}$$

La Tabla 2 muestra los mejores resultados obtenidos para diferentes medianas (p), y la Tabla 3 presenta el número de casos con $error \leq 1$, donde los paréntesis indican el número de medianas con este error. Las columnas muestran los resultados de los diferentes esquemas cooperativos modelados en la Base de Reglas del *Coordinador*.

**Cuadro 2.** Valores Mínimos

| **p** | SM | MR1 | RR1 | MR2 | RR2 |
|---|---|---|---|---|---|
| 10 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| 20 | 0,23 | 0,03 | 0,08 | 0,00 | 0,08 |
| 30 | 0,54 | 0,33 | 0,45 | 0,53 | 0,67 |
| 40 | 0,71 | 0,55 | 0,30 | 0,56 | 0,30 |
| 50 | 0,84 | 0,70 | 0,69 | 0,89 | 0,81 |
| 60 | 1,90 | 0,68 | 1,16 | 1,03 | 1,27 |
| 70 | 1,49 | 1,18 | 1,46 | 0,68 | 1,11 |
| 80 | 1,87 | 1,08 | 1,97 | 1,43 | 1,21 |
| 90 | 1,55 | 1,04 | 0,80 | 0,93 | 0,92 |
| 100 | 1,84 | 0,97 | 1,78 | 1,07 | 1,09 |

**Cuadro 3.** Número de Casos con $error \leq 1$

| **p** | SM | MR1 | RR1 | MR2 | RR2 |
|---|---|---|---|---|---|
| 10 | 8 | 8 | 7 | 7 | 9 |
| 20 | 6 | 6 | 7 | 10 | 9 |
| 30 | 6 | 4 | 7 | 5 | 6 |
| 40 | 1 | 4 | 4 | 6 | 7 |
| 50 | 2 | 1 | 1 | 1 | 2 |
| 60 | 0 | 1 | 0 | 0 | 0 |
| 70 | 0 | 0 | 0 | 1 | 0 |
| 80 | 0 | 0 | 0 | 0 | 0 |
| 90 | 0 | 0 | 1 | 1 | 1 |
| 100 | 0 | 1 | 0 | 0 | 0 |
| Total | 23(5) | 25(7) | 27(6) | 31(7) | 34(7) |

El primer análisis se hace para comparar los resultados logrados por la columna **SM** vs. las columnas **MR1** y **MR2**, o sea entre un esquema que no usa la memoria y otros dos que si la usan. Se nota que los esquemas que usan la

memoria siempre mejoran los resultados obtenidos por el esquema que no hacen uso de la memoria y obtienen un mayor número de casos con $error \leq 1$. Se puede concluir que el uso de la memoria contribuye a encontrar soluciones de más calidad y mayor robustez.

Otro análisis se enfoca sobre los resultados de **MR1** (usa la memoria y es independiente de los agentes) vs. **MR2** (usa la memoria y es dependiente de las características de los agentes). Hay una pequeña diferencia entre los valores obtenidos por **MR1** y **MR2**, aunque **MR2** logra resultados ligeramente superiores y es mas robusto su comportamiento.

Haciendo un análisis sobre el conjunto de todas las reglas, el esquema **RR2** obtiene los mejores resultados y mantiene un comportamiento más robusto. En esta regla al inicio de la búsqueda no se hace uso de la memoria permitiendo que los agentes resolvedores se comporten mas exploradores que explotadores; y luego, se activa la regla **MR2**, o sea el uso de la memoria y se ajusta el comportamiento de los agentes haciendo que se comporten mas explotadores que exploradores. Así, se logra un equilibrio global en la estrategia entre la exploración y la explotación del espacio de búsqueda.

## 4. Conclusiones

Los resultados obtenidos en los experimentos muestran los beneficios de la estrategia de coordinación propuesta, se ve cuan sentido tiene el uso de la memoria combinado con reglas difusas dentro de las estrategias cooperativas para obtener soluciones de alta calidad ante instancias muy difíciles sobre todo si se combinan reglas independientes y dependientes de los agentes y se aplican con cierto retraso de tiempo, ya que se obtiene un mejor equilibrio entre la exploración que se logra en mayor medida al inicio de la ejecución y una mayor explotación que se alcanza luego al encaminar el comportamiento del proceso de optimización de manera más directa hacia el refinamiento de las buenas soluciones que se van obteniendo.

Estos mecanismos de coordinación pueden ser aplicados a otro tipo de algoritmos de optimización; las reglas **SM**, **MR1** y **RR1** son independientes del tipo de agentes, no así las reglas **MR2** y **RR2** que sólo serían posibles en algoritmos en que la variación de alguno de sus parámetros de ajuste permitan un cambio en su comportamiento de búsqueda.

Partiendo de todos estos resultados se concluye que los mecanismos de coordinación modelados gracias a las técnicas de la lógica difusa y basados en el retraso del uso de la memoria mejoran la potencia de búsqueda de la estrategia cooperativa.

## Agradecimientos

Este trabajo ha sido financiado parcialmente por el Ministerio de Ciencia y Tecnología bajo los Proyectos TIC-2002-04242-C03-02 y SINTA-CC-TIN 2004-01411.

## Referencias


1. E. Alba, editor. *Parallel Metaheuristics: A New Class of Algorithms*. Wiley Publisher, 2005.
2. T. Crainic and M. Gendreau. Cooperative parallel tabu search for capacitated network design. *Journal of Heuristics*, 8:601–627, 2002.
3. T. Crainic, M. Gendreau, P. Hansen, and N. Mladenovic. Cooperative parallel variable neighborhood search for the p-median. *Journal of Heuristics*, 10:293–314, 2004.
4. T. Crainic and M. Toulouse. *Parallel Strategies for Metaheuristics*, volume Handbook of Metaheuristics, pages 475 – 513. Kluwer Academic Publisher, 2003.
5. V. Cung, S. Martins, C. Ribeiro, and C. Roucairol. *Essays and Surveys in Metaheuristics*, chapter Strategies for the Parallel Implementation of Metaheuristics, pages 263–308. Kluwer Academic Publisher, 2001.
6. H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems*. Springer Verlag, 2004.
7. D. Pelta, A. Blanco, and J. Verdegay. A fuzzy valuation-based local search framework for combinatorial problems. *Journal of Fuzzy Optimization and Decision Making*, 1(2):177–193, 2002.
8. D. Pelta, A. Blanco, and J. Verdegay. *Fuzzy Sets based Heuristics for Optimization*, chapter Fuzzy Adaptive Neighborhood Search: Examples of Application. Studies in Fuzziness and Soft Computing. Physica-Verlag, 2003.
9. D. Pelta, C. Cruz, A. Sancho-Royo, and J. Verdegay. Analyzing the behaviour of a multi-agents based cooperative, parallel strategy for combinatorial optimization. In *Proceedings of the 5th International Conference in Recent Advances in Soft Computing, RASC-04, Nottingham*, 2004.
10. D. Pelta, C. Cruz, A. Sancho-Royo, and J. Verdegay. *Fuzzy Set Techniques in Industrial Engineering. In Press*, chapter Fuzzy Sets based Cooperative Heuristics for Solving Optimization Problems. Springer, 2005.
11. D. Pelta, C. Cruz, A. Sancho-Royo, and J. Verdegay. Using memory and fuzzy rules in a co-operative multi-thread strategy for optimization. *Information Sciences*, 2006. In press.
12. G. Reinelt. Tsplib: A traveling salesman problem library. *ORSA Journal on Computing*, 3:376–384, 1991.
13. P. P. S. Duni Ekisoglu and M. Resende. *Models for Parallel and Distributed Computation - Theory, Algorithmic Techniques and Applications*, chapter Parallel metaheuristics for combinatorial optimization. Kluwer Academic, 2002.

# Retículos de conceptos multi-adjuntos

Jesús Medina, Manuel Ojeda Aciego y Jorge Ruiz Calviño

Dept. Matemática Aplicada
Univ. de Málaga
{jmedina,aciego,jorgerucal}@ctima.uma.es

**Resumen** Recientemente se han propuesto los retículos de conceptos generalizados como una nueva estructura general para manejar información con incertidumbre o incompleta en el marco de la teoría del análisis de datos (o conceptos formales).
En este artículo se presentan los retículos de conceptos multi-adjuntos que enriquecen el lenguaje permitiendo un mayor número de posibilidades al usuario que los retículos de conceptos generalizados y como consecuencia embeben también a otros enfoques más particulares. También se introduce el teorema fundamental de este nuevo marco.
*Palabras clave:* Conexión de Galois, retículos de conceptos, retículos multi-adjuntos

## 1. Introducción

En los últimos años se ha producido una mayor necesidad de desarrollar métodos de razonamiento que contemplen la posibilidad de incertidumbre, datos imprecisos o información incompleta. Por lo que se han presentado distintas propuestas que permiten explicar mejor los hechos observados, la especificación de afirmaciones, el razonamiento y/o la ejecución de programas bajo algún tipo de incertidumbre, independientemente de su procedencia.

Por una parte, se encuentra el marco multi-adjunto [15], generalización de varias programaciones lógicas no clásicas como la residuada, la difusa, la probabilista, posibilista, con anotaciones, etc. La estructura semántica de este marco general es la de retículo multi-adjunto, en el cual se considera un retículo junto con varios conjuntores e implicaciones formando pares adjuntos.

Por otra parte, se encuentran distintos enfoques que generalizan los retículos de conceptos clásicos dados por Ganter y Wille en [9], con la finalidad de permitir algún tipo de incertidumbre en los datos. Uno de estos enfoques es el que presentan Burusco y Fuentes-González en [3] donde introducen los retículos de conceptos difusos, los cuales fueron desarrollados por Pollant en [18] e independientemente considerados y generalizados por Bělohlávek en [1] considerando $L$-igualdades. Este enfoque es mejorado, para la igualdad clásica ($L = \{0, 1\}$), por Krajči que define el retículo de conceptos generalizado en [13,14].

En este artículo se probará que considerar un operador monótono creciente y continuo por la izquierda en un retículo como se define en [13] es equivalente

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 751-762, 2006.

a considerar un operador que conserve supremos de conjuntos cualesquiera y como consecuencia, este tipo de operadores posee un operador adjunto (residuo). Gracias a esto se podrá dar una generalización de los retículos multi-adjuntos que embeberá a este último enfoque con el fin de introducir una estructura en la que las propiedades exigidas a los conjuntores sean mínimas, y de esta forma poder utilizar aproximaciones de t-normas y/o conjuntores definidos a partir de una base de datos mediante el aprendizaje de una red neuronal, como se hace en [17], así como permitir la posibilidad de utilizar distintos conjuntores e implicaciones para enriquecer el lenguaje y de esta forma el usuario tenga un abanico más amplio y flexible de posibilidades.

Por ejemplo, si en una base de datos en la que se recoge la relación entre los días de la semana (objetos) y si son calurosos, fríos, con poca lluvia, sin mucho viento, etc. (atributos); se quiere saber cuál es el mejor día para pasear, buscaremos un día que no sea muy caluroso ni frío y que no llueva. Además, por la posibilidad de considerar distintas implicaciones, se podrá hacer distinción entre los objetos y atributos, y por tanto se podrían considerar días preferibles para pasear.

Además se recuerda un resultado introducido en [8] y que muestra que toda conexión de Galois $(^{\uparrow}, ^{\downarrow})$ tiene un retículo completo asociado definido de la forma $\mathcal{C} = \{\langle x, y\rangle \mid x^{\uparrow} = y, x = y^{\downarrow}; x \in L_1, y \in L_2\}$. Por lo tanto, a partir de este resultado los conjuntos de conceptos definidos en los enfoques dados en [1,3,14] son retículos completos directamente sin considerar la definición particular de la conexión de Galois.

El esquema del artículo es el siguiente: En la Sección 2 se presenta un resumen de los retículos de conceptos generalizados [13] y una introducción a los retículos de Galois; en la Sección 3 se introduce la estructura general de nuestro marco de trabajo, el marco multi-adjunto con tipos, y se estudia la relación entre los pares adjuntos y la noción de continuidad por la izquierda considerada en [13] para continuar, en la Sección 4, con la presentación de los retículos de conceptos multi-adjuntos y el Teorema Fundamental. En la Sección 5 se presenta un pequeño ejemplo ilustrativo, terminando con las conclusiones y dando algunas vías de trabajo futuro.

## 2. Definiciones preliminares

A lo largo de esta sección se introducirán algunas de las nociones y herramientas necesarias para el desarrollo de este artículo. En un principio se comenzará recordado los retículos de conceptos generales presentados en [13] y a continuación con una introducción a las conexiones de Galois.

### 2.1. Retículo de conceptos generalizado

Primeramente se observa que en [13] se introduce un concepto de continuidad para operadores definidos sobre retículos, llamada continuidad por la izquierda:

**Definition 1.** *Dado un retículo completo* $(L, \preceq_L)$ *y un conjunto parcialmente ordenado* $(P, \leq_P)$*, decimos que* $T\colon L \to P$ *es* continua por la izquierda *si cumple la siguiente propiedad: Dado* $p \in P$ *y* $X \subseteq L$*, si* $T(x) \leq_P p$ *para todo* $x \in X$*, entonces* $T(\sup X) \leq_P p$*.*

Ésta es utilizada para generalizar los retículos de conceptos difusos. Además, se consideran dos conjuntos no vacíos $A$ y $B$, que representan al conjunto de atributos y al de objetos, respectivamente; y una relación difusa entre ambos conjuntos, $R$, evaluada en un conjunto parcialmente ordenado $(P, \leq)$, esto es, $R\colon A \times B \longrightarrow P$.

Ahora, se consideran dos retículos completos[1] $(L_1, \preceq_1)$ $(L_2, \preceq_2)$ y una aplicación $(\cdot)\colon L_1 \times L_2 \longrightarrow P$ creciente y continua por la izquierda en ambos argumentos.

La tupla $(A, B, R, \cdot)$ es el contexto donde se definen los conceptos considerando aplicaciones $^{\uparrow}\colon {L_2}^B \longrightarrow {L_1}^A$ y $^{\downarrow}\colon {L_1}^A \longrightarrow {L_2}^B$ como siguen (donde $X^Y$ denota el conjunto de aplicaciones de $Y$ a $X$):

$$g^{\uparrow}(a) = \sup\{x \in L_1 \mid (\forall b \in B) x \cdot g(b) \preceq R(a,b)\}$$
$$f^{\downarrow}(b) = \sup\{y \in L_2 \mid (\forall a \in A) f(a) \cdot y \preceq R(a,b)\}$$

Krajči probó en [14] que el conjunto

$$\mathcal{C} = \{\langle g, f\rangle \mid g \in {L_2}^B, f \in {L_1}^A \text{ y } g^{\uparrow} = f, f^{\downarrow} = g\}$$

con el siguiente orden

$$\langle g_1, f_1\rangle \preceq \langle g_2, f_2\rangle \quad \text{si y sólo si} \quad g_1 \preceq g_2 \quad \text{si y sólo si} \quad f_2 \preceq f_1$$

es un retículo completo y define lo que se conoce como *retículo de conceptos generalizado* en $(A, B, R, \cdot)$, donde un *concepto* en $(A, B, R, \cdot)$ es cualquier elemento $\langle g, f\rangle \in \mathcal{C}$.

### 2.2. Conexiones de Galois y retículos de conceptos

En esta sección se recordará la definición de conexión de Galois y un resultado que muestra que cada conexión de Galois $(^{\uparrow}, ^{\downarrow})$ define un retículo completo, llamado *retículo de Galois* o *retículo de conceptos*.

**Definition 2.** *Consideremos dos conjuntos parcialmente ordenados* $(P_1, \leq_1)$ *y* $(P_2, \leq_2)$*, y aplicaciones* $^{\downarrow}\colon P_1 \to P_2$*,* $^{\uparrow}\colon P_2 \to P_1$*, decimos que* $(^{\uparrow}, ^{\downarrow})$ *forman una* conexión de Galois *entre* $P_1$ *y* $P_2$ *si y sólo si:*

1. $^{\downarrow}$ *y* $^{\uparrow}$ *son decrecientes.*
2. $p_1 \leq_1 {p_1}^{\downarrow\uparrow}$ *para todo* $p_1 \in P_1$*.*
3. $p_2 \leq_2 {p_2}^{\uparrow\downarrow}$ *para todo* $p_2 \in P_2$*.*

[1] Equivalentemente, se requiere en [14] semirretículos completos superiormente.

De la definición se obtiene el siguiente resultado:

**Corollary 1.** *Si* $(^{\uparrow}, ^{\downarrow})$ *es un conexión de Galois entre* $P_1$ *y* $P_2$ *entonces tenemos que para todo* $p_1 \in P_1$ *y* $p_2 \in P_2$*:*

- $p_1{}^{\downarrow\uparrow\downarrow} = p_1{}^{\downarrow}$
- $p_2{}^{\uparrow\downarrow\uparrow} = p_2{}^{\uparrow}$
- $p_2 \leq_2 p_1{}^{\downarrow}$ *si y sólo si* $p_1 \leq_1 p_2{}^{\uparrow}$

Si $P_1$ y $P_2$ son retículos completos podemos usar los resultados de la Sección 7 de [8] para establecer el siguiente teorema:

**Theorem 1.** *Sean* $(L_1, \preceq_1)$, $(L_2, \preceq_2)$ *retículos completos,* $(^{\uparrow}, ^{\downarrow})$ *una conexión de Galois entre* $L_1, L_2$ *y* $\mathcal{C} = \{\langle x, y\rangle \mid x^{\uparrow} = y, x = y^{\downarrow}; x \in L_1, y \in L_2\}$ *entonces* $\mathcal{C}$ *es un retículo completo, donde*

$$\bigwedge_{i\in I}\langle x_i, y_i\rangle = \langle \bigwedge_{i\in I} x_i, (\bigvee_{i\in I} y_i)^{\downarrow\uparrow}\rangle \quad y \quad \bigvee_{i\in I}\langle x_i, y_i\rangle = \langle (\bigvee_{i\in I} x_i)^{\uparrow\downarrow}, \bigwedge_{i\in I} y_i\rangle$$

Este teorema es interesante porque se puede aplicar en numerosas teorías de retículos de conceptos como en [1,3,9,14], independientemente de la definición de las conexiones de Galois.

## 3. Marco multi-adjunto con tipos y propiedades

Se quiere introducir una estructura más general que los retículos multi-adjuntos [15] usando tipos de forma similar a [5] ya que se pretende absorber la teoría difusa de retículos de conceptos generalizados. Para comenzar, se generalizan sus bloques básicos, los pares adjuntos:

**Definition 3.** *Sean* $(P_1, \leq_1)$, $(P_2, \leq_2)$, $(P_3, \leq_3)$ *conjuntos parcialmente ordenados y* $\&\colon P_1 \times P_2 \longrightarrow P_3$, $\leftarrow\colon P_3 \times P_2 \longrightarrow P_1$ *aplicaciones, decimos que* $(\&, \leftarrow)$ *es un* par adjunto con tipos *con respecto a* $P_1, P_2, P_3$ *si:*

- $\&$ *es creciente en ambos argumentos.*
- $\leftarrow$ *es creciente en el primer argumento y decreciente en el segundo.*
- $x_1 \leq_1 (x_3 \leftarrow x_2)$ *si y sólo si* $(x_1 \,\&\, x_2) \leq_3 x_3$ *donde* $x_i \in P_i$*.*

Esta última propiedad es conocida como la *propiedad de adjunción* y se puede interpretar como una regla tipo *modus ponens* multi-valuada. Desde un punto de vista categórico, esta condición surge al considerar al conjuntor como un bifuntor, y aplicar la propiedad de adjunción. Nótese también que las condiciones de frontera para los conjuntores no son necesarias.

El extender las definiciones dadas en distintos paradigmas de programación lógica [5,6,7,15,20] a un conjunto más general, en el que se utilizan diferentes implicaciones (Łukasiewicz, Gödel, producto, etc), conduce de forma natural a considerar varios *pares adjuntos con tipos* en el retículo. Esta es la intuición que encierra la noción de marco multi-adjunto con tipos, cuya definición formal viene dada por:

**Definition 4.** *Un* marco multi-adjunto con tipos *es un tupla*

$$(P_1, P_2, P_3, \leq_1, \leq_2, \leq_3, \&_1, \leftarrow_1, \ldots, \&_n, \leftarrow_n)$$

*tal que* $(\&_i, \leftarrow_i)$ *es un par adjunto con tipos con respecto a* $P_1, P_2, P_3$ *para todo* $i$.

Pero, ¿cuándo un conjuntor tiene un residuo (adjunto)? En [11] se dice que toda aplicación definida sobre dos retículos completos $L_1$, $L_2$, tiene residuo (el cuál es único) si y sólo si conserva supremos de conjuntos, esto es: la aplicación $T\colon L_1 \to L_2$ satisface $T(\sup(X)) = \sup(T(X))$ para todo subconjunto $X \subseteq L_1$.

Como consecuencia de este resultado, cuando en [12] se demuestra que toda t-norma tiene un único adjunto (residuo) en [0,1] si y sólo si es continua por la izquierda, es porque esta continuidad es equivalente a que conserve supremos de subconjuntos. Por tanto realmente lo interesante del resultado es decir cuál es el residuo de una t-norma.

En nuestro caso también tenemos retículos completos, por lo que se puede aplicar el resultado dado en [11]. Pero si queremos dar un resultado similar al dado en [12] la definición de continuidad para retículos completos más usada es la continuidad de Scott que exige que conserve supremos sólo de subconjuntos dirigidos, lo que hace que esta exigencia no sea suficiente para asegurar que una función continua Scott tenga residuo.

Sin embargo, la continuidad por la izquierda definida en la Subsección 2.1 sí nos aporta una caracterización de la existencia de residuo en términos de continuidad, como se observa a continuación:

**Proposition 1.** *Dados dos retículos completos* $(L_1, \preceq_1)$ *y* $(L_2, \preceq_2)$*, la aplicación* $T\colon L_1 \to L_2$ *es creciente y continua por la izquierda si y sólo si para todo subconjunto* $X \subseteq L_1$ *se tiene que* $T(\sup(X)) = \sup(T(X))$.

*Demostración.* Dado un subconjunto $X \subseteq L_1$, la desigualdad $\sup(T(X)) \preceq_2 T(\sup(X))$ es directa por el crecimiento y la propiedad del supremo.

Como $T(x) \preceq_2 \sup(T(X))$ para todo $x \in X$, por la continuidad por a izquierda de $T$ se tiene la otra desigualdad: $T(\sup(X)) \preceq_2 \sup(T(X))$.

Ahora, dado $y \in L_2$ y un subconjunto $X \subseteq L_1$, si $T(x) \preceq_2 y$ para todo $x \in X$ se tiene que $\sup(T(X)) \preceq_2 y$, por tanto, por hipótesis y la propiedad del supremo se tiene:

$$T(\sup X) = \sup(T(X)) \preceq_2 y$$

por lo que $T$ es continua por la izquierda. Para demostrar la monotonía, se considera el conjunto $X = \{x_1, x_2\}$ con $x_1 \preceq_1 x_2$, entonces, por hipótesis:

$$T(x_2) = T(\sup X) = \sup\{T(x_1), T(x_2)\}$$

lo que nos lleva a que $T(x_1) \preceq_2 T(x_2)$. □

Por tanto, se puede demostrar el siguiente resultado extraído de [2] similar al presentado en [12] para el caso lineal de $[0, 1]$.

Pero que se consideran retículos completos y que el conjuntor sólo tiene que cumplir que sea creciente y continuo por la izquierda.

**Corollary 2.** *Dados dos retículos completos* $(L_1, \preceq_1)$, $(L_2, \preceq_2)$, *un conjunto parcialmente ordenado* $(P, \leq)$ *y un conjuntor creciente* $\&\colon L_1 \times L_2 \longrightarrow P$, *se tiene que* $\&$ *es continuo por la izquierda en el primer argumento si y sólo si* $\&$ *y la aplicación definida para cada* $x \in L_1$, $y \in L_2$, $z \in P$ *como*

$$z \leftarrow y = \sup\{x \in L_1 \mid x \,\&\, y \leq z\}$$

*satisfacen la propiedad de adjunción.*

Como consecuencia inmediata del corolario anterior se tiene que los operadores dados en [13,14] tienen implicación adjunta y por tanto los resultados introducidos en estos artículos se pueden escribir en términos de pares adjuntos dentro de un retículo multi-adjunto con tipos. Esto hace que el estudio teórico no sea tan complicado y que el cálculo computacional sea más sencillo al poder utilizar directamente las implicaciones adjuntas.

Además en el marco multi-adjunto se pueden considerar distintos pares adjuntos lo que enriquece el lenguaje y abre un abanico de posibilidades al usuario como se observará más detenidamente en el Ejemplo 5.

## 4. Retículo de conceptos multi-adjunto

Primeramente se analizará una consecuencia del hecho de no exigir la conmutatividad a los conjuntores, que se utilizará para definir las conexiones de Galois en el marco multi-adjunto. Después se definirán las nociones básicas de retículos de conceptos en este marco, finalizando con el teorema fundamental del retículo de conceptos introducido.

Puesto que los conjuntores $\&$ no tienen por qué ser conmutativos, ver [16], se pueden considerar dos operadores adjuntos, $\swarrow$, $\nwarrow$, como se hace para retículos residuados[2] en [10], satisfaciendo las propiedades de adjunción:

$$\begin{array}{lllll} 1. & x \preceq_1 z \swarrow^i y & \text{si y sólo si} & x \,\&_i\, y \preceq_3 z \\ 2. & y \preceq_2 z \nwarrow_i x & \text{si y sólo si} & x \,\&_i\, y \preceq_3 z \end{array}$$

Claramente, desde el Corolario 2, se tiene que los conjuntores $\&$ definidos en [13], por ser continuos por la izquierda, tienen asociados estos operadores, los cuales se definen como:

$$z \swarrow y = \sup\{x \in L_1 \mid x \,\&\, y \leq z\}$$
$$z \nwarrow x = \sup\{y \in L_2 \mid x \,\&\, y \leq z\}$$

Por consiguiente, considerar este tipo de conjuntor, que es creciente y continuo por la izquierda en ambos argumentos, es equivalente a considerar la terna $(\&, \swarrow, \nwarrow)$ satisfaciendo las propiedades de adjunción (1) y (2), y siendo $\&$ creciente en ambos argumentos, que se llamarán *ternas adjuntas*.

Además estos operadores aportan una definición alternativa de las conexiones de Galois definidas en [13] como se observa en el siguiente lema:

[2] Observar que los retículos residuados son un caso particular de los retículos multi-adjuntos.

**Lemma 1.** *Dado un conjuntor creciente* $\&\colon L_1 \times L_2 \to P$ *y los operadores* $\swarrow, \nwarrow$ *satisfaciendo las propiedades de adjunción* (1) *y* (2)*, se tiene que:*

$$\sup\{x \in L_1 \mid (\forall b \in B) x \,\&\, g(b) \preceq_3 R(a,b)\} = \text{ínf}\{R(a,b) \swarrow g(b) \mid b \in B\}$$
$$\sup\{y \in L_2 \mid (\forall a \in A) f(a) \,\&\, y \preceq_3 R(a,b)\} = \text{ínf}\{R(a,b) \nwarrow f(a) \mid a \in A\}$$

*Demostración.* Usando que & y $\swarrow$ satisfacen (1) se tiene que el

$$\sup\{x \in L_1 \mid (\forall b \in B) x \,\&\, g(b) \preceq_3 R(a,b)\}$$

es igual al $\sup\{x \in L_1 \mid (\forall b \in B) x \preceq_1 R(a,b) \swarrow g(b)\}$, que es la caracterización de ínfimo en términos del supremo:

$$\text{ínf}\{R(a,b) \swarrow g(b) \mid b \in B\}$$

La otra igualdad se prueba simétricamente usando que & y $\swarrow$ satisfacen (2). □

Por tanto, dados $A$ y $B$ conjuntos no vacíos y $R\colon A \times B \longrightarrow P$ una relación difusa, se pueden definir las siguientes aplicaciones $^{\uparrow}\colon L_2^B \longrightarrow L_1^A$ y $^{\downarrow}\colon L_1^A \longrightarrow L_2^B$ más generales, por ejemplo, que las dadas en [3,13]:

$$g^{\uparrow}(a) = \text{ínf}\{R(a,b) \swarrow^i g(b) \mid b \in B\}$$
$$f^{\downarrow}(b) = \text{ínf}\{R(a,b) \nwarrow_i f(a) \mid a \in A\}$$

las cuales forman una conexión de Galois como se demuestra a continuación:

**Proposition 2.** *El par* $(^{\uparrow}, ^{\downarrow})$ *es una conexión de Galois entre* $L_1^A$ *y* $L_2^B$*.*

*Demostración.* Tenemos que probar que:

- $^{\uparrow}$ y $^{\downarrow}$ son decrecientes;
- $g \leq g^{\uparrow\downarrow}$ para todo $g \in L_2^B$; y
- $f \leq f^{\downarrow\uparrow}$ para todo $f \in L_1^A$.

El primer punto es trivial ya que las implicaciones son decrecientes en el segundo argumento. Para los demás se considera, sin pérdida de generalidad, que las implicaciones están asociadas a los objetos, entonces se puede escribir:

$$g^{\uparrow}(a) = \text{ínf}\{R(a,b) \swarrow^b g(b) \mid b \in B\} \quad \text{para todo } a \in A$$

Por lo tanto, dado $a \in A$ y $b \in B$ se tiene la siguiente cadena de desigualdades, gracias a la propiedad de adjunción:

$$\begin{aligned} g^{\uparrow}(a) = \text{ínf}\{R(a,b') \swarrow^{b'} g(b') \mid b' \in B\} &\preceq_2 R(a,b) \swarrow^b g(b) \\ g^{\uparrow}(a) \,\&_b\, g(b) &\preceq_2 R(a,b) \\ g(b) &\preceq_2 R(a,b) \nwarrow_b g^{\uparrow}(a) \end{aligned}$$

Como la desigualdad se da para todo $a \in A$, se obtiene, usando la propiedad del ínfimo, que:

$$g(b) \preceq_2 \text{ínf}\{R(a,b) \nwarrow_b g^{\uparrow}(a) \mid a \in A\}$$

La otra desigualdad se obtiene simétricamente. □

Se está ahora en disposición de presentar las nociones básicas de retículos de conceptos en el marco multi-adjunto. Un *contexto multi-adjunto* es una tupla

$$(A, B, R, \&_1, \swarrow^1, \nwarrow_1, \ldots, \&_n, \swarrow^n, \nwarrow_n)$$

satisfaciendo que $(\&_i, \swarrow_i, \nwarrow_i)$ son ternas adjuntas para todo $i \in \{1, \ldots, n\}$. Se entenderá por *concepto* al par $\langle g, f\rangle$ satisfaciendo que $g \in {L_2}^B$, $f \in {L_1}^A$ y que $g^{\uparrow} = f$ y $f^{\downarrow} = g$; siendo $(^{\uparrow}, ^{\downarrow})$ la conexión de Galois definida anteriormente.

Ahora se puede usar el Teorema 1, ya que $L_1^A$ y $L_2^B$ son retículos completos, para obtener la siguiente definición.

**Definition 5.** *Se dice que* $\mathcal{M} = \{\langle g, f\rangle \mid g \in L_2^B, f \in L_1^A \text{ and } g^{\uparrow} = f, f^{\downarrow} = g\}$, *con el siguiente orden:* $\langle g_1, f_1\rangle \preceq \langle g_2, f_2\rangle$ *si y sólo si* $g_1 \preceq g_2$ *(equivalentemente* $f_1 \preceq f_2$*), es un* retículo de conceptos multi-adjunto.

A continuación se presenta el Teorema Fundamental de retículos de conceptos multi-adjunto, que es un resultado en la linea de los dados en las teorías de retículos de conceptos. Para ello se necesita introducir primeramente la siguiente definición.

**Definition 6.** *Dados* $(L, \preceq)$ *y* $(V, \leq_V)$ *dos retículos completos, una aplicación* $f\colon L \to V$ *es densa inferiormente (densa superiormente) si y sólo si para todo* $v \in V$ *existe* $K' \subseteq f[K]$ *tal que* $v = \text{ínf}(K')$ *(*$v = \sup(K')$*).*

**Theorem 2.** *Sean* $(L_1, \preceq_1)$, $(L_2, \preceq_2)$ *retículos completos y* $(^{\uparrow}, ^{\downarrow})$ *una conexión de Galois entre* $L_1$ *y* $L_2$, *y* $(\mathcal{M}, \preceq)$ *el retículo de conceptos generalizado con respecto a* $(^{\uparrow}, ^{\downarrow})$ *entonces:*

*Si* $\mathcal{V} = (V, \leq_V)$ *es un retículo completo, entonces* $\mathcal{V}$ *es isomorfo a* $\mathcal{M}$ *si y sólo si existen aplicaciones* $\alpha\colon A \times L_1 \to V$, $\beta\colon B \times L_2 \to V$ *tales que:*

*1a)* $\alpha$ *es decreciente en el segundo argumento.*
*1b)* $\beta$ *es creciente en el segundo argumento.*
*2a)* $\alpha[A \times L_1]$ *es densa inferiormente.*
*2b)* $\beta[B \times L_2]$ *es densa superiormente.*
*3) Para cada* $a \in A$, $b \in B$, $x \in L_1$, $y \in L_2$*:*

$$\beta(b, y) \leq_V \alpha(a, x) \text{ si y sólo si } x \,\&_i\, y \preceq_3 R(a, b)$$

La demostración de este teorema se ha omitido por motivos de espacio. Aunque el hecho de usar la propiedad de adjunción la simplifica. Además, el uso de los operadores implicación adjuntos nos lleva a que el cálculo computacional de los conceptos en un contexto sea más sencillo.

## 5. Un ejemplo ilustrativo

Ahora se aplicará las posibilidades del lenguaje de este marco a un ejemplo introducido por Umbreit [19] y usado por [4]. También se comprobará que el marco multi-adjunto puede expresar mejor que otros enfoques las necesidades del usuario.

*Example 1.* El contexto es el siguiente: Se considera el caso lineal $L_1 = L_2 = L_3 = [0,1]$, como conjuntos de atributos y objetos los siguientes:

$$A = \{\text{calor}, \text{frío}, \text{poca lluvia}, \text{poco viento}\}$$
$$B = \{\text{Lun}, \text{Mar}, \text{Miér}, \text{Juev}, \text{Vier}, \text{Sáb}, \text{Dom}\}$$

y como relación $R\colon A \times B \to L_1$ la definida en la siguiente tabla.

| $R$ | calor | frío | poca lluvia | poco viento |
|---|---|---|---|---|
| Lun | 0,5 | 0,5 | 1 | 1 |
| Mar | 1 | 0 | 1 | 1 |
| Miér | 0,5 | 0,5 | 0 | 0 |
| Juev | 0,5 | 0,5 | 1 | 0 |
| Vier | 0 | 1 | 0 | 0 |
| Sáb | 0 | 1 | 0,5 | 0 |
| Dom | 0 | 1 | 1 | 1 |

Ahora se analizará el problema de encontrar el mejor día para *ir de paseo* que podría considerarse como un día de la semana en el que no haga ni frío ni calor y sin lluvia, así esta noción difusa se puede expresar mediante el subconjunto difuso $f\colon A \to [0,1]$ definido como:

$$f(\text{calor}) = 0{,}5,\ f(\text{frío}) = 0{,}5,\ f(\text{poca lluvia}) = 1,\ f(\text{poco viento}) = 0{,}5$$

y representado como: $f = \{\text{calor}/0{,}5, \text{frío}/0{,}5, \text{poca lluvia}/1, \text{poco viento}/0{,}5\}$. Se desea obtener un concepto multi-adjunto que represente la situación dada por $f$. Para ello podemos usar diferentes implicaciones, por ejemplo, la de Gödel o la de Łukasiewicz:[3]

$$b \leftarrow_G a = \begin{cases} 1, & \text{si } \ b \geq a; \\ b, & \text{en otro caso.} \end{cases} \qquad b \leftarrow_L a = \text{mín}\{1, 1 + b - a\}$$

Para empezar se comprueba que usando estas implicaciones como se hace en [4] se obtienen los mismos conceptos, esto es, si se considera la implicación de Gödel se obtiene que:

$$\begin{aligned} f^{\downarrow}(\text{Lun}) &= \text{ínf}\{R(a, \text{Lun}) \nwarrow_G f(a)\colon a \in A\} \\ &= \text{ínf}\{0{,}5 \nwarrow_G 0{,}5, 0{,}5 \nwarrow_G 0{,}5, 1 \nwarrow_G 1, 1 \nwarrow_G 0{,}5\} = 1 \end{aligned}$$

Si se hace lo mismo para los demás días se obtiene el valor 0 para todos. Una vez obtenidos los valores de $f^{\downarrow} = g$, se calculan los de $g^{\uparrow}$:

$$\begin{aligned} &g^{\uparrow}(\text{calor}) = \\ &= \text{ínf}\{R(\text{calor}, b) \swarrow^G g(b)\colon b \in B\} \\ &= \text{ínf}\{0{,}5 \swarrow^G 1, 1 \swarrow^G 0, 0{,}5 \swarrow^G 0, 0{,}5 \swarrow^G 0, 0 \swarrow^G 0\, 0 \swarrow^G 0, 0 \swarrow^G 0\} \\ &= 0{,}5 \end{aligned}$$

[3] Obsérvese que los conjuntores adjuntos a estas implicaciones son conmutativos entonces $\leftarrow_G = \swarrow^G = \nwarrow_G$ y $\leftarrow_L = \swarrow^L = \nwarrow_L$.

Si se hace lo mismo para las demás características se obtiene que $g^{\uparrow}(\text{frío}) = 0{,}5$ y que $g^{\uparrow}(\text{poca lluvia}) = g^{\uparrow}(\text{poco viento}) = 1$. Así, el mejor día para ir de paseo (según la definición considerada) es el lunes mientras que los demás no son propicios.

Si se usa usamos la implicación de Łukasiewicz se obtiene que las aplicaciones $f^{\downarrow}$, $g^{\uparrow}$ vienen definidas como:

$$f^{\downarrow} = \{\text{Lun}/1, \text{Mar}/0{,}5, \text{Miér}/0, \text{Juev}/0{,}5, \text{Vier}/0, \text{Sáb}/0{,}5, \text{Dom}/0{,}5\}$$
$$g^{\uparrow} = \{\text{calor}/0{,}5, \text{frío}/0{,}5, \text{poca lluvia}/1, \text{poco viento}/0{,}5\}$$

En este caso, el mejor día es el lunes, sin embargo martes, jueves, sábado y domingo son otras posibilidades mientras que miércoles y viernes no son buenos días.

Así, como ya se había comentado, los conceptos obtenidos $\langle f^{\downarrow}, f^{\downarrow\uparrow}\rangle$ son los mismos que en [4]. Sin embargo, en un contexto multi-adjunto se puede considerar que para *ir de paseo* es además preferible que sea fin de semana. Esta nueva consideración puede ser abordada tomando la implicación de Gödel para los días laborables y la de Łukasiewicz para el sábado y domingo. Obteniendo para este caso los siguientes resultados:

Sea $B_1 = \{\text{Lun}, \text{Mar}, \text{Miér}, \text{Juev}, \text{Vier}\}$ y $B_2 = \{\text{Sáb}, \text{Dom}\}$, entonces

$$f^{\downarrow}(b_1) = \text{ínf}\{R(a, b_1) \nwarrow_G f(a) \colon a \in A\} \quad \text{para} \;\; b_1 \in B_1$$
$$f^{\downarrow}(b_2) = \text{ínf}\{R(a, b_2) \nwarrow_L f(a) \colon a \in A\} \quad \text{para} \;\; b_2 \in B_2$$

por lo que se tiene que

$$f^{\downarrow} = \{\text{Lun}/1, \text{Mar}/0, \text{Miér}/0, \text{Juev}/0, \text{Vier}/0, \text{Sáb}/0{,}5, \text{Dom}/0{,}5\}$$

Para $g^{\uparrow} = f^{\downarrow\uparrow}$ se realizaría también teniendo en cuenta la relación entre los objetos e implicaciones:

$$\begin{aligned} &g^{\uparrow}(\text{calor}) = \\ &= \text{ínf}(\{R(\text{calor}, b_1) \swarrow^G g(b_1) \colon b_1 \in B_1\} \cup \{R(\text{calor}, b_2) \swarrow^L g(b_2) \colon b_2 \in B_2\} \\ &= 0{,}5 \end{aligned}$$

Si se hace lo mismo para las demás características se obtiene:

$$g^{\uparrow} = \{\text{calor}/0{,}5, \text{frío}/0{,}5, \text{poca lluvia}/1, \text{poco viento}/0{,}5\}$$

Ahora, aunque el usuario prefiere los fines de semana para dar un paseo, el lunes sigue siendo el mejor día, pero ahora el sábado y el domingo son mejores que los demás. Recordar que también se pueden dar nociones difusas relacionadas con los atributos, por ejemplo se puede estudiar *el tiempo en el fin de semana*, representado claramente por el subconjunto difuso:

$$g = \{\text{Lun}/0, \text{Mar}/0, \text{Miér}/0, \text{Juev}/0, \text{Vier}/0, \text{Sáb}/1, \text{Dom}/1\}$$

y fijarse con más interés en los atributos 'calor' y 'poca lluvia', considerando para estos distintas implicaciones.

## 6. Conclusiones y trabajo futuro

Se ha utilizado la relación dada en [8] entre conexiones de Galois y retículos de conceptos para probar directamente que los conjuntos de conceptos de [1,3,14] forman un retículo completo sin necesidad de conocer las definiciones de las conexiones de Galois.

También se ha probado la equivalencia entre la existencia de la implicación adjunta de un operador creciente y la continuidad por la izquierda definida en [14], continuidad más general que la continuidad Scott.

Este hecho se ha utilizado para introducir una nueva estructura, el retículo de conceptos multi-adjunto, que embebe al retículo de conceptos generalizado [13] y, como consecuencia, a diferentes extensiones difusas del retículo de conceptos clásico [9], como el de conceptos difusos en [3] y en [1] para el caso de la $\{0,1\}$-igualdad. Una de las características más interesantes es que en el contexto multi-adjunto cada objeto o atributo tiene asociada una implicación por lo que se pueden establecer subgrupos con distintos grados de preferencia.

Continuando con la comparación del marco multi-adjunto con otros enfoques difusos, se pretende estudiar la relación con los retículos de conceptos dados en [10]. Otro punto a tener en cuenta sería la introducción de $L$-igualdades con el fin de embeber completamente el retículo de conceptos difuso de [1].

## Agradecimientos

Partially supported by Spanish DGI project TIC2003-09001-C02-01.

## Referencias

1. R. Bělohlávek. Concept lattices and order in fuzzy logic. *Annals of Pure and Applied Logic*, 128:277–298, 2004
2. R. Bělohlávek y Vilém Vychodil. What is a fuzzy concept lattice? *Concept Lattices and Their Applications*, 34–45, 2005
3. A. Burusco y R. Fuentes-González. The study of the L-fuzzy concept lattice. *Mathware & Soft Computing*, 3:209–218, 1994
4. A. Burusco y R. Fuentes-González. Concept lattices defined from implication operators. *Fuzzy Sets and Systems*, 114:431–436, 2000
5. C.V. Damásio, J. Medina y M. Ojeda-Aciego. Sorted Multi-adjoint Logic Programs: Termination Results and Aplications. *Lect. Notes in Artificial Intelligence*, 3229:252–265, 2004.
6. C. V. Damásio y L. M. Pereira. Monotonic and residuated logic programs. *Lect. Notes in Artificial Intelligence*, 2143:748–759, 2001.
7. C. V. Damásio y L. M. Pereira. Hybrid probabilistic logic programs as residuated logic programs. *Studia Logica*, 72(1):113–138, 2002.
8. B.A. Davey y H.A. Priestley. Introduction to Lattices and Order. Second edition. *Cambridge University Press*, 2002.
9. B. Ganter y R. Wille. Formal Concept Analysis. *Mathematical Foundation*, Springer Verlag, 1999.

10. G. Georgescu y A. Popescu. Concept lattices and similarity in non-commutative fuzzy logic. *Fundamenta Informaticae*, 55(1):23–54, 2002.
11. G. Grätzer. General Lattice Theory. *Birkhäuser Verlag*, 1998.
12. P. Hájek. Metamathematics of Fuzzy Logic. *Kluwer Academic*, Trends in Logic, 1998.
13. S. Krajči. A generalized concept lattice. *Logic Journal of IGPL*, 13(5):543-550, 2005.
14. S. Krajči. The basic theorem on generalized concept lattice. Eds. V. Snášel , R. Bělohlávek, *Concept Lattices and Their Applications Workshop*, 25–33, 2004.
15. J. Medina, M. Ojeda-Aciego y P. Vojtáš. Multi-adjoint logic programming with continuous semantics. *Lect. Notes in Artificial Intelligence*, 2173:351–364, 2001.
16. J. Medina, M. Ojeda Aciego, A. Valverde y P. Vojtáš. Towards Biresiduated Multi-Adjoint Logic Programming. *Lect. Notes in Artificial Intelligence*, 3040:608–617, 2004.
17. E. Naito, J. Ozawa, I. Hayashi y N. Wakami. A proposal of a fuzzy connective with learning function. *Fuzziness Database Management Systems*, pages 345–364. Physica Verlag, 1995.
18. S. Pollandt. Fuzzy Begriffe. *Springer*, Berlin, 1997.
19. S. Umbreit. Formale Begriffsanalyse mit unscharfen Begriffen *Ph.D. Thesis, Halle, Saale*, 1995.
20. P. Vojtáš. Fuzzy logic programming. *Fuzzy Sets and Systems*, 124(3):361–370, 2001.

# Descripción lingüística de trayectorias de objetos obtenidas directamente de vídeo MPEG

Luis Rodríguez Benítez[1,4], Juan Moreno García[2,4], José Castro-Schez[3,4] y Luis Jiménez[3,4]

[1] Escuela Universitaria Politécnica de Almadén, Plaza Manuel Meca s/n, 13400, Almadén. luis.rodriguez@uclm.es
[2] Escuela de Ingeniería Técnica Industrial. Avda. Carlos III s/n. Toledo juan.moreno@uclm.es
[3] Escuela Superior de Informática. Paseo de la Universidad s/n. 13071, Ciudad Real {josejesus.castro,luis.jimenez}@uclm.es
[4] Departamento de Tecnologías y Sistemas de Información. Universidad de Castilla-La Mancha

**Resumen** Un problema de alto nivel y complejidad en un sistema de visión por computador es la descripción automática de escenas. Con información obtenida directamente de la señal comprimida de vídeo MPEG, el sistema presentado en este artículo aisla los objetos con movimiento en cada uno de los frames de una secuencia de vídeo basándose en el desplazamiento uniforme de los mismos. La información sobre el desplazamiento se extrae directamente de los vectores de movimiento MPEG. Además, utilizando lógica difusa, genera para cada frame una descripción semántica de la posición, dirección y velocidad de todos los objetivos detectados en el mismo. Por último, con todos estos datos, representa lingüísticamente la trayectoria seguida por cada objeto individual a lo largo de toda la secuencia de vídeo.

## 1. Vectores de movimiento MPEG

El macrobloque es la unidad básica en el stream MPEG y es un área de 16 por 16 píxeles donde se almacenan, además de otra información, los vectores de movimiento.

En una secuencia de vídeo normalmente sólo hay pequeños movimientos entre frame y frame. Por esta razón, los macrobloques se comparan entre frames y en vez de tener que codificar de nuevo todo el macrobloque, basta con codificar el desplazamiento del macrobloque de un frame a otro. Este desplazamiento se codifica en forma de vector, como puede observarse en la figura 1.

## 2. Elementos lingüísticos del sistema

### 2.1. Vectores de movimiento lingüísticos

Los Vectores de Movimiento Lingüísticos fueron definidos por primera vez en [6]. Esta definición se amplía para permitir al sistema la localización de objetivos múltiples.

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 763-768, 2006.

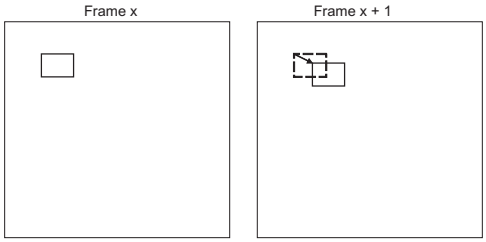



**Figura 1.** Vector de movimiento asociado a un macrobloque

**Definition 1.** *Un **vector de movimiento lingüístico** es una quíntupla*

$$LMV =< NumeroFrame, FP_r, FP_d, FP_v, FP_h > \quad (1)$$

*donde el primer elemento indica el número del frame en el que se localiza el vector de movimiento MPEG y los cuatro elementos restantes son una particularización de los intervalos lingüísticos vistos en [6], con longitud máxima dos.*

Cada $FP_j$ en el soporte de una variable de entrada $X_j$ se representa como:

$$[SA_j^m, \mu_{SA_j^m}(x_j), SA_j^{m+1}, \mu_{SA_j^{m+1}}(x_j)] \quad (2)$$

si el intervalo tiene longitud dos y si existe una única etiqueta cuyo valor de pertenencia es igual a 1 se podrá representar como:

$$[SA_j^m, 1] \quad (3)$$

Todas las etiquetas lingüísticas pertenecen a una variable continua $SA_j$. $FP_r$ y $FP_d$ representan las dos componentes de la velocidad, derecha_x y abajo_x, del vector de movimiento. $FP_v$ representa la posición vertical y $FP_h$ la posición horizontal del macrobloque que tiene asociado el vector de movimiento. Los macrobloques se identifican por un número desde el cero a un valor dado *n-1*, donde *n* representa el número total de macrobloques por frame. Los conjuntos de etiquetas lingüísticas utilizados, junto con la abreviatura de cada una de ellas se muestran en las tablas 1 y 2

El ejemplo presentado en la tabla 3 indica que el macrobloque se encuentra entre *Muy Arriba y Arriba* y la posición horizontal entre *Muy Derecha y Derecha*, su movimiento horizontal es *Rápido Izquierda* y no existe movimiento en la componente vertical (*Sin Movimiento*).

### 2.2. Vectores lingüísticos de movimiento válidos

**Definition 2.** *Se dice que un vector de movimiento lingüístico es válido (a partir de ahora VLMV) si aporta información sobre la dirección y la velocidad de un macrobloque, es decir, al menos uno de sus dos componentes derecha_x y abajo_x es distinto a la etiqueta "Sin Movimiento"*

**Cuadro 1.** Conjunto de etiquetas lingüísticas variable velocidad vertical y horizontal

| | | | |
|---|---|---|---|
| RD | Rápido Derecha | RA | Rápido Abajo |
| ND | Normal Derecha | NA | Normal Abajo |
| LD | Lento Derecha | LA | Lento Abajo |
| SM | Sin Movimiento | SM | Sin Movimiento |
| LI | Lento Izquierda | LU | Lento Arriba |
| NI | Normal Izquierda | NU | Normal Arriba |
| RI | Rápido Izquierda | RU | Rápido Arriba |

**Cuadro 2.** Conjunto de etiquetas lingüísticas variable posición vertical y horizontal

| | | | |
|---|---|---|---|
| MA | Muy Arriba | MI | Muy Izquierda |
| A | Arriba | I | Izquierda |
| CV | Centro Vertical | CH | Centro Horizontal |
| AB | Abajo | D | Derecha |
| MAB | Muy Abajo | MD | Muy Derecha |

### 2.3. Secuencia de vídeo lingüística

**Definition 3.** *Una* ***secuencia de vídeo lingüística*** *es el conjunto total de vectores de movimiento lingüísticos válidos que se encuentran en la secuencia de vídeo completa agrupados por frame.*

### 2.4. Objeto lingüístico

**Definition 4.** *Un Objeto Lingüístico es la descripción semántica de un objeto en movimiento y está representado como:*

$$LingObjeto{=}< NumeroFrame, Tamaño, LI_r, LI_d, LI_v, LI_h >$$

*donde el primer elemento indica el número de frame donde aparece el objeto, el segundo componente es el tamaño (número de vectores de movimiento lingüísticos válidos asociados al objeto) y los otros cuatro elementos son intervalos lingüísticos que contienen información sobre la velocidad y la posición del objeto.*

## 3. Descripción del sistema

### 3.1. Decodificación de la señal MPEG

La entrada al sistema es una secuencia de vídeo en formato MPEG, por tanto se necesita de una herramienta software para extraer toda la información de entrada. Se utilizará una de las "Herramientas MPEG Berkeley"[7] para extraer los vectores de movimiento, información básica de cada macrobloque y parámetros del vídeo (frecuencia de muestreo, tamaño, ...).

**Cuadro 3.** Ejemplo de VLMV

| Numero Frame | 39 |
|---|---|
| $FP_r$ | [Rápido Izquierda, 1.0] |
| $FP_d$ | [Sin Movimiento, 1.0] |
| $FP_v$ | [Muy Arriba, 0.575, Arriba, 0.425] |
| $FP_h$ | [Muy Derecha, 0.333, Derecha, 0.667] |

### 3.2. Obtención de la escena lingüística

Una vez almacenada toda la información necesaria, se extraen todos los vectores de cada frame de manera independiente, se fuzzifican, convirtiéndolos a vectores de movimiento lingüísticos y de éstos se filtran aquellos que aportan información válida (VLMVs). Los algoritmos necesarios para la realizacin de este proceso se describen con detalle en [6].

### 3.3. Agrupamiento intraframe

Este algoritmo es utilizado por nuestro sistema para localizar objetos que se caracterizan por un desplazamiento uniforme. **La idea principal en la que se basa el algoritmo es la localización, en cada frame de manera individual, de VLMVs localizados en posiciones çercanas.[en] la imagen y con velocidades "similares"**.

La çercaníaz parecido se define como una **medida difusa de similitud** [8] entre intervalos lingüísticos, o más concretamente, en generalizaciones de los mismos como son los vectores de movimiento lingüśticos, los objetos lingüísticos, ...

La entrada del algoritmo es la Secuencia Lingüística obtenida en la etapa anterior de manera directa desde la secuencia de vídeo MPEG. Como salida se obtienen todos los objetos con movimiento detectados agrupados por número de frame y representados como Objetos Lingüísticos.

**Definition 5.** *El cálculo de la distancia entre VLMVs se basa en una medida de distancia o separabilidad presentada en [8] que es una función real basada en el cálculo del area del conjunto difuso "Separación entre".*

*El valor normalizado se fuzzifica según una variable lingüística continua trapezoidal, cuyo soporte se muestra en la tabla 4, que indica el grado de similitud entre dos VLMVs.*

### 3.4. Agrupamiento interframe

La entrada al algoritmo es el conjunto $LINGOBJETO$ donde se almacenan todos los objetos con movimiento en la secuencia completa de vídeo ordenados por frame. Como salida obtenemos la trayectoria de cada uno de los objetos con movimiento en la secuencia de vídeo. Esta salida es el resultado final del sistema.

**Cuadro 4.** Conjunto de etiquetas lingüísticas variable Similitud

| Etiqueta | a | b | c | d |
|---|---|---|---|---|
| NADA | 0 | 0 | 0,10 | 0,15 |
| POCO | 0,10 | 0,15 | 0,35 | 0,40 |
| MEDIO | 0,35 | 0,40 | 0,60 | 0,65 |
| MUCHO | 0,60 | 0,65 | 0,85 | 0,90 |
| IGUAL | 0,85 | 0,90 | 1,00 | 1,00 |

## 4. Resultados experimentales

El software de simulación se ha desarrollado utilizando el lenguaje de programación python. La entrada al programa es una secuencia de vídeo en formato MPEG con una restricción: el tamaño es fijo, 300 macrobloques por frame organizados en 20 filas x 15 columnas.

En la tabla 5 se encuentra una salida resumida de los un experimento. Se muestran aquellos frames en los que cambia los valores de alguna etiqueta.

### 4.1. Desplazamiento horizontal de dos objetos

En el vídeo analizado se muestran dos objetos que se desplazan de izquierda a derecha. Una de ellos (objeto A) lo hace a mayor velocidad y atraviesa toda la pantalla. El otro (objeto B) aparece por la izquierda de la imagen y se detiene en el centro de la misma. Ambos tienen trayectorias parecidas pero en instantes temporales diferentes.

En la tabla 5 se observa como se detectan los dos objetos pero por el número de frame se deduce el retardo de aparición de la objeto A respecto a la B. La velocidad vertical y horizontal del objeto A es mayor que el B. Este último se va frenando hasta pararse en el centro de la imagen.

## 5. Conclusiones

Nuestro sistema obtiene de manera automática una descripción lingüística de la posición y velocidad de los objetos que aparecen en una secuencia de imágenes trabajando de manera directa sobre la senñal de vídeo comprimida. El coste computacional es bajo a la hora de obtener toda la información previa necesaria para la segmentación de los diferentes objetivos. Funciona de manera correcta con un simple proceso de fuzzificación mediante el cual se obtienen una serie de elementos linguísticos que se agrupan en dos etapas basándose en una función de similitud difusa.

## Agradecimientos

Este trabajo ha sido financiado por el Ministerio de Educación y Ciencia a través del proyecto de investigación DIMOCLUST (TIC2003-08807-C02-02).

**Cuadro 5.** Ejemplo número 1

| **Frame** | **Vel_x** | **Vel_y** | **Pos_x** | **Pos_y** |
|---|---|---|---|---|
| Objeto 1 | | | | |
| 27 | RD | LA | AB, MAB, | MI, I |
| 30 | RD | LA, SM | AB, MAB | MI, I |
| 39 | RD | LA, SM | AB, MAB | I, CH |
| 42 | ND, RD | LA, SM | AB, MAB | CH, D |
| 44 | ND, RD | LA | AB, MAB | D, MD |
| 50 | ND | SM | AB, MAB | MD |
| Objeto 2 | | | | |
| 37 | ND, LD | SM, LA | AB, MAB | MI |
| 43 | ND, LD | SM, LA | AB, MAB | MI, I |
| 56 | ND, LD | SM | AB, MAB | I, CH |
| 63 | LD | SM, LA | AB, MAB | I, CH |

## Referencias

1. Dubois, D. and Prade. H. Fuzzy sets and systems. Theory and Applications. Academic, New York. (1980)
2. Pilu, M.: On using raw mpeg motion vectors to determine global camera motion. Technical report, HP Laboratories. (1997)
3. Venkatesh, R., Ramakrishnan, K.: Background sprite generation using MPEG motion vectors. In Proceedings of ICVGIP. Ahmedabadn, India. (2002) 7–12
4. Kim, N., Kim, T. and Choi, J.S.: Motion analysis using the normalization of motion vectors on MPEG compressed domain. In Proceedings of ITC-CSCC. Phuket, Thailand. (2002) 1408–1411
5. Gilvarry, J.: Extraction of motion vectors from an MPEG stream. Technical Report. Dublin City University. (1999)
6. Rodriguez-Benitez, L., Moreno-Garcia, J., Castro-Schez, J.J., Jimenez, L., Garcia, S.: Linguistic Motion Description for an Object on Mpeg Compressed Domain. Eleventh International Fuzzy Systems Association World Congress (IFSA'05). Beijing, China. (2005) Vol. II 1064–1070
7. Berkeley MPEG tools. http://bmrc.berkeley.edu/frame/research/mpeg/
8. Castro-Schez, J.J., Castro, J.L., Zurita, J.M.: Fuzzy repertory table, a method for acquiring knowledge about input variables to machine learning algorithms. IEEE Transactions on Fuzzy Systems. (2004). 12(1) 123–139.

# Evaluación de la selección, traducción y pesado de los rasgos para la mejora del clustering multilingüe

S. Montalvo[1], A. Navarro[1], R. Martínez[2], A. Casillas[3] y V. Fresno[1]

[1] Dpt. Informática, Estadística y Telemática
Universidad Rey Juan Carlos
{soto.montalvo, victor.fresno}@urjc.es, axelux@gmail.com
[2] Dpt. de Lenguajes y Sistemas Informáticos
UNED
raquel@lsi.uned.es
[3] Dpt. Electricidad y Electrónica
Universidad del País Vasco
arantza.casillas@ehu.es

**Resumen** En este trabajo hemos realizado un estudio para evaluar el impacto de utilizar diferentes representaciones de documentos en el resultado del clustering multilingüe. Para ello, seguimos un modelo basado en la selección y traducción de rasgos. La selección se basa en la utilización de información sobre la categoría gramatical y el contexto. La traducción se ha llevado a cabo utilizando *EuroWordNet 1.0* y aplicando un método de desambiguación automática. Además, se han utilizado diferentes funciones de pesado de los rasgos (TF, TF-IDF y WIDF). El objetivo principal es estudiar la importancia de cada uno de estos elementos y así poder determinar una o varias combinaciones de ellos que conduzcan a obtener buenos resultados en el clustering multilingüe. La evaluación se ha llevado a cabo con un corpus comparable de noticias escritas en castellano e inglés. Se ha usado un algoritmo de clustering de partición de la librería CLUTO y la calidad de los resultados se ha determinado mediante una medida de evaluación externa. Los mejores resultados se obtienen representando con las entidades nombradas de todo el documento y con las funciones de pesado TF y TF-IDF.

## 1. Introducción

El clustering multilingüe parte de un conjunto de documentos escritos en varios idiomas y tiene como objetivo agruparlos de manera que se puedan obtener clusters o grupos multilingües. Un cluster multilingüe contendrá aquellos documentos que estén relacionados o traten del mismo tema aunque estén escritos en diferentes lenguas. Mientras que un cluster monolingüe estará compuesto únicamente de documentos relacionados escritos en el mismo idioma.

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 769-778, 2006.

El aumento de la cantidad de documentos electrónicos escritos en diferentes lenguas conlleva la necesidad de desarrollar sistemas que manejen toda esta información y faciliten su acceso a los potenciales usuarios. El clustering multilingüe puede facilitar tareas como la recuperación de información multilingüe (agrupando documentos antes y después de la recuperación), alineación de corpora paralelos y comparables, entrenamiento de parámetros para sistemas de traducción automática estadística, etc.

Los diferentes enfoques a la hora de abordar el clustering multilingüe se pueden clasificar en dos grandes grupos: por un lado, aquellos que hacen uso de técnicas de traducción y, por otro, aquellos que transforman el documento en una representación independiente del lenguaje.

En los sistemas basados en traducción, bien para traducir los documentos completos a una lengua eje, o bien para seleccionar ciertos rasgos y sólo traducir éstos a una lengua eje, es crucial la exactitud de la traducción obtenida. Los recursos bilingües que se utilizan, normalmente ofrecen varias posibilidades o sentidos en la traducción de una palabra y no es trivial elegir el adecuado. Aunque se pueden aplicar métodos de desambiguación automática, éstos no están libres de errores y no elegir el sentido de la traducción apropiado puede conducir a un agrupamiento erróneo.

Por otro lado, los sistemas que transforman el documento en una representación independiente del lenguaje tienen ciertas limitaciones. Por ejemplo, aquellos que trabajan con tesauros dependen fundamentalmente del alcance de éstos. La identificación de datos numéricos y de fechas puede resultar muy apropiada para ciertos tipos de clustering y documentos. Sin embargo, en otros casos este tipo de datos puede no ser relevante y resultar una fuente de ruido. Además, la identificación de cognados, que suele ser otra forma de representación independiente, está muy ligada al tipo de lenguas de que conste el corpus.

En este trabajo presentamos los resultados de un estudio que hemos llevado a cabo para evaluar el impacto de la utilización de diferentes representaciones de los documentos en el resultado del clustering multilingüe. Para ello, hemos utilizado el modelo basado en la selección y traducción de los rasgos. La selección se ha basado en seleccionar o no rasgos pertenecientes a diferentes categorías gramaticales, entidades nombradas y determinados contextos. La traducción se ha llevado a cabo mediante *EuroWordNet 1.0* [Vossen 1998], aplicando un método de desambiguación automática. También hemos utilizado diferentes funciones de pesado de los rasgos (TF, TF-IDF y WIDF). El objetivo principal es estudiar la importancia de cada uno de estos aspectos y, así, poder determinar una o varias combinaciones de ellos que conduzcan a la obtención de buenos resultados en el clustering multilingüe.

La evaluación se ha llevado a cabo con un corpus comparable de noticias escritas en castellano e inglés. Con el fin de utilizar medidas de evaluación externa, se ha recopilado un subconjunto de noticias comparable que ha sido agrupado manualmente y que ha servido como solución de referencia. Como el énfasis del estudio se ha puesto en la selección y traducción de los rasgos y no

en el algoritmo de clustering, se ha utilizado un algoritmo de partición bien conocido en la literatura.

El resto del artículo se estructura como sigue: en la Sección 2 se describen brevemente algunos trabajos relacionados. La Sección 3 explica, mostrando cada una de sus fases, el estudio realizado. En la Sección 4 se presenta la colección utilizada en la evaluación, así como los experimentos junto con los resultados obtenidos. Por último, la Sección 5 incluye las conclusiones y trabajo futuro.

## 2. Trabajos relacionados

El clustering multilingüe de documentos normalmente se aplica sobre corpus paralelos [Silva et. al. 2004] o corpus comparables ([Rauber et. al. 2001], [Mathieu et. al 2004], [Pouliquen et. al. 2004], [Chen and Lin 2000], [Steinberger et. al. 2002], [Lawrence 2003]).

Si tenemos en cuenta los trabajos basados en el uso de técnicas de traducción, se emplean dos estrategias: (1) traducir el documento completo a una lengua eje, y (2) traducir algunos rasgos del documento a una lengua eje.

Con respecto a la primera aproximación, algunos autores utilizan sistemas de traducción automática, mientras que otros traducen el documento palabra a palabra, consultando un diccionario bilingüe. En [Lawrence 2003] se presentan varios experimentos de clustering sobre un corpus comparable de Ruso e Inglés; varios de estos experimentos están basados en el uso de sistemas de traducción automática.

Cuando se trata de traducir sólo algunos rasgos del documento, en primer lugar es necesario seleccionar qué rasgos se van a traducir (normalmente nombres, entidades nombradas, verbos y adjetivos) para, a continuación, traducir dichos rasgos mediante un diccionario bilingüe o consultando un corpus paralelo.

En [Mathieu et. al 2004], antes del proceso de clustering, se lleva a cabo un análisis lingüístico que extrae los lemas y reconoce entidades nombradas de diversas categorías (lugar, organización, persona, expresión temporal, expresión numérica, evento). Por lo tanto, los documentos se representan mediante un conjunto de rasgos. Además, los autores tienen en cuenta su frecuencia para seleccionar los rasgos más relevantes. Finalmente, utilizan un diccionario bilingüe para traducir los rasgos seleccionados. En [Rauber et. al. 2001] los autores presentan una metodología que consiste en la extracción de todas las palabras que aparecen en $n$ documentos, exceptuando las palabras vacías de contenido. Posteriormente, mediante sistemas de traducción automática construyen un corpus monolingüe. Una vez finalizado el proceso de traducción, de forma automática, los documentos se organizan en diferentes clusters usando un método de aprendizaje no supervisado mediante redes neuronales.

Algunas aproximaciones llevan a cabo un proceso de clustering independiente en los documentos de cada lengua, es decir, un clustering monolingüe. Posteriormente, tratan de encontrar relaciones entre los clusters monolingües obtenidos, generando así clusters multilingües. Sin embargo, otros trabajos comienzan con un proceso de clustering multilingüe buscando relaciones

entre los documentos de todos los idiomas involucrados. Este es el caso de [Chen and Lin 2000], donde los autores proponen una arquitectura de resúmenes de noticias que incluye un proceso de clustering monolingüe y multilingüe. El clustering multilingüe toma como entrada los resultados de una fase previa de clustering monolingüe. Los autores seleccionan diferentes tipos de rasgos dependiendo del tipo de clustering: para el clustering monolingüe usan entidades nombradas y para el clustering multilingüe, además, tienen en cuenta los verbos.

Las estrategias de clustering que generan para cada documento una representación independiente del idioma en el que está escrito, intentan estandarizar o normalizar los contenidos de varias formas: (1) mapeando los contenidos a una representación independiente, o (2) reconociendo rasgos independientes del lenguaje dentro del texto. Ambas posibilidades se pueden emplear de forma aislada o combinada.

La primera aproximación requiere la existencia de recursos lingüísticos multilingües, como tesauros, para crear una representación del texto que consista en un conjunto de entradas de tesauro. Normalmente, en un tesauro multilingüe, los elementos de las diferentes lenguas se relacionan mediante entradas independientes de la lengua. Por lo tanto, dos documentos escritos en distinto idioma pueden ser considerados similares si tienen una representación parecida de acuerdo a lo que indica el tesauro. En algunos casos, es necesario el uso de tesauros combinados con métodos de aprendizaje automático para realizar un mapeo correcto de los documentos con el tesauro. En [Steinberger et. al. 2002] calculan la similitud semántica representando los contenidos de los documentos de forma independiente a la lengua en la que están escritos, por medio del tesauro *Eurovoc*.

La segunda aproximación, reconocer en el texto rasgos independientes de la lengua, implica poder identificar elementos como: fechas, números y entidades nombradas. Por ejemplo, en [Silva et. al. 2004] los autores presentan un método basado en lo que denominan Expresiones Relevantes (ER). Una expresión relevante es una unidad léxica de cualquier longitud extraída de los documentos mediante la herramienta LiPXtractor. Las expresiones relevantes se usan para extraer un conjunto de rasgos, pero los clusters obtenidos son monolingües.

Otros trabajos combinan la identificación de rasgos independientes del idioma (como números, fechas, ...) con el mapeo de los rasgos del texto con un tesauro. En [Pouliquen et. al. 2004] la similitud entre los clusters multilingües se basa en la combinación lineal de tres tipos de entradas: (a) cognados, (b) detección automática de nombres de referencias geográficas, y (c) los resultados de un proceso de mapeo de un sistema de clasificación multilingüe, que mapea los documentos en un tesaruo multilingüe (*Eurovoc*).

En [Steinberger et. al. 2004] proponen extraer características independientes del idioma usando gazetters y expresiones regulares, además de tesauros y sistemas de clasificación.

## 3. Representación y clustering de documentos

Para la representación de los documentos utilizamos el modelo de espacio vectorial [Salton and McGill 1983]. Según este modelo, para cada documento se obtiene un vector en el que cada componente representa el peso de un rasgo en dicho documento.

Nuestra propuesta para el clustering multilingüe de documentos se compone de las siguientes fases:

1. Selección de rasgos.
2. Generación de la representación intermedia.
3. Traducción de rasgos.
4. Generación de la representación final.
5. Clustering.

### 3.1. Selección de rasgos

En esta primera fase se seleccionan los rasgos que se van a tener en cuenta en la representación de cada documento. En nuestro enfoque, esta selección requiere que el corpus esté analizado morfo-sintácticamente, lematizado y con las entidades nombradas identificadas y categorizadas. En este trabajo únicamente hemos tenido en cuenta las entidades de las categorías PERSONA, LUGAR, ORGANIZACIÓN y MISCELÁNEA.

La selección se basa en 3 aspectos:

- La categoría gramatical.
  Normalmente se consideran categorías más discriminantes los nombres, verbos y adjetivos.
- Ser o no entidad nombrada.
  En el caso particular de los documentos de noticias, tiene sentido que las entidades nombradas sean consideradas como rasgos realmente discriminantes. Por ello, en principio, cabría pensar que las representaciones que incluyan dicho tipo de rasgos conseguirán mejores resultados en el clustering que aquellas que no los consideren. Aunque realmente serán útiles para la representación en la medida en que los recursos utilizados para la traducción sean capaces de dar cuenta de ellas.
- El contexto.
  Otro aspecto a tener en cuenta en la selección de rasgos es el contexto en el que se encuentran. En el estilo periodístico es habitual que en el primer párrafo se resuma el contenido primordial de la noticia. De ahí que nosotros hayamos considerado dos tipos de contexto en nuestro estudio: el documento completo y el primer párrafo.

### 3.2. Generación de la representación intermedia

Una vez extraídos los rasgos que se van a tener en cuenta, se generará una representación intermedia por cada parte monolingüe del corpus.

Esta representación intermedia consiste en una matriz, donde cada fila es un vector que se corresponde con uno de los documentos del corpus. Cada columna representa uno de los rasgos que aparecen en el corpus y que ha sido seleccionado.

Los valores de cada componente de los diferentes vectores que forman la matriz se asignan por medio de funciones de peso. En este estudio hemos utilizado funciones bien conocidas en la literatura (TF, TF-IDF y WIDF) que se describen a continuación:

*Term Frequency*, TF [Luhn 1957]: cada rasgo tiene una importancia proporcional al número de veces que aparece en el documento.

*Inverse Term Frequency*, TF-IDF [Salton and Yang 1973]: la combinación de pesos de un término $t$ en un documento $d$, siendo $N$ el número de documentos y $df(t)$ el número de documentos que contienen el rasgo $t$, viene dada por:

$$TF - IDF(d,t) = TF(d,t) \times IDF(t); \;\; IDF(t) = \log \frac{N}{df(t)} \tag{1}$$

*Weighted Inverse Term Frequency*, WIDF [Salton 1989]: extensión de IDF que incorpora la frecuencia del término sobre la colección de documentos:

$$WIDF(d,t) = TF(d,t) \sum_{i \epsilon D} TF(i,t) \tag{2}$$

### 3.3. Traducción de rasgos

Para traducir los rasgos se dispone de la base de datos léxica *EuroWordNet 1.0*. Con este recurso se traducen a castellano todos los rasgos que aparecen en la representación intermedia del corpus en inglés.

En la traducción, uno de los factores clave es la desambiguación automática. Cuando para un rasgo en inglés se obtiene más de un sentido posible como traducción hemos aplicado el método de desambiguación que se describe a continuación. De los diferentes sentidos obtenidos por *EuroWordNet*, se elige aquél que esté presente entre los rasgos de la matriz del corpus en castellano. Nuestra hipótesis es que dado que trabajamos con un corpus comparable, esperamos que la traducción correcta de una palabra aparezca, en la mayoría de los casos, en el corpus del otro idioma.

En el caso de que al intentar traducir un rasgo no se encuentre ninguna traducción, dicho rasgo se elimina de la representación, salvo que se trate de una entidad nombrada. Así, se contempla que pueda haber entidades nombradas, aunque no se puedan traducir, ya que pueden coincidir en ambas lenguas.

### 3.4. Generación de representación final

Una vez que se han generado dos representaciones intermedias, una por cada corpus monolingüe y, además, la del corpus en inglés se ha traducido, se fusionan en una única representación. Entonces, como representación final para el proceso de clustering multilingüe se dispone de una única matriz.

### 3.5. Clustering

Para realizar el clustering usamos un algoritmo de partición. En particular, el algoritmo *Direct* de la conocida librería CLUTO [Karypis 2002]. El número total de clusters que se quieren obtener es un dato que hay que proporcionar al algoritmo.

## 4. Evaluación

En esta sección se presenta el corpus con el que se realiza la evaluación, así como los experimentos realizados y los resultados obtenidos.

### 4.1. Corpus

Un corpus comparable es una colección de textos similares en diferentes idiomas o diferentes variedades de un mismo idioma. En este trabajo usamos una colección de noticias escritas en inglés y castellano, referentes al mismo periodo de tiempo. Las noticias se encuentran clasificadas y se trata de noticias de la agencia EFE que han sido recopiladas en el proyecto HERMES[4]. Esta colección se puede considerar como un corpus comparable. Para realizar la evaluación hemos usado un subconjunto de noticias que se compone de 79 noticias en castellano y 70 noticias en inglés, en total 149 noticias.

Para poder comprobar la bondad de los resultados del algoritmo de clustering con las diferentes representaciones se ha realizado una agrupación manual de la colección. Tres personas han sido las encargadas de leer los documentos y formar grupos atendiendo a sus contenidos. La solución manual se compone de 26 clusters, siendo todos ellos multilingües.

### 4.2. Experimentos y resultados

Se realizaron experimentos con diferentes combinaciones de todos los criterios de selección de rasgos descritos en la Sección 3.

La calidad de los resultados se ha evaluado mediante una medida de evaluación externa, la medida-F [van Rijsbergen 1974]. Esta medida compara la solución obtenida por nuestro sistema con la solución humana. La medida-F combina las medidas de precisión y recall:

$$F(i,j) = \frac{2 \times Recall(i,j) \times Precision(i,j)}{(Precision(i,j) + Recall(i,j)}, \tag{3}$$

denominamos clase al grupo de la solución humana y cluster al grupo devuelto por el sistema, así: $Recall(i,j) = \frac{n_{ij}}{n_i}$, $Precision(i,j) = \frac{n_{ij}}{n_j}$, donde $n_{ij}$ es el

[4] http://nlp.uned.es/hermes/index.html

número de miembros de la clase $i$ en el cluster $j$, $n_j$ es el número de miembros del cluster $j$ y $n_i$ el número de miembros de la clase $i$. Para todos los clusters:

$$F = \sum_i \frac{n_i}{n} max\{F(i,j)\}, \tag{4}$$

donde $n$ es el número de documentos. Esta función está acotada entre los valores 0 y 1, que representan la peor y mejor calidad de clustering respectivamente.

En la tabla 1 se presentan los mejores resultados obtenidos con las diferentes medidas de pesado utilizadas, TF, TF-IDF y WIDF respectivamente.

La primera columna de las tablas indica la categoría gramatical de los rasgos seleccionados: NOM (nombres), VER (verbos), ADJ (adjetivos), NE (entidades nombradas) y $1^{er}$ PAR (todos los rasgos de las categorías seleccionadas que aparezcan en el primer párrafo). La segunda columna representa la medida-F y la tercera columna indica la relación entre el número de clusters multilingües obtenidos y el número total de clusters multilingües que se deberían haber obtenido. Recuérdese que en el corpus de evaluación la solución manual tenía todos los clusters multilingües.

**Tabla 1.** Resultados de clustering con las diversas representaciones

| **Rasgos seleccionados** | **F. peso** | **medida-F** | **Clusters Multl./Total** |
|---|---|---|---|
| NOM, VER | TF | 0.8164 | 16/26 |
| NOM, VER, $1^{er}$ PAR | TF | 0.7214 | 15/26 |
| NOM, ADJ | TF | 0.8555 | 18/26 |
| NOM, ADJ, $1^{er}$ PAR | TF | 0.7769 | 21/26 |
| NOM, VER, ADJ | TF | 0.8027 | 16/26 |
| NOM, VER, ADJ, $1^{er}$ PAR | TF | 0.7321 | 14/26 |
| NE | TF | 0.8628 | 18/26 |
| NE, $1^{er}$ PAR | TF | 0.7012 | 15/26 |
| NOM, VER | TF-IDF | 0.8534 | 21/26 |
| NOM, VER, $1^{er}$ PAR | TF-IDF | 0.7372 | 19/26 |
| NOM, ADJ | TF-IDF | 0.8406 | 21/26 |
| NOM, ADJ, $1^{er}$ PAR | TF-IDF | 0.7517 | 22/26 |
| NOM, VER, ADJ | TF-IDF | 0.7984 | 20/26 |
| NOM, VER, ADJ, $1^{er}$ PAR | TF-IDF | 0.7570 | 21/26 |
| NE | TF-IDF | 0.8117 | 19/26 |
| NE, $1^{er}$ PAR | TF-IDF | 0.6823 | 21/26 |
| NOM, VER | WIDF | 0.6705 | 26/26 |
| NOM, VER, $1^{er}$ PAR | WIDF | 0.5560 | 25/26 |
| NOM, ADJ | WIDF | 0.7302 | 26/26 |
| NOM, ADJ, $1^{er}$ PAR | WIDF | 0.6486 | 26/26 |
| NOM, VER, ADJ | WIDF | 0.7090 | 26/26 |
| NOM, VER, ADJ, $1^{er}$ PAR | WIDF | 0.6155 | 25/26 |
| NE | WIDF | 0.7323 | 24/26 |
| NE, $1^{er}$ PAR | WIDF | 0.6747 | 22/26 |

Como era de esperar los resultados varían en función de la representación utilizada.

Los mejores valores de la medida-F se obtienen, en general, con las funciones de pesado TF y TF-IDF, quedando a bastante distancia las representaciones con la función WIDF. El único punto a favor de esta última es que sus soluciones, aunque de peor calidad, obtienen un número de clusters multilingües más cercano al de la solución manual.

En cuanto al tipo de rasgos, las entidades nombradas (NE) obtienen los mejores valores de medida-F en dos de las representaciones, y el tercer mejor valor en la otra. Estos resultados indican que se trata de rasgos muy representativos de los documentos del corpus. Por otra parte, las representaciones con NOM, ADJ y NOM, VER también obtienen buenos resultados con las tres funciones de pesado.

## 5. Conclusiones y trabajos futuros

Hemos realizado un estudio para determinar el impacto de la utilización de diferentes representaciones de los documentos en el resultado del clustering multilingüe. Para ello, partiendo del modelo basado en la traducción de rasgos de los documentos, el énfasis se ha puesto en una selección de rasgos basada en información obtenida de: las categorías gramaticales, el uso de las entidades nombradas y la elección del contexto. Además, se han utilizado diferentes funciones de pesado de los rasgos. Para la desambiguación en la traducción de los rasgos se ha propuesto un método sencillo basado en la naturaleza de los corpus comparables.

La experimentación se ha realizado sobre un corpus comparable de noticias escritas en castellano e inglés y se ha utilizado un conocido algoritmo de clustering.

Los resultados indican que las representaciones obtenidas con las funciones de pesado TF y TF-IDF obtienen un agrupamiento de más calidad que con la función WIDF. Por otra parte, las entidades nombradas (NE) resultan ser los rasgos que mejor representan el corpus utilizado en la experimentación, un corpus de noticias. También las representaciones con rasgos de tipo NOM, ADJ y NOM, VER muestran un buen comportamiento.

En cuanto a la elección del contexto, se aprecia que a menor tamaño de contexto los resultados empeoran. En todas las representaciones cuyo contexto es el primer párrafo, los resultados empeoran con respecto a las representaciones cuyo contexto es el documento completo.

Como posibles trabajos futuros están combinar diferentes recursos para la traducción con el fin de aumentar el número de entidades nombradas que se traducen, por ejemplo utilizar gazetters además de la base de datos léxica y aplicar reglas de equivalencia. Asimismo, la distinción entre las diferentes categorías de entidades nombradas puede resultar útil en la fase de selección de rasgos.

## Referencias

[Karypis 2002] Karypis G.: "CLUTO: A Clustering Toolkit". Technical Report: 02-017. University of Minnesota, Department of Computer Science, Minneapolis, MN 55455, 2002.

[Mathieu et. al 2004] Benoit Mathieu and Romanic Besancon and Christian Fluhr. "Multilingual document clusters discovery". RIAO 2004, p. 1-10, 2004.

[Pouliquen et. al. 2004] Bruno Pouliquen and Ralf Steinberger and Camelia Ignat and Emilia Käsper and Irina Temikova. "Multilingual and cross-lingual news topic tracking". Proceedings of the $20^{th}$ International Conference on computational Linguistics, p. 23-27, 2004.

[Rauber et. al. 2001] Andreas Rauber and Michael Dittenbach and Dieter Merkl. "Towards Automatic Content-Based Organization of Multilingual Digital Libraries: An English, French, and German View of the Russian Information Agency Novosti News". Third All-Russian Conference Digital Libraries: Advanced Methods and Technologies, Digital Collections Petrozavodsk, RCDI'2001.

[Silva et. al. 2004] Joaquin Silva and J. Mexia and Carlos Coelho and Gabriel Lopes. "A Statistical Approach for Multilingual Document Clustering and Topic Extraction form Clusters". Pliska Studia Mathematica Bulgarica, v.16, p. 207-228, 2004.

[Vossen 1998] Vossen, P. "Introduction to EuroWordNet". Computers and the Humanities Special Issue on EuroWordNet, 1998.

[Chen and Lin 2000] Hsin-Hsi Chen and Chuan-Jie Lin. "A Multilingual News Summarizer". Proceedings of 18th International Conference on Computational Linguistics, p. 159-165, 2000.

[Steinberger et. al. 2002] Ralf Steinberger and Bruno Pouliquen and Johan Scheer. "Cross-Lingual Document Similarity Calculation Using the Multilingual Thesaurus EUROVOC". CICling'2002, p. 415-424.

[Lawrence 2003] Lawrence J. Leftin. "Newsblaster Russian-English Clustering Performance Analysis". Columbia computer science Technical Reports.

[Steinberger et. al. 2004] Ralf Steinberger and Bruno Pouliquen and Camelia Ignat. "Exploting multilingual nomenclatures and language-independent text features as an interlingua for cross-lingual text analysis applications". SILTC 2004.

[Luhn 1957] H. P. Luhn. "A statistical approach to mechanized encoding and searching of literaty information". IBM Journal of Research and Development, 1957.

[Salton and McGill 1983] G. Salton and M. McHill. "Introduction to Modern Information Retrieval". McGraw-Hill, New York. 1983.

[Salton and Yang 1973] G. Salton and C.S. Yang. "On the specification of term values in automatic indexing". Journal of Documentation, 1973.

[Salton 1989] G. Salton. "Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer". Addison-Wesley, 1989.

[van Rijsbergen 1974] C.J. van Rijsbergen. "Foundations of evaluation". Journal of Documentation, 30, p. 365-373, 1974.

[Vossen 1998] P. Vossen. "Introduction to EuroWordNet". Computers and the Humanities Special Issue on EuroWordNet. 1998.

# Etiquetación morfológica y automática del español mediante mecanismos de aprendizaje computacional y toma de decisiones

J.M. Alcaraz y J.M. Cadenas

Dpto. Ingeniería de la Información y las Comunicaciones.
Facultad de Informática. Universidad de Murcia. Spain.
jmalcaraz@gmail.com, jcadenas@um.es

**Resumen** Uno de los problemas más urgentes en la era de la palabra digital es hacer posible que el ordenador identifique correctamente las palabras. El primer eslabón en este camino es la etiquetación morfológica de textos, que viene a ser la llave que abrirá el camino a una posterior identificación o desambiguación semántica. El presente trabajo se propone como meta diseñar un algoritmo eficiente y fiable que permita marcar las palabras del español con unas 90 etiquetas, fundamentalmente morfológicas. El proceso es automático y se apoya en técnicas "inteligentes" que posibilitan la toma de decisiones fiables. Asimismo, el sistema está diseñado para mejorar sus propios resultados mediante el autoaprendizaje continuo de hechos pasados o de aciertos consolidados.

*Palabras clave:* Aprendizaje Automático, Toma de Decisiones, Heurísticas, Corpus Lingüístico, Etiquetación, Morforlogía

## 1. Introducción

La lingüística computacional tiene como objeto el estudio y procesamiento del lenguaje natural mediante ordenadores. Es, por lo tanto, una disciplina en la que necesariamente deben hermanarse y trabajar conjuntamente tanto los expertos en informática como los expertos en lengua, ya que unos y otros trabajan sobre la misma realidad. No cabe duda que el trabajo implicado se acerca, como mínimo, a lo que suele denominarse "inteligencia artificial", expresión que hace referencia al proceso mediante el cual se pretende que una máquina emule o reproduzca artificialmente determinados comportamientos humanos, cual sería, en el caso que nos ocupa, la comunicación mediante el lenguaje.

La primera dificultad a la que se enfrentan las máquinas en relación con el lenguaje natural es el alto grado de complejidad que lo define. Cualquiera de las lenguas habladas en el mundo por diferentes comunidades de hablantes participa de esta característica. La jerarquización lingüística existe, pero no es siempre lineal y sencilla de percibir. En el caso presente nos ocuparemos especialmente del estadio más básico: la desambiguación de las formas generalmente denominadas palabras. Resulta que estas formas no solamente se clasifican en diferentes

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 779-790, 2006.

categorías (nombre, verbo, adjetivo, etc.), sino que, además y con frecuencia, la misma forma es polivalente en relación con las categorías a las que puede asociarse. El panorama se complica todavía más si accedemos al significado que cada una de las formas puede conllevar: en ocasiones una misma forma supera los 50 o 60 significados diferentes (ejemplo: hacer, dar), ello sin contar con los numerosos conjuntos léxicos (generalmente llamados expresiones o frases hechas), que comportan significados también diferentes.

Si cada palabra o forma fuera unívoca en su función y significado, no se plantearía ningún problema para la máquina. Pero la realidad dista mucho de tal situación. La complejidad e impredecibilidad del ser humano queda perfectamente reflejada e ilustrada en el lenguaje. Y este es el reto que debe superar la máquina para actuar como intermediario capaz de valerse de ese mismo lenguaje para comunicarse con otros seres humanos. La primera condición que debe superarse es que la máquina ha de entender el lenguaje natural.

En los últimos veinte años han cobrado fuerza, como herramientas útiles para el análisis lingüístico, los llamados corpus lingüísticos, grandes recopilaciones de textos (20, 100, o más millones de palabras) que pueden ser tratados mediante programas informáticos. Este hecho permite algo que antes era impensable: obtener en pocos segundos, datos de gran interés sobre el uso que hacen los hablantes de las palabras, las oraciones o los significados. Si dichas recopilaciones son representativas de la lengua estudiada, se infiere la posibilidad de proyectar los resultados a niveles de generalización. Aquí reside uno de los principales apoyos sobre los cuales puede sustentarse el aprendizaje de las máquinas.

Un corpus lingüístico es un conjunto de datos lingüísticos (pertenecientes al uso oral y/o escrito de la lengua), sistematizados según determinados criterios, suficientemente extensos en amplitud y profundidad de manera que sean representativos del total del uso lingüístico o de alguno de sus ámbitos y dispuestos de tal modo que puedan ser procesados mediante ordenador con el fin de obtener resultados varios y útiles para la descripción y el análisis ([10]). Por lo tanto, los textos recopilados pueden referirse al lenguaje oral o escrito, y dentro de cada variante, a registros diferentes, a regiones geográficas diversas, etc.

A efectos de tratamiento computacional, el corpus es más útil conforme la máquina es capaz de recabar mayor información textual. Así, de la mera identificación de cada palabra por el espacio en blanco que media entre cada una de ellas, puede pasarse al reconocimiento de la categoría gramatical (nombre, verbo, adjetivo, etc.), o a la identificación de flexiones morfológicas (masculino/femenino, singular/plural, etc.). El ideal es lograr también captar los significados que subyacen en cada forma y en esto se está trabajando intensamente en los últimos diez o quince años. Superar cada una de estas fronteras implica la posibilidad de que el ordenador pase de un estadio a otro más avanzado, lo cual le permitirá compilar mayor información y, por lo tanto, afianzarse con mayor firmeza en un posible "proceso comunicativo" basado en el lenguaje natural.

El primer estadio significativo y útil para mejorar el proceso comunicativo es el dominio por parte del ordenador del nivel morfológico. Para llegar a este nivel la máquina debe partir de un texto que no posee información acerca de la

naturaleza de cada una de las palabras que lo componen. Se trata, por tanto, de que el ordenador supere este vacío mediante un trabajo previo de lo que los lingüistas llaman "etiquetación" morfológica de los textos.

Existen muchas publicaciones respecto al proceso para llevar a cabo la evolución a un nivel superior, y casi todas ellas responden al esquema adoptado por Mc Energy y Wilson, [1], que queda ilustrado en la siguiente figura:

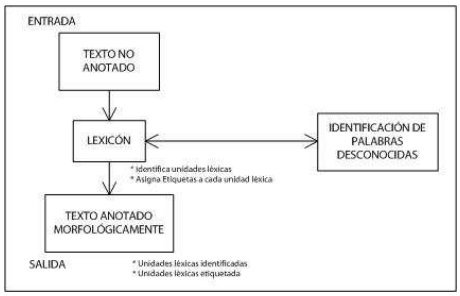



**Figura 1.** Proceso para la anotación morfosintáctica de un corpus (Mc Energy y Wilson)

La Fig. 1 refleja que el proceso para la anotación morfológica de un corpus lingüístico ha de proporcionar técnicas para identificar las diferentes unidades léxicas de un texto. Una vez identificadas tales unidades léxicas, debe procederse a la asignación de las etiquetas morfológicas pertinentes, basándose para ello en un lexicón o diccionario electrónico. De entre las diferentes implementaciones realizadas para llevar a cabo este proceso se puede resaltar la implementación para la etiquetación de la lengua española llevada a cabo por la Universidad Politécnica de Cataluña, conocida con el nombre de MACO (Morphological Analyser Corpus Oriented) [3]. MACO está siendo utilizado como base en diferentes proyectos que intentan crear sistemas más avanzados de desambiguación. Estos sistemas pueden actuar en niveles más altos de la lengua, como es el caso de MXPOST [4] en el nivel sintáctico, o por el contrario, puede perfeccionar el proceso de MACO, como ocurre en TreeTagger [6], que utiliza modelos estocásticos y árboles de decisión para llevar a cabo la desambiguación, o en Relax [5], que posibilita la mejora del proceso mediante restricciones contextuales.

Lo que se pretende en nuestro estudio no es un diseño totalmente innovador, sino más bien, una aportación complementaria que pretende contribuir al incremento de la fiabilidad en la etiquetación morfológica. Las características distintivas se refieren principalmente a la forma en la que se maneja el conocimiento heurístico y la fuente del mismo.

El trabajo se desarrolla con la siguiente estructura: en la sección 2 se explica la alternativa propuesta para llevar a cabo el proceso de etiquetación morfológica. En esta sección, presentamos las distintas heurísticas utilizadas y el mecanismo para el aprendizaje en la etiquetación del corpus. En la sección 3 se muestra la

metodología para las pruebas realizadas y los resultados obtenidos. La sección 4 finaliza con las conclusiones obtenidas y con propuestas para estudios posteriores.

## 2. Metodología para la etiquetación morfológica automática de un corpus lingüístico

El proceso de etiquetación morfológica de un corpus lingüístico desde un punto de vista externo podría describirse como el trabajo de un sistema inteligente que recibe como entrada un texto no anotado y produce como salida un texto anotado al nivel deseado, en el cual cada palabra recibe una etiqueta de carácter morfológico (categoría y flexión principalmente).

Para conseguir un nivel de fiabilidad elevado en el proceso de etiquetación, es preciso tomar decisiones fundamentales sobre la fuente de conocimiento en la que se basará el proceso de etiquetación, del cual se obtendrán las diferentes categorías gramaticales asociadas a la forma de las palabras. Esta fuente de información y consulta se denomina lexicón o diccionario electrónico.

### 2.1. El corpus lingüístico como diccionario electrónico

La calidad del diccionario electrónico es uno de los factores críticos para mejorar el rendimiento y el porcentaje de fiabilidad del sistema de etiquetación. La tendencia actual consiste en utilizar un diccionario almacenado de forma electrónica. Aquí se propone otra alternativa, al menos parcialmente diferente, que consiste en utilizar como fuente de información un corpus lingüístico ya etiquetado. Tiempo atrás, esto hubiera sido imposible, ya que no existían todavía corpus etiquetados suficientemente grandes y representativos del español. Pero la situación ya ha cambiado en este aspecto.

La forma de utilizar el corpus como lexicón consiste en extraer del mismo las diferentes categorías gramaticales de las palabras que lo componen y con ellas crear el lexicón. En el lexicón etiquetado, para cada palabra $P$ se pueden obtener las $P_i$ de categorías gramaticales asociadas a ella.

Se ha utilizado como corpus lingüístico etiquetado el corpus Cumbre-2M, [2], un conjunto de dos millones de palabras de la lengua española etiquetadas morfológicamente, revisado y validado manualmente. Una vez definida la fuente de conocimiento, se establece como punto de partida un algoritmo de decisión basado en árboles de regresión. Cuando se precisa información, el sistema formula preguntas en cada uno de los nodos que lo integran y en la solución se proporciona conocimiento heurístico sobre las categorías gramaticales.

Veamos con más detenimiento qué tipo de información (conocimiento heurístico) es utilizado para llevar a cabo la discriminación en el proceso de decisión que tiene lugar durante la etiquetación.

### 2.2. Heurística de la frecuencia

La heurística de la frecuencia denotada por $H_f$ consiste en almacenar el conocimiento del número de veces que una categoría gramatical ha sido elegida por un

experto en el proceso de etiquetación morfosintáctica. Esta información ha de conservarse de forma actualizada y consistente. La heurística es posible debido a que se está utilizando como fuente de información un corpus lingüístico ya etiquetado y revisado por "expertos". En consecuencia, se puede obtener información sobre el número de veces que una determinada palabra fue etiquetada con una categoría gramatical concreta. Esta información es susceptible de ser utilizada en posteriores procesos de etiquetado.

| Forma de Palabra | Categoría Gramatical | Frecuencia |
|---|---|---|
| Volante | Adjetivo | 2 |
| Volante | Nombre Común | 25 |

**Cuadro 1.** Conocimiento heurístico para la heurística de la frecuencia

La tabla 1 muestra la información heurística sobre la palabra "volante". Esta palabra, como muchas otras, es ambigua y por tanto puede poseer varias categorías gramaticales ("Adjetivo", "Nombre Común"). Lo que se pretende resaltar en el ejemplo es solo la frecuencia asignada por el experto a cada categoría.

### 2.3. Heurística de la frecuencia en el contexto

Dado que partimos de un corpus lingüístico como fuente de conocimiento, podemos aprovechar este aspecto para obtener información que no podríamos obtener de un diccionario electrónico en el que las palabras constasen como formas aisladas. En nuestro caso podemos recabar también la ayuda del contexto.

La heurística de la frecuencia en el contexto denotada por $H_c$ se basa en la heurística anteriormente descrita. Nosotros añadimos información extraída del contexto oracional en que se encuentra la palabra que esté siendo etiquetada. Esta información puede incrementar considerablemente los aciertos en el proceso de etiquetado, ya que las palabras co-textuales pueden ser decisivas para determinar la categoría gramatical de la forma en cuestión. La manera de obtener información a partir del contexto oracional se ha limitado a tener en cuenta la categoría gramatical de la siguiente palabra. En nuestro caso podemos recabar información sobre el número de veces que una palabra ha sido etiquetada de un modo determinado en el contexto en que aparece cada forma.

El ejemplo de la tabla 2 muestra el conocimiento heurístico almacenado sobre la palabra "volante"para esta heurística. Comprobamos que el experto ha etiquetado la palabra "volante" con la categoría gramatical de "adjetivo", al mismo tiempo que sigue la categoría gramatical de "Conjunción Coordinante".

Existe una relación importante entre las dos heurísticas, relación que se ha de cumplir siempre para todas las palabras que componen el lexicón, con el fin de ganar en consistencia en el sistema.

$$H_f(P_i) = \sum_{j=0}^{n} H_c(P_i, C_j) \tag{1}$$

| Forma De Palabra | Categoría | Siguiente Categoría | Frecuencia |
|---|---|---|---|
| Volante | Adjetivo | Conjunción Coordinante | 1 |
| Volante | Adjetivo | Nombre Común | 1 |
| Volante | Nombre Común | Articulo Determinado | 11 |
| Volante | Nombre Común | Articulo Contracto | 1 |

**Cuadro 2.** Conocimiento heurístico para la heurística de la frecuencia en el contexto

A partir de (1) se determina que la "heurística de la frecuencia" para una categoría gramatical concreta de la palabra que se está estudiando $H_f(P_i)$ ha de corresponderse en todo momento con la suma de todas las "heurísticas de la frecuencia en el contexto", $H_c$, de esa misma categoría $P_i$, teniendo como palabra sucesora una cualquiera, la cual, puede poseer $n$ categorías gramaticales, cada una de ellas denotada por $C_j$.

### 2.4. Metodología de etiquetación automática

Al comienzo del proceso de etiquetación se comprueba si la palabra que se está etiquetando no se encuentra ya etiquetada por algún módulo anterior, como puede ser el módulo encargado del tratamiento de las excepciones. Si la palabra ya se encuentra etiquetada, se ignora y se procede con la siguiente palabra. En nuestro sistema, una palabra puede encontrarse en los siguientes estados: unívoca, ambigua y fallo en el diccionario.

La forma de actuación variará dependiendo del estado de la palabra. Si la palabra es univoca, solo posee una categoría gramatical y no hay lugar para la confusión en la etiquetación. Si la palabra es un fallo en el diccionario porque no se ha podido encontrar ninguna categoría gramatical para el lema que posee dicha palabra, deberá de insertarse como desconocida, aunque este fallo sea debido a deficiencias del diccionario y no del algoritmo de etiquetación. (Por lema se entiende toda aquella forma a la que pueden reducirse las diferentes flexiones a que una palabra da lugar. Así, el lema de "canta, cantó, cantaríamos, etc.", es cantar; mesa/mesas, tiene mesa como lema, etc.. En general, por lema se entiende cada una de las entradas de un diccionario).

Imaginemos que no conocemos el lema asociado a la palabra "sudorificación" en el lexicón. La palabra pasa al estado de fallo en el diccionario. La alternativa fácil sería diagnosticar que esta palabra posee una categoría desconocida. Pero otra alternativa consistiría en extraer información complementaria -si la hubiere- sobre esa palabra para determinar si se puede etiquetar, aunque sea de forma parcial. En este caso sí sería posible, porque todas las palabras que acaban en "-ción" son automáticamente "Nombre Común Masculino Singular". Esta regla constaría en el lexicon y bastaría para solventar el problema.

También es posible que la palabra se presente con el estado de ambigua. En este caso hay que seleccionar la categoría adecuada partiendo de todas las posibilidades proporcionado por el lexicón.

Como discriminante inicial se procede a preguntar al sistema si alguna vez el experto ha etiquetado esta palabra (el hecho de encontrarse en el diccionario no implica que el experto la haya etiquetado alguna vez):

$$\sum_{i=0}^{n} H_f(P_i) > 0 \tag{2}$$

siendo $n$ el número total de categorías gramaticales del lexicón para la palabra estudiada.

Si esta palabra no ha sido etiquetada anteriormente por el experto, no se cumple (2), ya que para cualquier categoría la heurística de la frecuencia será 0. En este caso no se puede obtener el conocimiento heurístico descrito anteriormente, de modo que se utiliza la frecuencia de etiquetado del lema de la palabra, que ha de ser como mínimo de 1. La fórmula aplicada es:

$$P_k \ / \ max(L(P_i)) \qquad \forall i = 0, \ldots, n \tag{3}$$

En esta expresión, definimos $L$ como una función que devuelve la frecuencia del lema asociado a una categoría gramatical $P_i$ en el lexicón y partiendo de que el lexicón posee $n$ categorías gramaticales para la palabra estudiada. De los diferentes lemas asociados a la palabra se elegirá, como mejor opción, el que posea mayor frecuencia de etiquetado $P_k$. Luego se procederá a su etiquetación.

Ahora si se da el caso en el que sí que existe información acerca de la frecuencia de etiquetado del experto, es decir que el experto si que ha etiquetado esta palabra con anterioridad y por tanto se cumple (2), entonces de entre todas las frecuencias asociadas a las posibilidades de elección se comprueba cual es la más frecuente $F_m$ (5) y si esta frecuencia constituye un porcentaje mayor de un umbral de fiabilidad $\upsilon_0$ (explicado a continuación) entonces se etiqueta con la categoría asociada a esta posibilidad que es la más frecuente:

$$F = \sum_{i=0}^{n} H_f(P_i) \tag{4}$$

Antes de proseguir, conviene recordar que el valor dado a la función $F$ es la suma de las "heurísticas de la frecuencia", $H_f$, de todas las categorías gramaticales ($n$) de la palabra estudiada.

$$F_m = max(H_f(P_i)), \quad \exists P_m \ / \ H_f(P_m) = F_m \tag{5}$$

La fórmula propuesta en (5) determina que $F_m$ es el valor más alto de la "heurística de la frecuencia"para todas las categorías gramaticales de la palabra estudiada. También concreta $P_m$ como la categoría que ha proporcionado dicho valor. En el supuesto de existir información acerca de la frecuencia de etiquetado llevado a cabo por el experto, se cumpliría (2) y de entre todas las frecuencias asociadas a las categorías gramaticales se escogerá la más frecuente $F_m$ (5). Si esta frecuencia constituye un porcentaje mayor que el umbral de fiabilidad $\upsilon_0$ establecido, entonces se etiqueta con la categoría asociada a esta posibilidad $P_m$, que es la más frecuente.

El umbral de fiabilidad es un parámetro externo del sistema y viene a representar el porcentaje de seguridad que se requiere para tomar la decisión de selección de una determinada categoría. Se ha comprobado que, como mínimo, para obtener un cierto grado de seguridad en la etiquetación, este umbral no puede ser inferior al 70 %. La toma de decisión podría formalizarse como:

IF $(F \ / \ F_m * 100) > \upsilon_0$
  Se etiqueta la palabra con el $P_i$ asociado a $F_m$ $(P_m)$
ELSE
  Se procede a otro método de decisión
ENDIF

En realidad, aunque la opción correcta sea la de etiquetar la palabra con la categoría que ha superado este umbral, sería posible también decidir etiquetarla como falsa con el fin de corregir errores de convergencia en el diccionario electrónico. Por tanto, si resulta que el experto ya ha etiquetado con anterioridad esta forma, pero no se encuentra presente en el diccionario, podemos decidir anotarla como desconocida para que el sistema la incorpore al diccionario cuando el experto la valide. La decisión dependerá de la finalidad que se desee en el procesamiento del corpus: si interesa que la aplicación discurra con rapidez, se optará por la opción primera. Si se desea obtener mayor seguridad en el etiquetado, se elegirá la opción con validación posterior.

Si en el proceso anterior no se supera el umbral de fiabilidad, se recurrirá al contexto que rodea la palabra, para determinar cuál es la categoría gramatical. Pero antes de entrar en detalles, analicemos el árbol de decisión en este estadio.

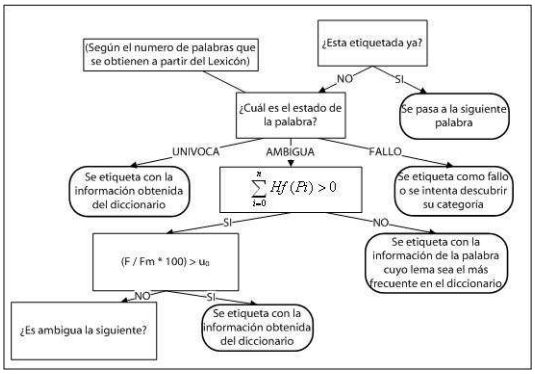



**Figura 2.** Visión parcial del árbol de regresión en el proceso de etiquetación

Si el estado de la siguiente palabra es de unívoca o de fallo en el diccionario, usaremos la "heurística de la frecuencia en el contexto", anotando como siguiente categoría gramatical la categoría de la palabra continua $C_k$. Por tanto, si tomamos (4) -(5) y cambiamos la función heurística, obtenemos (6)-(7).

$$F = \sum_{i=0}^{n} H_c(P_i, C_k) \tag{6}$$

$$F_m = max(H_c(P_i, C_k)), \quad \exists P_m \ / \ H_c(P_m, C_k) = F_m \tag{7}$$

En cuanto a la decisión sobre la etiqueta a asignar, dependerá de si la opción más frecuente, $F_m$, supera respecto al total $F$ un porcentaje mayor que un segundo umbral de confianza $\upsilon_1$. Si supera dicho umbral, esta opción será la elegida y se procederá a su etiquetado. En caso contrario, se procederá a otra forma de decisión (método de decisión análogo al código anteriormente expuesto).

En caso de que no se supere el umbral de confianza, se incorpora el factor estocástico en el sistema, es decir, se seleccionará una categoría dependiendo de la probabilidad dictaminada por el porcentaje de las frecuencias de etiquetado de todas las categorías aplicables a la palabra estudiada, procediendo a etiquetar la palabra con dicha categoría, seleccionada de forma aleatoria.

Si la siguiente palabra es también ambigua, se procederá de forma análoga al caso en el que no lo era, pero teniendo en cuenta cada una de las posibles categorías gramaticales que puede tomar la palabra siguiente.

En definitiva, la forma de proceder vendrá determinada por la posibilidad de que el experto hubiera o no etiquetado la palabra siguiente con anterioridad. Si el experto no ha etiquetado nunca esta palabra, se procederá a la selección aleatoria de una categoría de la palabra que esté siendo estudiada en función de las frecuencias de etiquetado del lema asociado a las posibles categorías. Teniendo en cuenta siempre que ahora estas frecuencias hay que tomarlas de la matriz de opciones resultante de las posibles categorías de la palabra estudiada y de las diferentes categorías que puede tomar la siguiente palabra.

En caso contrario, en el que existen palabras etiquetadas por el experto con esta forma, se procedería a calcular el producto matricial de los dos abanicos de posibilidades correspondientes a las categorías de la palabra que está siendo el foco de estudio y de su palabra siguiente. Una vez que poseemos esta matriz, comprobamos si de todas las opciones resultantes, existe alguna que supera un tercer umbral de confianza $\upsilon_2$. De ser así, se procederá a etiquetar la categoría de esta palabra con esta posibilidad.

$$F = \sum_{i=0}^{n} \sum_{k=0}^{m} H_c(P_i, C_k) \tag{8}$$

Las formula (8) es análoga a la descritas en (6). En ella se ha introducido una doble sumatoria para representar la matriz correspondiente a la combinación de las ambigüedades de la palabra estudiada y de la que es la sigue.

El peor de los casos ocurriría si no se cumpliera el umbral de confianza $\upsilon_2$. Entonces, de esta matriz de posibilidades se seleccionará una de ellas, de manera similar a como se hizo anteriormente, con un criterio de probabilidad dictaminado por el porcentaje de frecuencia de etiquetado de esta opción. Una vez seleccionada, se procederá a su etiquetado.

La rama del árbol que se ocupa de las decisiones en el caso de que la palabra siguiente a la estudiada sea también ambigua tendría el este aspecto Fig. 3.

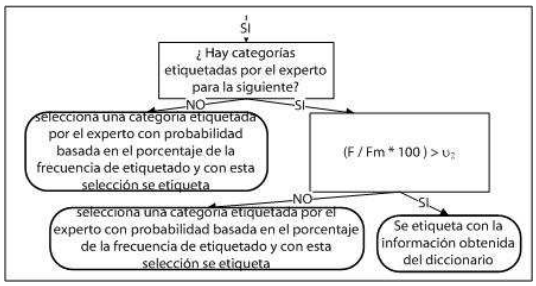



**Figura 3.** Árbol de regresión: rama en la que la palabra siguiente es ambigua

### 2.5. Mecanismo para el aprendizaje en la etiquetación del corpus

La forma en la que evoluciona el sistema radica en el aprendizaje que éste lleva a cabo a la vez que el experto realiza la etiquetación manual o la valida. El sistema mejora continuadamente su proceso de etiquetación, lo que conlleva un aumento de su fiabilidad con el transcurrir del tiempo. Incluso permite la adaptación a las futuras tendencias de la lengua: si una determinada palabra cae en desuso, este hecho será detectado por el sistema y la frecuencia decreciente afectará a la toma de decisiones en la etiquetación.

Si el experto determina que una palabra pertenece a una categoría gramatical en un contexto oracional determinado, en tal caso se producirá un aprendizaje asistido por el experto. La forma de realizar este aprendizaje consiste en modificar las heurísticas aumentando en una unidad la frecuencia de etiquetado, lo que hace que para la categoría gramatical $P_i$ ya etiquetada se produzca que:

$$\begin{aligned} H_f(P_i) &= H_f(P_i) + 1 \\ H_c(P_i, C_j) &= H_c(P_i, C_j) + 1 \\ L(P_i) &= L(P_i) + 1 \end{aligned} \tag{9}$$

En (9) se expresa el aumento de la frecuencia en las diferentes heurísticas. En futuros procedimientos de etiquetación en modo automático se tendrán en cuenta los nuevos valores resultantes de aplicar (9) a la hora de tomar decisiones. Esto conlleva que la información que adquiere e incorpora el sistema se ralentice, ya que éste solo aprende a partir de los datos que introduce el experto. Por tanto, el sistema por sí solo no puede crear conexiones semánticas automáticas. Sin embargo, la información que se incorpora es muy precisa, ya que ha sido el propio experto el que la introducido y/o validado. Este mecanismo de aprendizaje permite incrementar la fiabilidad del etiquetado con el paso del tiempo.

## 3. Test realizado y resultados obtenidos

Las pruebas de fiabilidad en el proceso de etiquetado se han realizado comparando los resultados obtenidos de un corpus previamente etiquetado Co-NLL 2002

[7], con los resultados obtenidos con nuestro etiquetador. El corpus Co-NLL 2002 cuenta con 273,000 palabras etiquetadas morfosintácticamente.

La fuente de conocimiento de nuestro sistema o etiquetador automático se basa en el corpus Cumbre-2M. Cumbre-2M consta de unos dos millones de palabras etiquetadas a nivel morfosintáctico, de las cuales se extrajeron las categorías gramaticales de unas 23,500, que son las que integran inicialmente el diccionario electrónico. Los parámetros utilizados como umbral en la toma de decisiones han sido asignados con el mismo valor del 70 % ($\upsilon_0, \upsilon_1, \upsilon_2$). Los resultados obtenidos del test reflejan que la fiabilidad del proceso de etiquetación para el corpus de Co-NLL 2002 (no se tuvieron en cuenta los fallos del diccionario, ya que este tipo de fallos no es debido a la eficiencia del método de etiquetación sino a la calidad del lexicón que se esté utilizando) se sitúa entorno al 91, 1 %.

Este porcentaje puede parecer bajo si lo comparamos con resultados obtenidos en otros grupos de investigación, como por ejemplo con el software Relax, que dice alcanzar una fiabilidad del 97 %.El resultado no es totalmente satisfactorio porque está supeditado a la propia ambigüedad del lenguaje, que impide alcanzar el 100 % de precisión en la fuente de información que estamos utilizando.

El porcentaje de error presente en Cumbre-2M se sitúa entre un 6,1 % y un 6,7 %. Además al ser utilizado como fuente de conocimiento, esto conlleva que en el inicio del proceso evolutivo del sistema se obtengan menores porcentajes de fiabilidad. Pero no hay que olvidar que el sistema puede aprender y evolucionar y, por tanto, aumentará su tasa de fiabilidad en el momento en el que el experto empiece a intervenir, corrigiendo y validando errores.

## 4. Conclusiones

En la actualidad existen varias implementaciones que resuelven el problema de la etiquetación morfosintáctica de la lengua española proporcionando un porcentaje muy alto de fiabilidad. De todas ellas, las más fiables son las que se basan en árboles de decisión para llevar a cabo la tarea de etiquetación.

Cuando se iniciaron los primeros modelos de etiquetación no se disponía aún de corpus lingüísticos de gran tamaño susceptibles de ser utilizadas como fuente de conocimiento, en lugar de utilizar solamente un diccionario electrónico. La alternativa que ahora se ofrece es la utilización de un corpus de gran tamaño como fuente de conocimiento para obtener, por un lado, las posibles categorías gramaticales de cada una de las palabras que lo componen, y, por otro lado, obtener información aportada durante la etiquetación de dicho corpus. De esta manera es posible partir de esta información como fuente heurística de conocimiento y realizar la etiquetación de una manera más fiable.

Las pruebas realizadas con nuestro prototipo se basan en un sistema que todavía no ha aprendido del trabajo de validación del experto; únicamente posee el conocimiento obtenido del corpus Cumbre-2M. De ahí que los resultados obtenidos no sean aún plenamente satisfactorios.

El carácter diferenciador de este método es que la evolución debida a la retroalimentación del sistema durante el proceso de aprendizaje puede incrementar considerablemente el porcentaje de aciertos.

Además, conseguir una fiabilidad aceptable ya desde el principio permite que los resultados obtenidos puedan ser utilizados para emprender un etiquetado similar a otros niveles más avanzados, como podrían ser el sintáctico y el semántico. De hecho actualmente ya se utilizan los datos obtenidos en el proyecto de investigación HUM2004-00080 cuyo objetivo es el diseño de un prototipo de desambiguación automática del significado de las palabras en el lenguaje natural.

## Agradecimientos

Este trabajo viene promovido por los estudios realizados dentro del proyecto de investigación con referencia HUM2004-00080, (Análisis, tipificación y formalización de las acepciones de términos léxicos polivalentes a partir del contexto, en inglés y en español, y desarrollo de un prototipo de desambiguación automática), dirigido por el profesor Aquilino Sánchez y dentro del Grupo de Investigación LACELL, de la Universidad de Murcia.

Los autores agradecen al Ministerio de Educación y Ciencia y al Fondo Europeo de Desarrollo Regional (FEDER) por su soporte parcial, bajo el proyecto TIN2005-08404-C04-02, para el desarrollo de este trabajo.

## Referencias

1. Mc Energy, T., Wilson A.: Corpus Linguistics. Edinburgh University Press. Introductory course on corpus linguistic, based on the book (1996), disponible: http://www.ling.lanc.ac.uk/monkey/ihe/linguistics/contents.htm
2. Sánchez, A.: Diseño y elaboración de un corpus representativo del Español Actual (España e Hispanoamerica), financiado por la Editorial SGEL s.a. Universidad de Murcia (1992-1998)
3. Acebo S., Ageno A., Climent S., Farreres J., Prado L., Ribas F., Rodríguez H., Soler O.: MACO - Morphological Analyser Corpus Oriented. Universidad de Politécnica de Cataluña (1994). Información en http://www.lsi.upc.es/ñlp/
4. Adwait R.: MXPOST - A Maximum Entropy Part-Of-Speech Tagger. University of Pennsylvania (1996)
5. Acebo S., Ageno A., Climent S., Farreres J., Prado L., Ribas F., Rodríguez H., Soler O.: RELAX. Universidad de Politécnica de Cataluña (1999). Información en http://www.lsi.upc.es/ñlp/
6. Heid U.: Tree Tagger. University if Stuttgart, Institute for Romance Linguistics and the Institute for Computer Science (1997)
7. Conference on Computational Natural Language Learning (CoNLL) (2002). Disponible en: http://www.cnts.ua.ac.be/conll2002/
8. Civil M.: Criterios de etiquetación y desambiguación morfointáctica del corpus en español. Sociedad Española para el Procesamieno del Lenguaje Natural. Accesible a través de: http://www.sepln.org/
9. Daudé J.: Enlace de Jerarquías Usando el Etiquetado por Relajación. PhD. Tesis, Dep. de LSI, Universitat Politécnica de Catalunya, July (2005)
10. Sánchez A. et al.: Cumbre - Corpus lingüístico del español contemporáneo. Madrid. SGEL s.a. (1995)

# Máxima verosimilitud con dominio restringido aplicada a clasificación de textos

Jesús Andrés Ferrer[1] y Alfons Juan Císcar[1,2]

[1] Universidad Politécnica de Valencia
[2] Instituto Tecnológico de Informática

**Resumen** En el área de reconocimiento de formas y tecnologías de la percepción se suele utilizar modelos estadísticos inductivos, es decir, modelos que aprenden a partir de ejemplos. Este tipo de modelos padecen el problema del *sobreentrenamiento*, que se puede solucionar con una etapa adicional a la de estimación paramétrica, denominada *suavizado*; y que normalmente responde a métodos heurísticos. No obstante, estos heurísticos presentan ciertos inconvenientes teóricos. Proponemos un método para evitar el problema del suavizado que además se encuadra dentro de la técnica de máxima verosimilitud y sus variantes, como el algoritmo EM. Esta técnica se evaluará en la práctica con tareas de clasificación de documentos, usando para ello el clasificador naïve bayes, tanto en su versión simple como en forma de mixtura.

## 1. Introducción

En los últimos años se han desarrollado técnicas de *reconocimiento de formas y tecnologías de la percepción* para abordar problemas que en un principio parecían estar fuera del dominio de la automatización, tales como: reconocimiento de dígitos manuscritos, reconocimiento de voz, traducción automática, etc. Las técnicas utilizadas en este ámbito normalmente hacen uso de modelos inductivos estadísticos, en los que el sistema aprende a partir de un conjunto de ejemplos.

Un modelo estadístico esta formado por un conjunto de parámetros que distribuyen la probabilidad en el dominio de los posibles valores de los datos (dominio de eventos). El aprendizaje inductivo para modelos estadísticos consiste, por lo tanto, en hallar el vector paramétrico que hipotéticamente ha generado los datos. Para alcanzar dicho fin, existen varias técnicas estadísticas, siendo máxima verosimilitud (MV) la más extendida.

La técnica de estimación por máxima verosimilitud (EMV) [3] consiste en seleccionar el vector paramétrico que maximiza la función de verosimilitud de una *muestra*. Esta función presenta una importante desventaja puesto que el vector paramétrico seleccionado normalmente es demasiado específico para los datos proporcionados (la muestra), siendo incapaz, por lo tanto, de alcanzar una elevada generalización. Este problema se conoce como el problema del *sobreentrenamiento*.

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 791-803, 2006.

Con la finalidad de reducir el impacto de este sobreentrenamiento existen tanto técnicas heurísticas (o pseudo-heurísticas) como teóricas. Una de las técnicas más extendidas consiste en perturbar levemente el vector paramétrico óptimo, mediante una técnica *heurística* de forma que se fuerza la generalización del vector paramétrico de forma "artificial". Sin embargo, también existen técnicas estadísticas que en principio carecen de este problema, tales como el entrenamiento *Bayesiano* o el entrenamiento por *máxima probabilidad a posteriori* (MAP[3]).

Las 2 técnicas de estimación mencionadas definen *una distribución a priori* sobre el dominio de los parámetros. De esta forma, los parámetros pasan a ser una variable aleatoria más que debe ser marginalizada. Esto conlleva una serie de desventajas tanto prácticas como teóricas. La desventaja más importante, es que para realizar predicciones se necesita obtener un modelo estadístico libre de los parámetros, lo que requiere el cálculo de integrales multidimensionales que son sólo resolubles analíticamente para ciertas distribuciones a priori (denominadas *priors conjugadas*). En este sentido, la aproximación MAP nace como un intento para solucionar el problema de las integrales calculando el punto máximo de dichas integrales, eliminando así la desventaja de cálculo pero perdiendo exactitud y validez teórica. De cualquier modo, la elección de una prior conjugada introduce un conocimiento a priori que afecta a ambos métodos y cuyo efecto en el proceso de aprendizaje es determinante. En este sentido o se selecciona una prior no informativa mediante un *análisis de referencia* (ver [1]) lo que añade dificultades teóricas adicionales; o se selecciona la prior conjugada; o esta selección se plantea como parte del problema de modelado.

En este artículo, analizaremos una técnica que llamaremos *máxima verosimilitud con dominio restringido (en adelante MVDR)* . El objetivo de dicho procedimiento consiste en utilizar máxima verosimilitud pero sobre un subdominio de todo el dominio paramétrico. De este modo, si se pueden seleccionar a priori los vectores paramétricos problemáticos hacia los que tiende el sobreentrenamiento, estos se excluyen del dominio. De este modo, se busca el vector paramétrico máximo verosímil pero dentro del nuevo dominio evitando así el sobreentrenamiento degenerado. Con este fin, utilizaremos teoría de optimización con restricciones y concretamente las condiciones de Kunh-Tucker [2].

Para analizar las implicaciones prácticas de este método, hemos aplicado dicha técnica al problema de clasificación de documentos mediante técnicas naïve bayes multinomial (ver [5,7]). En concreto hemos utilizado un modelo multinomial simple y un modelo de mixturas de multinomiales.

En la sección 2 introducimos el problema de la clasificación de documentos así como el suavizado. En la sección 3 se propone la técnica formal de suavizado de MVDR. En la sección 4 se introduce el modelo de mixturas. Los experimentos están concentrados en la sección 5. Finalmente, en la sección 6 comentamos las conclusiones que se pueden extraer.

[3] del inglés "Maximum A posteriry Probability"

## 2. Clasificación de textos

Dado un conjunto de categorías ($\{1, \ldots, C\}$) y dado un texto ($\mathbf{w} = w_1 \ldots w_l$) perteneciente a una categoría desconocida $c$, el problema de clasificación de textos consiste en determinar la categoría $c$ de dicho texto $\mathbf{w}$. Para ello utilizamos el clasificador de Bayes:

$$\hat{c}(\mathbf{w}) = \arg\max_c \, \mathrm{p}(c)\, p(\mathbf{w} \,|\, c) \tag{1}$$

Realizando la asunción naïve bayes según la cual la probabilidad de una palabra en un documento es independiente de su posición y contexto; cada documento se puede representar como un vector de conteo $\mathbf{x}$. Así pues, el $d$-ésimo elemento del vector $(x_d)$ representa el número de veces que aparece la $d$-ésima palabra del vocabulario en el texto $\mathbf{w}$. Por lo tanto, la probabilidad del vector $\mathbf{x}$ condicionada a la categoría $c$ sigue una distribución multinomial con vector de prototipos $\boldsymbol{\theta}_c$:

$$\mathrm{p}(\mathbf{x} \,|\, c) = p(l \,|\, c) \frac{l!}{\prod_d x_d!} \prod_{d=1}^{D} \theta_{cd}^{x_d} \tag{2}$$

Donde $p(l \,|\, c)$ es la probabilidad a priori sobre la longitud del documento. Normalmente este término no suele ser determinante a la hora de calcular la clase del documento por ello se asume una longitud de documento fija.

Con todas las hipótesis anteriores, el clasificador de Bayes es:

$$\hat{c}(\mathbf{x}) = \arg\max_c \, p(c) \prod_d \theta_{cd}^{x_d} \tag{3}$$

Los parámetros del modelo son: $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_c, \mathrm{p}(c) \,:\, \forall c \in \{1, \ldots, C\}\}$.

Conocida una muestra $\bar{\mathbf{X}} = \{(\mathbf{x}_n, c_n)\}_{n=1}^{N}$, la función a optimizar es la verosimilitud logarítmica[4]:

$$\begin{aligned} LL(\boldsymbol{\Theta}; \bar{X}) &= \sum_n \log(p(\mathbf{x}_n, c_n)) \\ &= \sum_c N(c) \log(\mathrm{p}(c)) + \sum_d x_{+d}^c \log(\theta_{cd}) + \mathrm{cte} \end{aligned} \tag{4}$$

Donde $N(c)$ es el número de documentos de la categoría o clase $c$ en la muestra $\bar{X}$; y $x_{+d}^c$ es el número de ocurrencias de la palabra $d$ en los textos de la categoría $c$, i.e., $x_{+d}^c = \sum_{n:c=c_n} x_{nd}^c$.

El estimador máximo verosímil maximiza la verosimilitud logarítmica (4) en el soporte de $\boldsymbol{\Theta}$. Esto se puede expresar en términos del siguiente problema de optimización:

$$\hat{\boldsymbol{\Theta}} = \arg\max_{\boldsymbol{\Theta}} \{LL(\boldsymbol{\Theta}; \bar{X})\}, \text{ s.a } \sum_c p(c) = 1 \;,\; \forall c \,:\, \sum_d \theta_{cd} = 1 \tag{5}$$

[4] Puesto que es un proceso de maximización y la función logaritmo es creciente es equivalente a maximizar la función de verosimilitud

Resolviendo este problema de optimización se obtiene el estimador máximo verosímil:

$$\hat{\mathrm{p}}(d \,|\, c) = \frac{x^c_{+d}}{x^c_{++}} \qquad \hat{\mathrm{p}}(c) = \frac{N(c)}{\sum_{c'} N(c')} \tag{6}$$

donde $x^c_{++}$ es el número de apariciones de palabras en todos los documentos de la categoría $c$, i.e., $x^c_{++} = \sum_d x^c_{+d}$.

En la ecuación (6) se observa el problema del sobreentrenamiento. Concretamente, si $x^c_{+d} = 0$ para alguna categoría $c$, entonces, $\hat{\theta}_{cd} = 0$, y por lo tanto la probabilidad para un texto en el que aparezca una palabra no observada en la muestra para dicha categoría tiene una probabilidad nula de pertenencia a dicha categoría; lo que se resume en una pérdida de generalidad.

Normalmente, a partir de la fórmula de estimación (6) se obtiene una fórmula *suavizada*. Por ejemplo, se suele utilizar las técnicas de *suavizado de Laplace* y de *Descuento Absoluto*. La primera técnica añade un pseudo-contador $\epsilon > 0$ a cada contador de cada dimensión y clase. La segunda técnica consiste en un conjunto de técnicas que acumulan una masa de probabilidad descontando un pseudo-contador $\epsilon > 0$ a todos los contadores y redistribuyendo luego esa masa entre los contadores atendiendo a una distribución $\beta$ que normalmente o bien distribuye la masa en todas las dimensiones (*interpolación*) o bien sobre todos los contadores con un valor menor que el parámetro de suavizado $\epsilon$ (*back-off*). En [5] y [7] se analizan en detalle los diferentes métodos de suavizado.

## 3. Modelo multinomial con dominio restringido

En esta sección se analiza la técnica de estimación de máxima verosimilitud con dominio restringido (MVDR) para el caso multinomial. La técnica se analiza como un problema de estimación para una distribución multinomial con un vector de prototipos $\boldsymbol{\theta}_d$. La extensión al problema de clasificación es poco más que directa, basta con repetir el proceso con cada clase. De forma análoga asumiremos que la muestra $\bar{\mathbf{X}}$ sólo contiene muestras de dicha distribución multinomial. Por último, dado un conjunto ($\mathbb{A} = \{a_1, \ldots, a_A\}$) de índices de palabras del vocabulario ($a_i \in \{1, \ldots, D\}$) denotaremos por $x_{+\mathbb{A}}$ a la suma de los contadores de estas palabras, i.e., $x_{+\mathbb{A}} = \sum_{a \in \mathbb{A}} x_{+a}$.

El principal problema de sobreentrenamiento en el caso multinomial se produce en las dimensiones en las que han habido pocas observaciones o incluso ninguna observación . Este problema se puede resolver asegurando que cada dimensión del espacio paramétrico sea mayor o igual que un cierto valor $\epsilon_d$. Es decir, resolviendo el problema de maximización de la verosimilitud pero con un *espacio restringido de parámetros* definido por:

$$\forall d : 1 \leq d \leq D \;:\; 1 - \epsilon_d \geq \theta_d \geq \epsilon_d \text{ , donde } \sum_{d=1}^{D} \epsilon_d \leq 1 \tag{7}$$

Restringiendo el dominio paramétrico de $\boldsymbol{\theta}$ a aquellos vectores que cumplen las restricciones (7) se obtiene el siguiente problema de optimización:

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta}; \bar{\mathbf{X}}) \ , \ \text{sujeto a:}$$
$$\sum_d \theta_d = 1 \tag{8}$$
$$\forall d \ : \ 1 \leq d \leq D \ : \ \theta_d \geq \epsilon_d$$

**Teorema 1 (Caracterización)** *Sea el problema de optimización de la ecuación* (8) *y definidos los dos conjuntos:* $\mathbb{E}(\{e_1, \ldots, e_E\})$ *y* $\mathbb{I}$ $(\{i_1, \ldots, i_I\})$*; tales que :*

$$\forall e \in \mathbb{E} \ : \ \frac{x_{+e}}{x_{+\mathbb{I}}}\Big(1 - \epsilon_{\mathbb{E}}\Big) \leq \epsilon_e \qquad \forall i \in \mathbb{I} \ : \ \frac{x_{+i}}{x_{+\mathbb{I}}}\Big(1 - \epsilon_{\mathbb{E}}\Big) > \epsilon_i \tag{9}$$

*donde* $\{1, \ldots, D\} = \mathbb{E} \cup \mathbb{I}$*, y* $\mathbb{E} \cap \mathbb{I} = \emptyset$*. Un punto máximo de la función es:*

$$\forall i \in \mathbb{I} \qquad \hat{\theta}_i = \tfrac{x_{+i}}{x_{+\mathbb{I}}}\Big(1 - \epsilon_{\mathbb{E}}\Big) \tag{10}$$
$$\forall e \in \mathbb{E} \qquad \hat{\theta}_e = \epsilon_e \tag{11}$$

*Donde* $x_{+d} = \sum_{n=1}^{N} x_{nd}$*,* $\epsilon_{\mathbb{A}} = \sum_{a \in \mathbb{A}} \epsilon_a$*, y* $x_{+\mathbb{A}} = \sum_{a \in \mathbb{A}} \sum_{n=1}^{N} x_{na}$*; asumiendo que* $\mathbb{A} \subseteq \mathbb{D}$*, y* $d \in \mathbb{D} = \{1, \ldots, D\}$ □

El teorema 1 proporciona una caracterización de la solución. La demostración se obtiene con una simple aplicación de las condiciones de Kunh-Tucker al programa matemático definido en la ecuación (8). Se puede observar que para el caso en que $\mathbb{E} = \emptyset$ todas las dimensiones de $\boldsymbol{\theta}$ se corresponden con el estimador máximo verosímil.

El método constructivo de la solución lo proporciona el Algoritmo 3.1. Inicialmente se introducen todos los índices en un conjunto $\mathbb{T}$, conjunto que aspira a cumplir la restricción del conjunto $\mathbb{I}$ del teorema 1. Una vez inicializado se recorre en busca de algún elemento que no cumpla la condición, se extrae del conjunto $\mathbb{T}$ y se introduce en el conjunto $\mathbb{E}$, actualizado las probabilidades para que cumplan la nueva hipótesis. De esta manera al final del proceso se obtienen dos conjuntos $\mathbb{E}$ y $\mathbb{T}$ que cumplirán las condiciones del teorema 1. Se puede demostrar (ver apéndice A) que el algoritmo siempre produce una solución óptima (si las restricciones permiten que exista).

Es importante reseñar que los métodos de suavizado mencionados en la sección anterior (ver [5]) se pueden utilizar para conseguir que después de suavizar el vector de parámetros cumpla las restricciones. No obstante, como se ha analizado en esta sección no es la solución máximo verosímil.

Finalmente, aquí hemos analizado cómo hallar una solución dado un vector de restricciones $\epsilon$; sin embargo, la forma de elegir $\epsilon_d$ para cada clase y palabra, queda delegada a un dominio heurístico. En el resto del documento trabajaremos con un valor $\epsilon_d = \varepsilon / D$. En un futuro, este vector se podría obtener con técnicas de optimización funcional.

**Algoritmo 3.1** Algoritmo para obtener el estimador MVDR

**Input:** Una muestra $\mathbf{X}$, y $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_D)^T$
**Output:** El estimador máximo verosímil con dominio restringido (MVDR) de $\boldsymbol{\theta}$ si este existe

$\mathbb{E} = \emptyset$; $\mathbb{T} = \mathbb{D}$; $\epsilon_{\mathbb{E}} = 0$; $x_{+\mathbb{T}} = \sum_{t\in\mathbb{T}} x_{+t}$;
$\forall d \in \mathbb{D}$
  **if**$\left(\frac{x_{+d}}{x_{+\mathbb{D}}} \leq \epsilon_d\right)$
  {$x_{+\mathbb{T}} = x_{+\mathbb{T}} - x_{+d}$; $\mathbb{T} = \mathbb{T} - \{d\}$; $\mathbb{E} = \mathbb{E} \cup \{d\}$; $\hat{\theta}_d = \epsilon_d$; $\epsilon_{\mathbb{E}} = \epsilon_{\mathbb{E}} + \epsilon_d$;}
f$\forall$
`has_changed=true;`
`while(has_changed)`
`{ has_changed=false;`
  $\forall t \in \mathbb{T}$
    **if**$\left(\frac{x_{+t}}{x_{+\mathbb{T}}}(1-\epsilon_{\mathbb{E}}) \leq \epsilon_t\right)$ {$x_{+\mathbb{T}} = x_{+\mathbb{T}} - x_{+t}$; $\mathbb{T} = \mathbb{T} - \{t\}$; $\mathbb{E} = \mathbb{E} \cup \{t\}$;
      $\hat{\theta}_t = \epsilon_t$; $\epsilon_{\mathbb{E}} = \epsilon_{\mathbb{E}} + \epsilon_t$; `has_changed=true; }`
  f$\forall$
$\forall t \in \mathbb{T}$ : $\hat{\theta}_t = \frac{x_{+t}}{x_{+\mathbb{T}}}(1-\epsilon_{\mathbb{E}})$ *f*$\forall$
`}`

## 4. Modelo de mixturas multinomiales

Normalmente, los textos que se intentan clasificar automáticamente, están compuestos por una mezcla de tópicos. Por ello parece sensato (y en la práctica lo es) modelizar la probabilidad condicional del vector de contadores del texto como una mixtura de multinomiales. Nuevamente por simplificar notación se asumirá que sólo disponemos de una clase fija, siendo fácil extender su uso para cada categoría sin más que reproducir la estimación para cada categoría $c$. El modelo incompleto de mixturas es el de la siguiente ecuación como:

$$\mathrm{p}(\mathbf{x}) = \sum_{i=1}^{I} \mathrm{p}(i)\,\mathrm{p}(\mathbf{x}\,|\,\theta_i) \tag{12}$$

donde $\mathrm{p}(\mathbf{x}\,|\,\theta_i)$ sigue una distribución multinomial de longitud *fija* $l$, y vector de probabilidades $\theta_i$.

Para estimar las probabilidades de este modelo usamos el algoritmo EM (propuesto en [4]) con una variable oculta por cada elemento de la muestra $x_n$ que indica a que tópico pertenece ($\mathbf{z}_n$). Este algoritmo se basa en un proceso iterativo en el que alternativamente se van ejecutando dos pasos: el paso E y el paso M. En el paso E se calcula una estimación de la variable oculta $\mathbf{z}_n$, concretamente en la iteración $k$:

$$z_{ni}^{(k)} = \frac{p^{(k)}(i)\prod_d \theta_{id}^{(k)^{x_{nd}}}}{\sum_{i'=1}^{I} p^{(k)}(i')\prod_d \theta_{i'd}^{(k)^{x_{nd}}}} \tag{13}$$

En el paso M (de maximización) se usan los valores calculados en el paso E para maximizar la verosimilitud logarítmica completa obteniendo:

$$p^{(k+1)}(i) = \frac{\sum_n z_{ni}^{(k+1)}}{N} \tag{14}$$

$$\theta_{id}^{(k+1)} = \frac{\sum_{n=1}^{N} z_{ni}^{(k+1)} x_{nd}}{\sum_{n=1}^{N} z_{ni}^{(k+1)} \sum_{d=1}^{D} x_{nd}} \tag{15}$$

En el paso M se resuelve un problema de maximización en el que se pueden introducir las restricciones de la ecuación (7). Análogamente al teorema 1 , se puede obtener un teorema equivalente y un algoritmo similar al algoritmo 3.1, exceptuando que en este caso hay un conjunto de índices para cada componente de la mixtura. Concretamente, bastaría sustituir las expresiones $x_{+d}$ por $x_{+d}^{(i)} = \sum_{n=1}^{N} z_{ni} x_{nd}$; y $x_{+\mathbb{A}}$ por $x_{+\mathbb{A}}^{(i)} = \sum_{a \in \mathbb{A}} \sum_{n=1}^{N} z_{ni} x_{na}$ asumiendo que $\mathbb{A} \subseteq \mathbb{D}$.

Este método presenta una propiedad teórica muy interesante: garantiza la convergencia del EM si se aplica en cada paso de maximización. Normalmente, el suavizado consiste en entrenar el modelo sin suavizado y una vez alcanzada la convergencia del algoritmo iterativo suavizar el vector de prototipos final; o en su defecto realizar el suavizado en cada iteración con la esperanza de que el algoritmo converja. Por lo tanto, este método de algún modo formaliza el proceso de suavizado con el algoritmo EM.

## 5. Experimentos

Teniendo en mente que nuestro objetivo es comprobar las diferencias entre los modelos de suavizado clásicos y la técnica analizada en el documento; se han realizado dos tipos de experimentos: con datos simulados y con datos reales.

### 5.1. Datos simulados

Con el objetivo de analizar el comportamiento de MVDR en relación a las técnicas clásicas se han generado datos simulados de una multinomial en los que el $d$-ésimo elemento del prototipo sigue una de las siguientes distribuciones:

- *Datos generados según la ley de Zipf:* esta ley modeliza la probabilidad de aparición de la $d$-ésima palabra más probable sobre un vocabulario:

$$\theta_d = \frac{1/d^2}{\sum_{d'} 1/d'}$$

- *Un vector escalonado:* en este caso la probabilidad del $\theta_d$ es proporcional a 3,2,1 o 0 según $d$ pertenezca a los conjuntos $[0, \frac{D}{4}]$, $]\frac{D}{4}, \frac{2D}{4}]$ ,$]\frac{2D}{4}, \frac{3D}{4}]$ o $]\frac{3D}{4}, D]$ respectivamente.

En todos los casos se ha calculado el promedio de los resultados repitiendo los experimentos 100 veces. Así mismo, se han muestreado los distintos valores del parámetro de suavizado, para los suavizados siguientes:

- El suavizado clásico de interpolación lineal: $\tilde{\theta}_d = (1-\varepsilon)\hat{\theta}_d + \frac{\varepsilon}{D}$
- El nuevo método estimador MVDR obtenido mediante el algoritmo (3.1) con un vector de suavizado: $\epsilon = \varepsilon/\mathbf{D}$

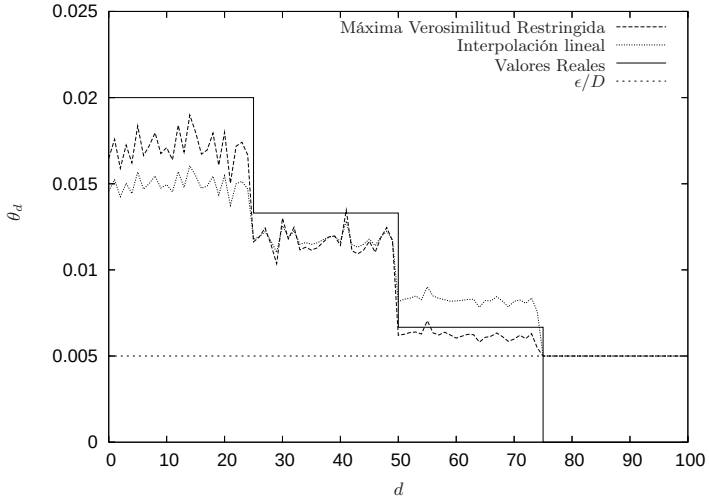



**Figura 1.** Resultados de simulación con distribución de 3 escalones con un espacio de parámetros de 100 dimensiones y con un suavizado de $\varepsilon = 0{,}5$. El eje horizontal representa las dimensiones $d$, y el eje vertical el valor del parámetro de la correspondiente dimensión, es decir, $\theta_d$.

Las implicaciones prácticas se pueden resumir en dos. En primer lugar, para valores pequeños de $\varepsilon$, ambos suavizados estiman un vector paramétrico casi idéntico. Por otra parte, conforme estos valores crecen la interpolación lineal deforma el espacio, mientras que la versión restringida de MV se ajusta mucho mejor a los parámetros (figura 1). La deformación comentada es directamente proporcional al número de dimensiones que se excluyen, como en el caso de la figura 2. Además, esta diferencia se vuelve significativa para valores elevados del suavizado, cuando se excluyen dimensiones con información bien estimada y posiblemente discriminativa.

### 5.2. Datos reales

En esta sección se analizan los resultados experimentales con 2 corpus: 20 Newsgroup y EuTrans-I. En ellos se comparan los modelos de suavizado clásicos con los modelos de interpolación lineal y máxima verosimilitud con dominio restringido. Además se realiza un estudio experimental del comportamiento de las mixturas.

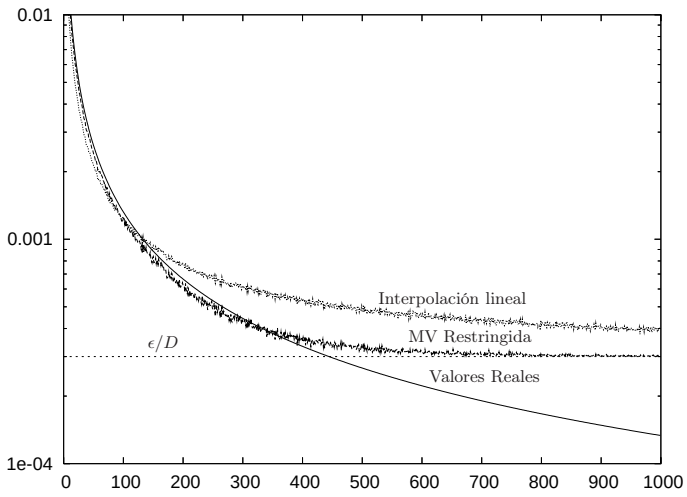



**Figura 2.** Resultados de simulación según la ley de Zipf con un espacio de parámetros de 1000 dimensiones y con un suavizado de $\varepsilon = 0{,}3$. El eje horizontal representa las dimensiones $d$, y el eje vertical la probabilidad del parámetro $\theta_d$ en escala logarítmica.

El 20 Newsgroup es un corpus extraído de un grupo de noticias que consta de 20 clases (grupos) distintos, como su propio nombre indica. El corpus consta de 20K texto (1K por clase) un vocabulario de 91K palabras con un total de $3,5$M apariciones de palabras. Por lo tanto, se trata de un problema complejo.

El segundo corpus es EuTrans-I definido en [6]. Se trata de un corpus bilingüe de Español e Inglés, en un contexto restringido. Refleja conversaciones comunes en el mostrador de un hotel, que se han generado semi-sintéticamente a partir de un pequeño conjunto de frases de referencia obtenidas de pequeños diccionarios de viaje. Se usaron 4 pseudolocutores para generar cada frase. Es muy importante, por lo tanto, tener presente el hecho de que cada persona genera frases de distintas categorías, que se solapan entre locutores. El objetivo es detectar que locutor a generado una frase dada. El corpus consta de 8K frases bilingües con 2K frases por locutor. El tamaño del vocabulario es de 679 palabras en Español y de 503 palabras en Inglés, con un total de apariciones de 86K palabras en Español y de 80K en Inglés.

En la figura 3 se puede observar como con sólo eliminar los contadores nulos, se obtienen buenos resultados con interpolación lineal y la maximización restringida. Pese a que las diferencias experimentales entre estos dos métodos en escala logarítmica es pequeña o inexistente, a medida que se aumenta el umbral de suavizado van empeorando ambos método; aumentando la diferencia entre ambos siendo en estos casos mejor el modelo interpolado. Este suceso se debe a que MVDR pierde información discriminativa conforme aumenta el valor de suavizado ($\varepsilon$), mientras que el suavizado interpolado pierde verosimilitud en el

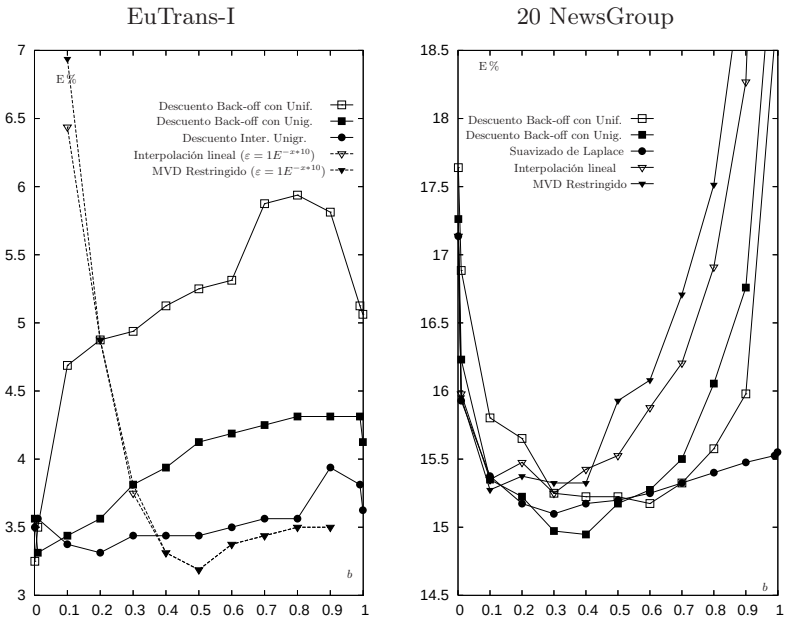



**Figura 3.** Resultados obtenidos con la tarea de EuTrans-I y 20 Newsgroup con MVDR, y otras técnicas de suavizado. La interpolación lineal y la MVDR están dibujadas en escala logarítmica en el caso de EuTrans-I. $b$ es el parámetro de suavizado.

espacio paramétrico, pero no pierde poder discriminativo. No obstante, pese a ello MVDR obtiene resultados similares. En este sentido, es importante resaltar que los modelos de unigrama aportan mucha más información que los modelos uniformes y el hecho de que obtengan mejores resultados no desvirtúa la técnica. De hecho, la selección de un modelo uniforme para el vector $\epsilon$ se debe a motivos de simplicidad y análisis. Por lo tanto, los resultados comparables con MVDR son aquellos obtenidos con los modelos uniformes.

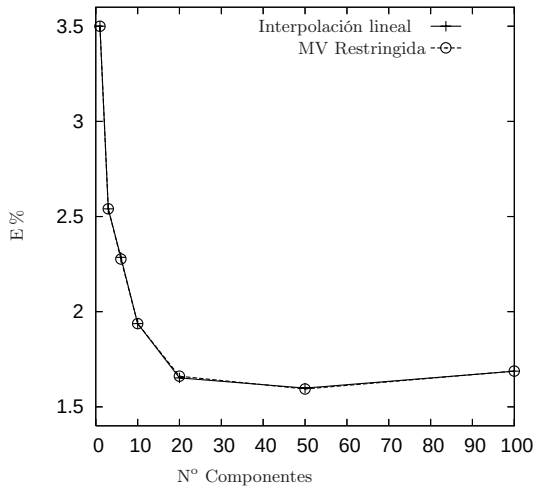



**Figura 4.** Resultados obtenidos en la tarea Eutrans-I en inglés con máxima verosimilitud restringida e interpolación lineal, incrementando el número de componentes del clasificador de mixturas multinomiales.

La figura 4 muestra el comportamiento experimental de las mixturas. Se puede observar que al utilizar las mixturas el algoritmo funciona mejor conforme aumenta el número de mixturas hasta llegar a un punto a partir del cual empeora el comportamiento por falta de muestras. Además, las diferencias entre ambos suavizados en este caso son inapreciables, puesto que se ha utilizado el mejor suavizado de la gráfica 3, que es muy pequeño $1E^{-8}$.

## 6. Conclusiones

En este artículo analiza una técnica con muy buenas propiedades teóricas dentro del marco de la estimación por máxima verosimilitud y sus variantes. Así mismo, se ha demostrado tanto en la práctica como en teoría el buen funcionamiento de dicha técnica obteniendo prestaciones similares o mejores que los métodos

clásicos de suavizado, pero con un sólido marco formal. Dentro de este marco se incluye el hecho de que este tipo de suavizados asegura la convergencia del algoritmo EM.

## Agradecimientos

Este trabajo ha sido respaldado por "La Conselleria d'Empresa Universitat i Ciència" de la "Generalitat Valenciana" bajo la beca CTBPRA/2005/004 y parcialmente por la Comisión Interministerial de Ciencia y Tecnología española (CICYT) bajo el proyecto ITEFTE (TIC2003-08681-C02-02).

## Referencias

1. José M. Bernardo and Adrien F.M. Smith Wiley. *Bayesian Theory*. Wiley, sep 2001.
2. S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
3. Morris H. DeGroot. *Probability and Statistics*. Addison-Wesley, Massachusetts, 2 edition, 1986.
4. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statist. Soc. Ser. B*, 39(1):1–22, 1977.
5. Alfons Juan and Hermann Ney. Reversing and Smoothing the Multinomial Naive Bayes Text Classifier. In *Proc. of the 2nd Int. Workshop on Pattern Recognition in Information Systems (PRIS 2002)*, pages 200–212, Alacant (Spain), April 2002.
6. Enrique Vidal et al. Final report of Esprit research project 30268 (EuTrans): Example-based language translation systems, deliverable D0.1c, September 2000.
7. David Vilar, Hermann Ney, Alfons Juan, and Enrique Vidal. Effect of fature smoothing methods in text classification tasks. In *Proc. of the 2nd Int. Workshop on Pattern Recognition in Information Systems (PRIS 2004)*, 2004.

## A. Demostración del algoritmo 3.1

**Lema 1** *Bajo las condiciones del teorema 1, sea d un índice ($d \in \{1,\ldots,D\}$) tal que*

$$\frac{x_{+d}}{x_{+\mathbb{D}}} \leq \epsilon_d, \tag{16}$$

*entonces $d \in \mathbb{E}$ y $\hat{\theta}_d = \epsilon_d$.*

**Demostración**
La siguiente ecuación es cierta:

$$1 = (1-\epsilon_{\mathbb{E}}) + \epsilon_{\mathbb{E}} \geq (1-\epsilon_{\mathbb{E}}) + \frac{x_{+\mathbb{E}}}{x_{+\mathbb{I}}}(1-\epsilon_{\mathbb{E}}) = \left(1 + \frac{x_{+\mathbb{E}}}{x_{+\mathbb{I}}}\right)(1-\epsilon_{\mathbb{E}}) \tag{17}$$

Debido a que los $\theta_e$ de $e \in \mathbb{E}$ cumplen la propiedad de la ecuación (9). A partir de la desigualdad de la ecuación (17) podemos demostrar la siguiente propiedad:

$$\left(\frac{x_{+\mathbb{I}} + x_{+\mathbb{E}}}{x_{+\mathbb{I}}}\right)(1-\epsilon_{\mathbb{E}}) \leq 1 \qquad \Rightarrow \qquad \left(\frac{x_{+d}}{x_{+\mathbb{I}}}\right)(1-\epsilon_{\mathbb{E}}) \leq \frac{x_{+d}}{x_{+\mathbb{D}}} \tag{18}$$

A partir de la hipótesis del teorema 1 (ecuación (18)) y la propiedad de la ecuación (16):

$$\frac{x_{+d}}{\sum_{i\in\mathbb{I}} x_{+i}} \left(1 - \sum_{e\in\mathbb{E}} \epsilon_e\right) \leq \epsilon_d$$

Con lo que $d$ cumple la propiedad para estar dentro del conjunto $\mathbb{E}$. □

**Lema 2** *Sea una partición de $\mathbb{D}$: $\mathbb{E}$ y $\mathbb{T}$, tal que:*

$$\mathbb{E} = \{e_1, \ldots, e_E\} \ , \qquad \forall e \in \mathbb{E} \ : \quad \frac{x_{+e}}{x_{+\mathbb{T}}}\Big(1 - \epsilon_{\mathbb{E}}\Big) \leq \epsilon_e \tag{19}$$

*y sea $t$ un elemento del conjunto $\mathbb{T}$ ($t \in \mathbb{T}$), tal que:*

$$\frac{x_{+t}}{x_{+\mathbb{T}}} (1 - \epsilon_{\mathbb{E}}) \leq \epsilon_t, \tag{20}$$

*entonces se puede definir una nueva partición $\mathbb{E} \cup \{t\}$ y $\mathbb{T} - \{t\}$, con las propiedades de $\mathbb{E}$ y $\mathbb{T}$ respectivamente.*

**Demostración**
A partir de la ecuación (19), se puede llegar a :

$$1 - \frac{x_{+t}}{x_{+\mathbb{T}}} \geq 1 - \frac{\epsilon_t}{(1 - \epsilon_{\mathbb{E}})} \qquad \Leftrightarrow \qquad \frac{1}{x_{+\mathbb{T}}} (1 - \epsilon_{\mathbb{E}}) \geq \frac{1}{x_{+\mathbb{T}} - x_{+t}} (1 - \epsilon_{\mathbb{E}} - \epsilon_t)$$

Multiplicando una de las desigualdades anteriores por $x_{+e}$ a ambos lados para cualquier valor índice $e$ en $\mathbb{E}$, se obtiene:

$$\frac{x_{+e}}{x_{+\mathbb{T}}} (1 - \epsilon_{\mathbb{E}}) \geq \frac{x_{+e}}{x_{+\mathbb{T}} - x_{+t}} (1 - \epsilon_{\mathbb{E}} - \epsilon_t) = \frac{x_{+e}}{x_{+\mathbb{T}'}} (1 - \epsilon_{\mathbb{E}'}) \tag{21}$$

Uniendo las ecuaciones (21) y (19); y con la definición de $\mathbb{E}'$, en mente, se demuestra fácilmente el lema. □

**Lema 3** *Bajo las condiciones del lema (2) si: $\forall t \in \mathbb{T} \ : \ \frac{x_{+t}}{x_{+\mathbb{T}}} (1 - \epsilon_{\mathbb{E}}) > \epsilon_t$, Entonces $\mathbb{I} = \mathbb{T}$ y $\mathbb{E} = \mathbb{E}$ es la solución MVDR definida en el teorema 1.*

La corrección del algoritmo 3.1 es sencillamente una concatenación de los lemas 1, 2, y 3.

# Resolución con datos lingüísticos de un problema de decisión

Mª Socorro Garcia[1], Mª Teresa Lamata[2]

[1] Dpto de Electrónica, Tecnología de Computadoras y Proyectos.
Universidad de Politécnica de Cartagena. Murcia, España
socorro.garcia@upct.es
[2] Dpto. Ciencias de la Computación e Inteligencia Artificial.
Universidad de Granada.18071- Granada. España
mtl@decsai.ugr.es

**Resumen.** La inteligencia artificial (IA) es la ciencia que enfoca su estudio a lograr la comprensión de entidades inteligentes. Es evidente que las computadoras que posean una inteligencia a nivel humano (o superior) tendrán repercusiones muy importantes en nuestra vida diaria. Y dentro de los campos de la I.A. cabe destacar la toma de decisión, donde la I.A. supone una gran ayuda. En este artículo presentamos una metodología en un problema de toma de decisión donde la información que se dispone es de tipo lingüístico. De esta manera podemos tener sistemas donde la entrada de los datos en un ordenador pueden ser de un tipo y la salida pueden ser del mismo o distinto tipo, haciendo así ordenadores más inteligentes.

## 1 Introducción

Desde que se empezaron a desarrollar las primeras computadoras digitales (años 30 y 40), se empezó a formar la idea de modelar la inteligencia humana con ellas Turing (computación), Von Neumann (máquinas autorreplicativas), Wiener [12] y Rosenblueth (cibernética). En Dartmouth, en 1956, se acuñó el término inteligencia artificial, durante una conferencia convocada por McCarthy a la cual asistieron, entre otros, Minsky [11], Newell y Simon. Debido a que la inteligencia artificial tuvo muchos padres no hay un consenso para definir este concepto, pero para nuestros fines podemos decir que la inteligencia artificial se encarga de modelar la inteligencia humana en sistemas computacionales.

Desde entonces ha habido una explosión en las investigaciones relativas a "hacer que la máquina piense". Primero se desarrollaron técnicas para la solución de los problemas de razonamiento lógico, visión y comprensión del lenguaje, entre otras. Pero pronto quedó claro que con las técnicas y herramientas que se utilizaban no se iba a poder modelar la inteligencia humana completamente. Entonces se recurrió a técnicas alternativas dentro de la inteligencia artificial como la lógica difusa y el soft computing Zadeh [16], la teoría de la posibilidad, inteligencia artificial distribuida (agentes inteligentes), redes neuronales, sistemas adaptativos y muchas técnicas híbridas (redes neurodifusas, agentes expertos, etcétera).

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 804-814, 2006.

Los sistemas de inteligencia artificial incluyen a las personas, los procedimientos, el hardware y software, los datos y los conocimientos necesarios para desarrollar sistemas, y máquinas de computación que presenten características de inteligencia. El objetivo del desarrollo de sistemas de IA contemporáneos no es el reemplazo completo de la toma de decisiones de los humanos, pero sí duplicarlas para ciertos tipos de problemas bien definidos.

Se define la inteligencia artificial como aquella inteligencia exhibida por artefactos creados por humanos (es decir, artificial). A menudo se aplica hipotéticamente a los computadores. El nombre también se usa para referirse al campo de la investigación científica que intenta acercarse a la creación de tales sistemas.

En el análisis de decisión multicriterio Luce y Raiffa [10], Lootsma [9] y Triantaphyllou [15], se evalúan un número de alternativas que se comparan utilizando varios criterios. El objetivo es proporcionar un soporte a los decisores en el proceso de elegir entre las diversas alternativas. En otras palabras, las técnicas ayudan al decisor a articular sus preferencias cuando el entorno de decisión es complejo. Un requisito previo en todos estos métodos es que el decisor sea capaz de proporcionar la información en la forma más precisa posible.

La mayoría de las veces, el decisor no es capaz de definir de manera estricta la importancia de los criterios y/o la bondad de las alternativas con respecto a cada criterio. En estas situaciones utilizamos medidas o cantidades que no son exactas pero sí aproximadas y que se ajustan a la realidad. En general para el decisor es más sencillo cuando evalúa sus juicios mediante términos lingüísticos. En estos casos el concepto de número difuso es más adecuado que el de número real

El propósito de este artículo es presentar un caso real de toma de decisión donde debido a la confidencialidad de la información y donde por las características de la empresa los datos que se pueden manejar sólo son lingüísticos, con lo que implementamos una metodología para la ayuda a la decisión en un problema de mantenimiento donde trabajaremos solamente con información de tipo lingüística.

## 2 Un problema de decisión en mantenimiento

Consideramos una Fábrica de Motores que está especializada en la producción, venta y mantenimiento de motores diesel de cuatro tiempos, velocidad alta y media, de aplicación naval, propulsión de carros de combate, plantas de generación eléctrica, plantas de cogeneración “llave en mano” y tracción ferroviaria.

Uno de los pasos fundamentales que hay que dar en el proceso de mantenimiento y reparación de un motor es la limpieza de cada uno de sus componentes. El proceso de comprobación y reacondicionamiento de cada componente requiere que cada pieza tenga un alto grado de limpieza, de lo contrario el proceso de reparación se encuentra con un gran escollo.

Se parte de un determinado tipo de piezas con unas dimensiones definidas y un grado de suciedad concreto que hay que eliminar.

Al ser muchas las piezas que se tratan, se van a considerar solamente las piezas más significativas. Por otra parte, los aspectos de la pieza que afectan al sistema de lavado son: el tipo de suciedad, el tamaño de la pieza, la cantidad de piezas a limpiar y

el material del que están hechas estas piezas.

**Tabla 1**: Piezas que se van a tener en cuenta

| Tipo de pieza | Tipos de suciedad | Tamaño | Peso |
|---|---|---|---|
| Pistón MAN 48/60 | Carbonilla Incrustada, Aceites y Grasas | ∅500 x 750 | 439 kq |
| Culata MAN 48/60 | Carbonilla Incrustada, Aceites y Grasas | 1053x816x500 | 1300 kg |
| Camisa MAN 48/60 | Carbonilla Incrustada, Aceites y Grasas | 1250 x ∅652 | 750 kg |
| Aro de fuego MAN 48/60 | Carbonilla Incrustada, Aceites y Grasas | 158 x ∅652 | 100 kg |
| Colectores de escape MAN 48/60 | Carbonilla Incrustada | ∅900 x 900 | 120 Kg |
| Carcasa de turbina MTU 956 16V | Carbonilla Incrustada | 500x400x600 | 150 Kg |
| Rodete de turbina MTU 956 16V | Carbonilla Incrustada | 400x400x250 | 130 Kg |
| Bloque motor MTU 956 16V | Aceites y grasas | 1200x1200x3000 | 3000 Kg |
| Caja combustible motor MTU 956 16V | Restos de combustible y pintura | 1200x500x180 | 78 Kg |

Los tipos de suciedad a los que tenemos que enfrentarnos son:

- *Carbonilla***:** La carbonilla es un aglomerado de un fino polvo de carbón y compuestos in-quemados procedentes del combustible o el aceite. Este tipo de suciedad se encuentra allí donde los gases de escape estén presentes. Es decir en Pistón, Culatas, Colectores de escape, Turbos etc... Este tipo de suciedad se encuentra siempre formando costras sólidas adheridas fuertemente a las paredes de las piezas.
- *Aceites y grasas***:** Proceden del aceite del motor y los aditivos de los diversos fluidos que evolucionan dentro de los circuitos del motor. Este tipo de suciedad está normalmente menos adherida a la pieza y no se encuentra englobando productos sólidos. De aquí en adelante a estas piezas las llamaremos piezas con suciedad leve.
- *Otros:* Dentro de este apartado se puede mencionar las capas de pintura de las piezas; que en principio no se tienen por qué considerar como suciedad; y las incrustaciones calcáreas del agua de refrigeración.

Las piezas que se van a tener en cuenta así como los tipos de suciedad que se van a considerar a la hora de elegir el sistema de limpieza son los que se muestran en la tabla 1. Se busca por tanto un sistema de limpieza que sea capaz de limpiar una gran variedad de piezas de diferentes dimensiones así como con distintos grados de suciedad. Los procesos para la limpieza de piezas son muy diversos y cada uno tiene sus ventajas y sus inconvenientes.

- *Limpieza convencional*

Dentro de este grupo se consideran las lavadoras industriales, que aplican un producto a chorro sobre la pieza, básicamente el producto aplicado es agua con un tanto por ciento de detergente alcalino no superior al 5%.

- *Limpieza química*

La limpieza química consiste en la aplicación de un producto químico agresivo para desleír la suciedad y eliminarla con un posterior aclarado. Esta limpieza química se hace por inmersión en un baño de producto químico en una pila de limpieza química, donde podría mejorarse el efecto desengrasante químico con chorros a presión y baño con turbulencias, este efecto mecánico añadido incrementa la eficacia y la capacidad de limpieza de la máquina.

Los resultados observados con este tipo de máquinas son satisfactorios para piezas de suciedad leve y moderada, para piezas de suciedad elevada como las que llevan carbonilla los resultados satisfactorios se obtienen aplicando largos tiempos de proceso.

- *Limpieza térmica*

La limpieza térmica es una alternativa a la limpieza química en los casos de piezas que lleven carbonilla. Este tipo de limpieza se fundamenta en el hecho de que los sólidos englobados en la carbonilla pueden ser disgregados si se terminan de quemar los inquemados que están presentes.

El proceso tiene tres fases. En la primera se produce un calentamiento de la pieza de modo que los in-quemados se queman y dejan de englobar los sólidos que se adherían a la pieza. En la segunda fase se produce un barrido de la superficie de la pieza en un lecho fluidizado de gránulos sólidos para que el polvo de carbón sea arrastrado de la superficie de la pieza. Finalmente el tercer paso es un proceso de eliminación de los restos sólidos de gránulos por agitación y soplado.

Este procedimiento es muy usado por los restauradores de motores diesel y muy recomendado por su eficacia. El problema de este método además de que las máquinas encontradas no se ajustan al tamaño de pieza que se están considerando y que estas tienen un coste de adquisición muy elevado, es que las elevadas temperaturas de proceso necesarias producen en las piezas defectos estructurales inaceptables en los estándares de los procesos de mantenimiento de los motores. Por lo que hace que descartemos a priori esta alternativa.

- *Limpieza mecánica*

Dentro de este grupo podemos considerar el chorreo con arena u otro material proyectable (sílice, vidrio,…), es un método que sirve para eliminar mecánicamente incrustaciones allí donde se encuentren sitios accesibles, pero en las piezas normalmente hay conductos y recovecos que o bien no son accesibles o son susceptibles de coger partículas abrasivas que después son soltadas durante el funcionamiento, produciendo averías. Este proceso es a veces imprescindible, pero no se puede considerar como un proceso completo de limpieza sino como un apoyo en ciertos casos muy concretos. Por lo tanto, esta alternativa tampoco la consideraremos a priori.

- *Limpieza por ultrasonidos*

La limpieza por ultrasonidos es en realidad un proceso mixto, ya que aprovecha la acción de un detergente convencional y la acción mecánica de unas ondas de choque y cavitaciones que se producen en el recipiente que contiene la pieza.

## 3 Aplicación del modelo difuso para la toma de decisión en la adquisición de un sistema de lavado de piezas

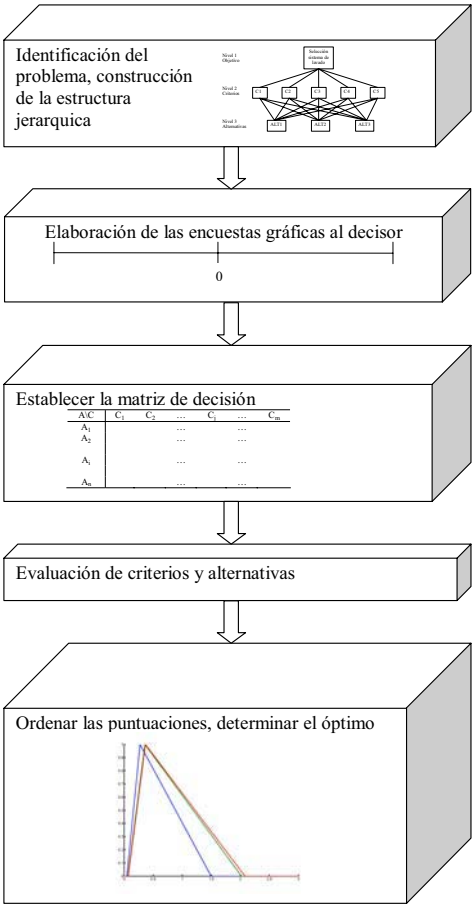



**Fig 1**: Algoritmo

La teoría de los conjuntos difusos y la lógica difusa es una herramienta que surge para poder representar datos imprecisos, por lo tanto lo aplicaremos en este contexto como una vía para representar cuantitativamente la información y poder manipular la imprecisión en los problemas que tienen relación con el mantenimiento. Así, los conjuntos difusos y los números difusos son apropiados para representar parámetros imprecisos, así como para poder manipularlos con operaciones propias y definidas en esta teoría. Es por ello, por lo que vamos a tratar con valores imprecisos en lugar de tratar con datos, pudiendo concluir que la teoría es más potente y por lo tanto más creíble. Los pasos a realizar se analizan en la fig1, y los iremos desgranando a lo largo de este epígrafe.

- *Enunciar el objetivo global del problema e identificar los criterios que influyen en el objetivo global*.

El problema del que se parte es el siguiente; contamos con piezas que tienen diversos tipos de suciedad, con geometría muy diferente, y necesitamos un proceso de trabajo que exige rapidez y flexibilidad a la hora de limpiar piezas. Además se deben tener en cuenta criterios como el coste total de operación anual, la productividad del sistema empleado, la capacidad de carga del sistema, la eficiencia en la limpieza y la salubridad de los productos utilizados. El objetivo global del problema es decidir cual es el mejor sistema de lavado de piezas.

Nos encontramos con abordar la solución al problema de manera lingüística ya que debido a las características de la empresa los datos son reservados, para ello la recogida de datos se desarrolló de manera lingüística, mediante la combinación con la teoría de conjuntos difusos.

- *Estructura jerárquica*

**Alternativas:**

- A1: Lavado convencional
- A2: Lavado químico
- A3: Lavado ultrasonidos

**Criterios:**

- C1: Coste Total de operación anual.
- C2: Productividad volumétrica de la Pila
- C3: Capacidad de carga
- C4: Eficiencia en la limpieza
- C5: Salubridad

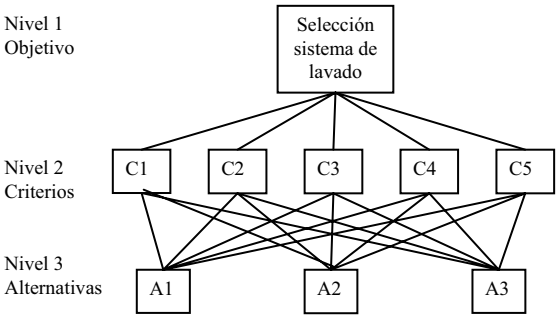



**Fig 2**:
Estructura jerárquica del problema

- *Elaboración de las encuestas gráficas al decisor*

La escala puede representarse de manera numérica, verbal o gráfica. En nuestro caso particular, mediante unas encuestas gráficas podemos encontrar la relación entre las etiquetas lingüísticas y los valores numéricos para estas etiquetas lingüísticas que define el decisor.

En este artículo, todos los elementos de la matriz de decisión y de los vectores de pesos se representan mediante números difusos triangulares, un número difuso $\tilde{x}$ expresa el significado de "alrededor de x". Cada función de pertenencia se define mediante tres parámetros (a,b,c,). Para obtener estos valores numéricos asociados con las etiquetas lingüísticas preparamos unas escalas gráficas donde evaluamos el punto izquierdo, el punto medio y el punto derecho del rango sobre el que la función está definida.

**Tabla 2**: Etiquetas lingüísticas para la bondad de las alternativas

| **Bondad** | **Número difuso** |
|---|---|
| Muy malo | [-10,-8,-6] |
| Malo | [-4.2,-1.7,-1.5] |
| Regular | [-1.5,-0.4,0] |
| Bueno | [0,1.7,3.7] |
| Muy bueno | [6,8,10] |

Igualmente repetimos las dos encuestas para la definición por parte del decisor de las etiquetas de importancia, definiendo en este caso por parte del decisor las siguientes etiquetas: Poco importante, moderadamente importante e importante. Tabla 3

Para normalizar las etiquetas de los criterios aplicamos la siguiente normalización:

$$c_j = \frac{C_j}{\sum_{j=1}^{n} C_n} \qquad (1)$$

**Tabla 3**: Etiquetas lingüísticas para la importancia de los criterios

| Importancia | Número difuso | Número difuso normalizado |
|---|---|---|
| Poco importante | [0,0.4,2.2] | [0,0.024,0.149] |
| Moderadamente importante | [4.2,4.6,6.8] | [0.159,0.279,0.462] |
| Importante | [6.3,6.5,8.4] | [0.239,0.394,0.571] |

- *Establecer la matriz de decisión*

Para el establecimiento de la matriz de decisión nuevamente se le pasó al decisor una encuesta en la que se le hacían una serie de preguntas concretas sobre el comportamiento de cada una de las alternativas bajo cada uno de los criterios y así como de la importancia que daba a cada uno de los criterios, utilizando como respuestas a estas preguntas las etiquetas anteriormente definidas por el mismo

| | Importante | Moderadamente Importante | Poco Importante | Moderadamente Importante | Poco Importante |
|---|---|---|---|---|---|
| | $C_1$ | $C_2$ | $C_3$ | $C_4$ | $C_5$ |
| $A_1$ | Bueno | Regular | Regular | Regular | Bueno |
| $A_2$ | Bueno | Bueno | Muy bueno | Bueno | Muy malo |
| $A_3$ | Regular | Muy bueno | Bueno | Bueno | Bueno |

- *Evaluación de criterios y alternativas*

Hacemos la traducción de las etiquetas lingüísticas de la matriz de decisión a los números difuso que definen cada una de las etiquetas y aplicamos el método de decisión multicriterio de la suma ponderada Triantaphyllou [15], dada la sencillez del mismo.

El método de suma ponderada clásico se define matemáticamente como sigue: Supongamos que el decisor asigna un conjunto de pesos $c=(c_1,...c_n)$ a los atributos $x_j$, $j=1,...,n$. La valoración de la alternativa $A_i$ se calcula como:

$$u_j = \sum_{j=1}^{n} c_j \cdot r_{ij} \tag{2}$$

donde $r_{ij}$ es la valoración de la $i$-esima alternativa bajo el $j$-esimo atributo con una escala comparable.

La alternativa preferida $A^*$ se selecciona como:

$$A^* = \left\{ A_j / \max u_j \right\} \tag{3}$$

**Tabla 4**: Evaluación de criterios y alternativas

| | **Pesos de Cj** | $A_1$ | $A_2$ | $A_3$ |
|---|---|---|---|---|
| $c_1$ | [0.239,0.394,0.571] | [0,1.7,3.7] | [0,1.7,3.7] | [-1.5,-0.4,0] |
| $c_2$ | [0.159,0.279,0.462] | [-1.5,-0.4,0] | [0,1.7,3.7] | [6,8,10] |
| $c_3$ | [0,0.024,0.149] | [-1.5,-0.4,0] | [6,8,10] | [0,1.7,3.7] |
| $c_4$ | [0.159,0.279,0.462] | [-1.5,-0.4,0] | [0,1.7,3.7] | [0,1.7,3.7] |
| $c_5$ | [0,0.024,0.149] | [0,1.7,3.7] | [-10,-8,-6] | [0,1.7,3.7] |
| ***Solución local*** | | | | |
| | | [0,0.670,2.113] | [0,0.670,2.113] | [-0.383,-0.157,0] |
| | | [-0.693,-0.112,0] | [0,0.474,1.709] | [0.954,2.232,4.62] |
| | | [-0.223,-0.009,0] | [0,0.0408,0.551] | [0,0.0408,0.551] |
| **Solución final** | | **[-1.609,0.4778,2.664]** | **[-1.49,1.467,6.082]** | **[0.571,2.631,7.431]** |

**Tabla 5**: Resultados

| Alternativas | $A_j$ $\left(A_j^a, A_j^b, A_j^c\right)$ |
|---|---|
| $A_1$ | [-1.609,0.4778,2.664] |
| $A_2$ | [-1.49,1.467,6.082] |
| $A_3$ | [0.571,2.631,7.431] |

Dentro del algoritmo que se plantea, también sería posible utilizar otros métodos de decisión multicriterio para la evolución de los criterios y alternativas como podrían ser el método analítico jerárquico (AHP) expuesto por Saaty [13, 14] o el método TOPSIS que puede verse en Chen y Hwang [1] así como cualquiera de los muchos que existen en la literatura y posteriormente hacer un estudio comparativo de los resultados obtenidos en cada caso.

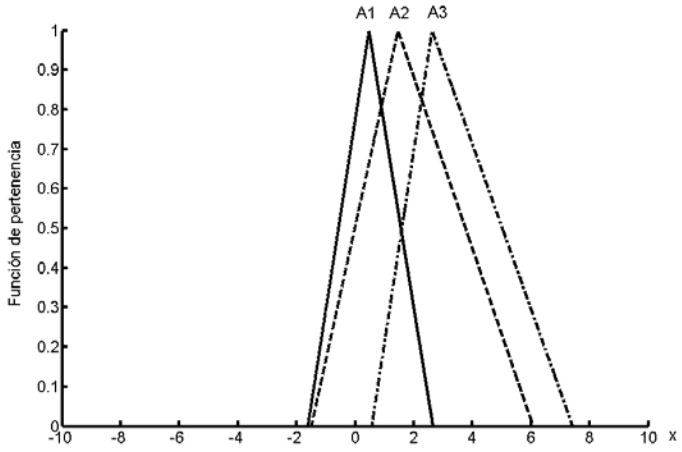



**Fig 3**: Resultados de la evolución de los criterios y las alternativas

- Ranking

$$\mathbf{A_3} \succ \mathbf{A_2} \succ \mathbf{A_1}$$

ya que:

$A_3^a = 0.571 > A_2^a = -1.49 > A_1^a = -1.609$

$A_3^b = 2.631 > A_2^b = 1.467 > A_1^b = 0.4778$

$A_3^c = 7.431 > A_2^c = 6.082 > A_1^c = 2.664$

**Tabla 6**: Resultados lingüisticos.

| | |
|---|---|
| $A_1$ | Mala |
| $A_2$ | Regular |
| $A_3$ | Buena |

La mayoría de las veces, el decisor no es capaz de definir la importancia de los criterios o en este caso la bondad de las alternativas con respecto a cada criterio de manera estricta. En general para el decisor es más fácil cuando evalúa sus juicios mediante términos lingüísticos

Obtenemos una solución en términos lingüísticos, pero también es posible considerar un distancia entre esta solución y las diferentes etiquetas, definidas en la tabla 2, en nuestro caso la distancia Hamming, de esta manera la solución se da en los mismos términos que la información obtenida del decisor.

## 4 Conclusiones

Se nos había planteado un problema de decisión multicriterio que en primera instancia tratamos de resolver mediante un modelo jerárquico. En nuestro caso solo teníamos información en términos lingüísticos.

Se intentó abordar el problema intentando simular el comportamiento del decisor, para ello mediante encuestas se obtuvo el conocimiento del experto así como su disposición ante el problema. La definición de números difusos triangulares nos permitió definir las etiquetas lingüísticas que el decisor empleaba en su proceso de decisión.

Esta aproximación al problema hace que podamos manejar de manera más real problemas de decisión en los que metodologías implementadas en ordenador resultan muy interesantes desde el punto de vista de la I.A. a la hora de simular el comportamiento humano.

El decisor, una vez resuelto el problema en función de sus preferencias estuvo de acuerdo con los resultados obtenidos.

## Agradecimientos

Este artículo se ha elaborado bajo los proyectos TIN2005-02418 y TIN2005-24790-E/ financiados por la DGICYT.

## Referencias

[1] S.J. Chen, C.L. Hwang, Fuzzy Multiple Attribute Decision Making: Methods and Applications, Springer-Verlang, Berlin, 1992
[2] C.L. Hwang, K. Yoon, Multiple Attibute Decision Methods and Applications, Springer, Berlin Heidelberg, 1981.
[3] Kacprzyt, J, Yager, R.R. (2001). Linguistic summaries of data using fuzzy logic. International Journal of General Systems, vol. 30, No 2, 133-154.
[4] Kablan, M. M., "Decision support for energy conservation promotion: an analytic hierarchy process approach," Energy Policy, vol. 32, no. 10, pp. 1151-1158, July2004.
[5] E.E. Kerre, The use of fuzzy set theory in electrocardiological diagnostics, in : M.M. Gupta and E. Sanchez, Eds., Approximate Reasoning in Decision Analysis (North-Holland, Amsterdam, 1982) 277-282
[6] Y.J. Lai, T.Y. Liu, C.L. Hwang, TOPSIS for MODM, European Journal of Operational Research 76 (3) (1994) 486-500
[7] M.T. Lamata. Ranking Alternatives with Ordered Weighted Averaging Operators. International Journal of Intelligent Systems, Vol 19(5), 473-482 (2004).
[8] M.T. Lamata. The graded numbers in the Analytic Hierarchy Process . Operations Research and Decisions, 125-138, 4 - 2003.
[9] Lootsma, F. (1999). Multi-Criteria Decision Analysis via Ratio and Difference Judgement. Kluwer Academic Publishers, Dordrecht.
[10] Luce R.D, Raiffa H.(1967). Games and Decisions: Introduction and Critical Survey. New York: John Wiley and Sons.
[11] Minski, Marvin La sociedad de la mente. La inteligencia humana a la luz de la inteligencia artificial. Ediciones Galapago. Argentina 1986
[12] Wiener, Norbert Cibernetica o el control y comunicaciones en animales y máquinas. Superínfimos 2-Tusquets Editores. Barcelona 1985 (ed. Origin. 1948) Bibliografía específica para un curso de Vida Artificial.
[13] Saaty.T.L. (1980). The Analytic Hierarchy Process. Macgraw Hill.
[14] Saaty.T.L. (1989). Group Decision Making and the AHP. Springer Verlag. New York.
[15] Triantaphyllou E (2000). Multi-Criteria decision making methods: A comparative study. The Netherlands: Kluwer Academic.
[16] Zadeh, L.A. (1975). The concept of a linguistic variable and its application to approximate reasoning: Part 1. Information Sciences 8, 199-249.
[17] Zadeh, L.A, Kacprzyt, J. (eds) (1999). Computing with Words in Information / Intelligent Systems. Part 1. Foundations, Physica-Verlag (Springer-Verlag), Heidelberg and New York.
[18] Zadeh, L.A, Kacprzyt, J. (eds) (1999). Computing with Words in Information/ Intelligent Systems. Part 2. Foundations, Physica-Verlag (Springer-Verlag), Heidelberg and New York.
[19] M. Zeleny, Multiple Criteria Decision Making, McGraw-Hill, New York, 1982

# Teorías del lenguaje: Alcance y crítica

Francisco Ureña Rodríguez

Universidad. de Sevilla. Facultad de Filosofía.
fraurerod@alum.us.es

**Resumen.** Este artículo pretende reflejar el carácter de la investigación homónima, que no es otro que establecer una crítica de los denominadores comunes de las grandes teorías del lenguaje natural en el siglo XX, y que tiene como base el rechazo a la continuidad de gran parte de los pensadores contemporáneos de una epistemología de corte moderno que comporta toda una serie de prejuicios e ideas asimiladas acríticamente. Se opta, en cambio, por una hermenéutica que sobredimensione el papel del intérprete como método para la reflexión última sobre el lenguaje, dentro de la cual debe entenderse la relación de éste con la cultura y con el individuo.

*Palabras clave:* Inteligencia natural, lenguaje natural, representación del conocimiento

## 1 Introducción

El propósito de una teoría del lenguaje no es otro que arrojar luz sobre la naturaleza de aquello que permite a todo hablante producir y comprender una infinidad de oraciones en una lengua, sobre la base de un vocabulario y de un conjunto de reglas que son finitos. El acercamiento a esta cuestión puede hacerse desde diversas perspectivas. O bien atendiendo a un cuerpo de conocimientos englobado bajo la etiqueta de Ciencias Cognitivas y del Cerebro, donde incluiríamos materias como la Psicología o la Neurofisiología; desde otras disciplinas científicas como la Antropología y la Lingüística; o desde una perspectiva filosófica, que tendría el principal cometido de elaborar una crítica de aquellos aspectos de las teorías científicas sobre el lenguaje en las que entran en juego conceptos cuyo uso o naturaleza pueden estar sujetos a discusión, debido sobre todo a que tales conceptos tienen tras de sí una dilatada historia de controversias en torno a su estatus.

Desde esta última perspectiva filosófica, la tarea aludida en las líneas anteriores, y que está sujeta a una mayor polémica, surge cuando se pretende explicar, desde las teorías del lenguaje, cómo el individuo llega a adquirir su competencia lingüística y en virtud de qué se llega a comprender las oraciones. Me estoy refiriendo a las partes de estas teorías donde intervienen conceptos como el de «significado» y aquellos vinculados a la idea de mente, como son el de «intencionalidad» o el de «representación mental».

El objetivo de esta investigación es mostrar que una investigación hermenéutica de carácter diacrónico sobre los conceptos fundamentales de la epistemología de la

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 815-820, 2006.

tradición occidental, pueden explicar la naturaleza de tales conceptos, de modo que a su vez esto sirva para transformar aquello que entendemos por una teoría del lenguaje, así como también para clarificar su utilidad e incluso considerar su posible pertinencia.

## 2 Antecedentes históricos en torno a la problemática de una teoría del lenguaje

La cuestión del lenguaje como vehículo de nuestros pensamientos, y la naturaleza de la relación entre nuestro pensamiento y el mundo, fueron temas que no estuvieron suficientemente enfocados hasta que fueron tratados por filósofos de los siglos XVII y XVIII, para los cuales el lenguaje no es sino un «espejo» de los pensamientos, y por tanto, los principios fundamentales del mismo y los mecanismos que lo subyacen en su uso, dependen en última instancia de la mente.

La respuesta al problema del significado y de la mente, desde sus orígenes modernos hasta el siglo XX, ha sido eminentemente una respuesta «constructiva» (Rorty, 1979). Con esta etiqueta, se hace referencia a toda producción intelectual que pretende elaborar un marco teórico de carácter permanente como base para la reflexión sobre determinadas cuestiones de índole, por lo general, filosófica. En el caso que nos ocupa, se trata de los problemas sobre la producción y comprensión del lenguaje, y su vinculación a los conceptos de significado y mente.

Es posible encontrar la presencia de enfoques de espíritu "constructivo", que tuvieron su origen en la Modernidad, y muy especialmente en Descartes, en las teorías más recientes sobre el lenguaje, donde existen, cuatrocientos años más tarde, los mismos propósitos que motivaron a los pensadores modernos a elaborar sus edificios epistemológicos, además de, claro está, los mismos aspectos susceptibles de duras críticas, en tanto que tales constructos reposan sobre "dogmas" gnoseológicos que pueden ponerse al descubierto mediante un análisis diacrónico de los conceptos clave de las teorías en cuestión.

Hay que añadir también que, las teorías constructivas propuestas para dar cuenta de estos problemas, a menudo han entrado en polémicas cuyo propósito ha sido principalmente el de relativizar el marco de referencia desde el cual reflexiona uno de los interlocutores, al subsumirlo bajo los cánones del marco de referencia asumido por el otro interlocutor. Además, dado el carácter de éste tipo de polémicas, nunca llega a establecerse *qué* va a considerarse una solución o término de las mismas, lo cual desvirtúa en cierto modo las capacidades explicativas o clarificadoras de éstas discusiones.

El interés de la investigación se centra en la crítica a los grandes grupos de teorías sobre el lenguaje del siglo XX[1] (Valdés, 1998), cuyo denominador común en torno a la problemática citada es el carácter constructivo descrito anteriormente. El primero

1 Esta distinción, aunque está inspirada en los diferentes conceptos centrales de estas teorías, responde también a una estrategia de carácter metodológico, que tiene por objeto facilitar el análisis histórico de la influencia del pensamiento moderno en dichas teorías.

de estos grupos estaría conformado por las teorías que giran en torno a los conceptos de verdad y verificación. Un segundo grupo estaría compuesto por aquellas teorías que, *inter alia*, intentan proporcionar una explicación de la noción fregeana de sentido utilizando nociones intencionales como las de «mundo posible. Y el tercer grupo de estas teorías estaría formado por aquellas en las que juegan un papel clave, para la explicación semántica, conceptos psicológicos como el de «intención» y «representación mental».

## 3 Crítica a la epistemología heredada en las teorías sobre el lenguaje

La tesis central de esta investigación afirma que no hay vínculo alguno entre un proyecto encaminado a establecer las condiciones de verdad de las oraciones de un lenguaje natural cualquiera, y los criterios para perfilar lo que pudiese ser la estructura esencial o última de la realidad. Por el contrario, se pretende mostrar que, en el contexto de una lengua, se opta por ofrecer una explicación de cómo el significado de las oraciones depende de la capacidad interpretativa de los agentes. Por ello, se rechaza la idea de que las palabras tengan significado absoluto, es decir, que tengan influencia sistemática más allá de las oraciones en las que ocurren.

La investigación propone una reconstrucción histórica del proceso de transformación de las ideas de Descartes en seguidores como Spinoza o Berkeley, dando lugar al nacimiento de conceptos como el de "idea" y a una concepción de la Filosofía entendida más bien como epistemología (búsqueda de fundamentos del conocimiento) que como una discusión racional sobre las cuestiones científicas, éticas o religiosas de la cultura. Por ello, el legado más duradero de Descartes no son sus ideas particulares, sino el convencimiento[2], por parte de los han venido tras él, de que en filosofía primaba la búsqueda de certezas fundamentales más que, como el propio nombre de esta disciplina indica, un "amor a la sabiduría". Estas ideas tuvieron su cumbre en la obra de Kant *Kritik der reinen Vernunft*, donde se hizo explícita de una manera ya famosa en nuestros días la distinción entre verdades necesarias y verdades constituyentes, y con ella la separación tajante entre las ciencias naturales y las humanas. No obstante, en esta investigación se considera necesario mostrar mediante un análisis histórico crítico dos cosas: que esta concepción tan exitosa en la filosofía moderna y gran parte de la contemporánea supone que los objetos a los que se refieren determinadas proposiciones imponen la verdad a las mismas, suposición que tiene su origen en un dogma que recibe el nombre de "principio platónico"[3]; y que en toda la tradición epistemológica moderna este principio ha sido objeto de diversas reinterpretaciones a modo de metáforas sobre cómo una ya supuesta "mente" aprehende el mun-

2 Además de dicho convencimiento, que duda cabe que debemos al pensador francés otras consecuencias de carácter epistemológico, como es el dualismo mente-mundo, y la sustitución paulatina de la dicotomía "apariencia-realidad" por la de "interior-exterior".

3 Este principio expresa que a las diferencias en certeza deben corresponder diferencias en los objetos conocidos (Rorty, 1979).

do. Metáforas en su gran mayoría apoyadas en la imagen de nuestra percepción visual, a modo de "ojo de la mente" que contempla el mundo.

En el siglo XX ha habido intentos de vincular la reflexión sobre las relaciones entre lenguaje, mente y mundo a perspectivas dualistas de tipo cartesiano que perpetuan los prejuicios que se pretende denunciar, lo que empobrece el discurso filosófico sobre esta cuestión, pues estos enfoques se encuentran demasiado comprometidas con marcos ideológicos "desgastados". Uno de estos intentos ha sido el planteamiento de la conducta sobre la base de una ontología de representaciones mentales, algo que tuvo su origen en la reacción contra el conductismo lógico. Otro de estos intentos solapados ha sido el de establecer analogías entre las relaciones lenguaje-mundo, y el problema cartesiano de las relaciones pensamiento-mundo.

Resulta más que conveniente la tarea de un análisis histórico que ponga de manifiesto la evolución de las ideas referidas a la elaboración de una teoría del lenguaje, y más en concreto de una teoría del significado que operase dentro de ella. Ello es debido a que la asunción acrítica de los prejuicios señalados ha encaminado predominantemente a los autores a pensar en una teoría del significado al modo de una herramienta formal que explicase las relaciones entre palabras y mundo, mediante la cual pudieran plantearse con claridad los problemas filosóficos para una más fácil resolución de los mismos, eludiendo el hecho de que cualquier marco referencial, por muy amplio, abarcador o abstracto que sea, para constituirse como tal debe adoptar una determinada configuración desde la cual se trazan y determinan sus límites, en tanto que se ven "afectados" por los condicionamientos sociales, históricos, psicológicos, económicos, etc., de sus creadores. Los planteamientos «constructivos» suponen un esencialismo (McDowell, 1996) que constituye el blanco de críticas de esta investigación, ya que ésta postura nunca queda justificada salvo por el dogmatismo platónico mencionado, el cual es posible abandonar si partimos, a la hora de reflexionar sobre el lenguaje, de las críticas de Quine al empirismo, dirigidas a lo que de esencialismo metafísico tiene éste, así como también las realizadas posteriormente por Davidson (Davidson, 1984).

## 4 Hacia un nuevo modo de afrontar el problema de la reflexión en torno al lenguaje

Los resultados que persigue la investigación presente están encaminados a mostrar las relaciones existentes entre las distintos tipos de proferencias, entendidas éstas como partes de una práctica social heredada mediante la cultura y que ejerce sobre el individuo una influencia a la hora de utilizar un determinado vocabulario, y una conducta ante el uso del mismo. También pretende esclarecer la relación entre la influencia señalada y la comprensión de los procesos inferenciales en el lenguaje natural; todo ello sin obviar la propia disposición humana para el lenguaje. Ello desplazaría la cuestión de la elaboración de una epistemología en torno al lenguaje natural en favor de una hermenéutica del mismo. Se considera en esta investigación que la hermenéutica resulta la mejor opción cuando se pretende "limpiar" la reflexión del lenguaje de prejuicios epistemológicos de origen moderno. Para ello, es preciso re-

nunciar a la idea de correspondencia entre oraciones y representaciones mentales, así como entre oraciones y mundo. Una epistemología basada en marcos conceptuales rígidos dificulta la práctica de una reflexión sobre el lenguaje crítica, dialéctica, ya que aquella tiene como propósito establecer un marco permanente de reflexión sobre el lenguaje, atentando contra el carácter sensible del mismo para adecuar su propia normatividad a los diferentes contextos históricos y sociales.

Este trabajo, por lo tanto, propone la defensa de la hermenéutica como contexto último dentro del cual ha de entenderse toda reflexión sobre el lenguaje natural, y rechaza el empleo de marcos epistemológicos estrechamente vinculados a principios metafísicos para la reflexión sobre el lenguaje, pues a menudo éstos se presuponen acríticamente. Esta hermenéutica se desarrollará no como una indagación sobre la intención del hablante o del que produce una expresión escrita, y la proferencia o el texto escrito en cuestión, sino estableciendo un giro que sobredimensione el papel del intérprete. Por ello una de las características de una hermenéutica así consiste en el intento de hacer de los intérpretes los creadores finales del significado. De esto se sigue el abandono del lenguaje como vehículo de la significación, vinculando éste fenómeno a la interacción entre la propia lengua y los procesos de mediación creativa del lector u oyente. No hay pues un encuentro neutral con la propia lengua, ya que la actualización de la misma se llevará a cabo por diferentes agentes que la interpretan atendiendo a diferentes trasfondos u horizontes culturales. Este subjetivismo del significado acentúa el carácter holístico del mismo, recuperando gran parte de la experiencia humana del lenguaje, que es mucho más amplia que la canonizada por la epistemología positivista moderna.

Esta versión hermenéutica tiene en principio como objeto un distanciamiento con respecto a las perspectivas epistemológicas en torno al significado que se han desarrollado en el siglo XX, las cuales han sobrevalorado y distorsionado nociones como fundamento o verdad, facilitando en muchas ocasiones polémicas en las cuales jamás quedaba claro qué se consideraría una solución o "salida" las mismas. Además, supone un cambio en relación con el *locus* del significado, que lo desvincula de todo objetivismo e independencia con respecto a los agentes, y dibuja una perspectiva acerca de la reflexión sobre el significado mucho más afín a la innegable condición histórica, social y biográfica del mismo, soslayada por las teorías sobre el lenguaje de corte epistemológico.

## Referencias

1. Davidson, D. (1984), *Inquiries into truth and interpretation*. 2 ed. Clarendon Press, Oxford (2001).
   A la ya clásica denuncia hecha por Quine al empirismo en «Two dogmas of empiricism» (la distinción fundamental entre verdades que son analíticas y las sintéticas; y la creencia de que todo enunciado que tenga sentido es equivalente a alguna construción lógica basada en términos que se refieren a la experiencia inmediata), hay que añadir la hecha por Davidson en *Inquiries into truth and interpretation,* donde denuncia la existencia de un "tercer dogma", a saber, la idea de marcos ontológicos o esquemas conceptuales como mediadores epistemológicos entre el mundo y nuestro discurso acerca de él.
2. McDowell, J. (1996), *Mind and World*. Harvard University Press, Camdridge, Mass.

3. Rorty, R. (1979), *Philosophy and the mirror of nature.* Princeton University Press. Princeton.
4. Valdés, Luis Ml. (1998), «El significado: los constructores». En: Acero, J. J. (ed) *Filosofía del lenguaje I. Semántica*. Trotta – C.S.I.C., Madrid.

# Traducción múltiple con transductores de estados finitos a partir de corpus bilingües

María-Teresa González y Francisco Casacuberta

Dept. Sistemas Informáticos y Computación
Universidad Politécnica de Valencia
46022 Valencia
{mgonzalez, fcn}@dsic.upv.es

**Resumen** Los transductores de estados finitos son modelos muy útiles en diferentes áreas del reconocimiento de formas. En trabajos anteriores se ha mostrado que son adecuados en la traducción automática. También se ha mostrado que inferir transductores a partir de corpus múltiples permitía obtener mejores traducciones que utilizando corpus bilingües. Dado que en el mundo real existen pocos corpus totalmente múltiples, pero sin embargo sí que existen bastantes corpus bilingües en diferentes idiomas y que pertenecen a la misma tarea. En este documento proponemos dos métodos que permiten inferir transductores de estados finitos múltiples a partir de corpus bilingües. También mostraremos resultados obtenidos de la experimentación realizada.

## 1. Introducción

Los transductores de estados finitos [1] a lo largo de los años han sido utilizados en diferentes áreas como pueden ser el reconocimiento sintáctico de formas [2,3] y en el procesado del lenguaje [4].

Una de las aplicaciones de los transductores es la que se realiza en *traducción del lenguaje*. Entendiendo como traducción el proceso en el que a partir de una frase de entrada en un idioma arbitrario, se obtiene una frase de salida en otro idioma diferente al original preservando el significado.

Sabemos que existen métodos de traducción con mayor potencia que los transductores de estados finitos [5], pero nos declinamos por éstos últimos debido a que su coste computacional es abordable y obtienen bastante buenos resultados en la práctica.

Otra de las características interesantes, en el campo de la traducción, es que se pueden inferir automáticamente a partir de un conjunto de muestras finitas [3].

Hoy en día, para traducir de un idioma a varios, lo que se viene haciendo es inferir un transductor de estados finitos para cada corpus bilingüe (corpus en donde las muestras son del tipo *(fuente,traducción)*), donde la fuente fuera del

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 821-829, 2006.

mismo idioma. También se ha probado que se puede llegar a inferir transductores de estados finitos múltiples[1] a partir de corpus múltiples[2] y de este modo obtener mejoras en las traducciones. Este método es el llamado GIATIM [6]. En este trabajo, lo que proponemos es inferir un único transductor de estados finitos múltiple, es decir, que permita traducir de un idioma a varios, tal y como se realizaba en GIATIM, pero utilizando corpus bilingües. Lo que se pretende con estos nuevos métodos es mantener las ventajas que se obtenían infiriendo con corpus múltiples, pero sin la necesidad de tenerlos. Al inferir esta clase de transductores obtenemos la ventaja de no tener un transductor para cada idioma y su consiguiente ahorro conputacional, ya que sólo es necesario un proceso de búsqueda y no uno por cada idioma al que se quiere traducir. Además, como se verá en los resultados, también tendremos otra ventaja, que es que un idioma puede llegar a beneficiarse de esta manera de la influencia de otro.

En la Sección 2 se exponen los métodos para inferir transductores de estados finitos múltiples con corpus de entrenamiento bilingües. En la Sección 3 se muestran los resultados obtenidos de la experimentación realizada con estos métodos y tomando para entrenamiento unos corpus bilingües derivados de los del proyecto *EuTrans* [7] de los que también se hablará en esta sección. Por último, una sección con las conclusiones y los trabajos futuros.

## 2. Métodos para inferir transductores de estados finitos para traducción múltiple a partir de corpus bilingües

En [8], se exponía un método para inferir transductores de estados finitos a partir de corpus bilingües llamado GIATI y en [6] uno para inferir transductores de estados finitos a partir de corpus múltiples llamado GIATIM. Las técnicas que se exponen en este documento, son para intentar aprovechar los beneficios que llegan a obtenerse con GIATIM pero sin necesidad de tener a priori corpus múltiples.

A continuación se describen los dos métodos desarrollados y algunos conceptos básicos para el mejor entendimiento de los métodos.

### 2.1. Alineamientos estadísticos

Los modelos de traducción estadística introducidos en [9] y [10] se basan en el concepto de *alineamiento* entre una frase fuente y su traducción, es decir, una palabra está *alineada* con la palabra que la produce.

[1] Transductores de estados finitos que tienen como entrada una fuente y como salida varias traducciones.

[2] Corpus en donde las muestras son del tipo *fuente/traducción1/.../traducciónN*, es decir, que dada una frase a traducir hay varias traducciones en diferentes idiomas para la misma. Ejemplo: *una habitación doble # a double room # una camera doppia # ein Doppelzimmer*, en donde la primera es la frase fuente en castellano y las siguientes su traducción al inglés, italiano y alemán respectivamente.

Formalmente, tomaremos el *alineamiento* de un par de traducción $(\mathbf{s}, \mathbf{t}) \in \Sigma^* \times \Delta^*$ como una función $\mathbf{a} : \{1, \ldots, |\mathbf{t}|\} \rightarrow \{0, \ldots, |\mathbf{s}|\}$. El caso particular $\mathbf{a}(j) = 0$ significa que la posición $j$ en $\mathbf{t}$ no está alineada con ninguna posición en $\mathbf{s}$. Por lo tanto, la probabilidad de traducir una frase dada $\mathbf{s}$ en $\mathbf{t}$ por un alineamiento $\mathbf{a}$ es $\Pr(\mathbf{t}, \mathbf{a}|\mathbf{s})$.

Algunas aproximaciones para estimar un alineamiento $\hat{\mathbf{a}}$ se propusieron en [10]. Aunque hoy en día hay herramientas de uso público que permiten realizar alineamientos óptimos entre pares de frases como es GIZA++ [11]. Un ejemplo de un alineamiento *castellano-inglés* sería el siguiente:

*¿ Cuánto cuesta una habitación individual por semana ?*
*How (2) much (2) does (3) a (4) single (6) room (5) cost (3) per (7) week (8) ? (9)*

Donde cada número entre paréntesis representa la posición que ocupa en la frase fuente (castellano) la/s palabra/s con la/s que se alinea la (posición de la) palabra de la frase traducida (inglés) que lo precede. Por ejemplo, en el caso anterior, la posición 1 de la frase traducida (*How*) está alineada con la posición 2 de la frase fuente (*Cuánto*), que formalmente lo representaremos como $\mathbf{a}(1) = 2$.

### 2.2. Método 1

La idea de este método es la de convertir los corpus bilingües en un corpus múltiple con el que con GIATIM podemos inferir transductores de estados finitos múltiples.

Formalmente el método sigue la siguiente técnica para aprender un transductor estocástico de estados finitos: dado un conjunto de muestras finitas $A_1$ de $(\mathbf{s_1}, \mathbf{t}_1) \in \Sigma^* \times \Delta_1^*$, $A_2$ de $(\mathbf{s_2}, \mathbf{t}_2) \in \Sigma^* \times \Delta_2^*$, $\ldots$, $A_N$ de $(\mathbf{s_N}, \mathbf{t}_N) \in \Sigma^* \times \Delta_N^*$ con $N \geq 1$ (varios corpus bilingües en donde las frases fuente pertenecen al mismo idioma pero no son las mismas):

1. Se infiere un transductor de estados finitos con GIATI para cada corpus bilingüe.
2. Se crea un corpus múltiple a partir de varios bilingües. Para ello, para cada muestra de $A_1, A_2, \ldots, A_N$ obtener la traducción. En el caso que dicha fuente no tenga ya una traducción en el corpus bilingüe correspondiente, se obtiene con el transductor correspondiente inferido en el paso anterior.
3. Se infiere un transductor de estados finitos múltiple con GIATIM.

Este método aquí presentado lo llamaremos GIATIM2.

### 2.3. Método 2

La idea de este método es la de ir infiriendo el transductor de estados finitos múltiple de una forma gradual. Ésto se realiza mediante un proceso iterativo en el que en cada iteración se inferirá un transductor de estados finitos que se utilizará en la siguiente iteración.

Formalmente el método sigue la siguiente técnica para aprender un transductor estocástico de estados finitos: dado un conjunto de muestras finitas $A_1$ de $(\mathbf{s_1}, \mathbf{t}_1) \in \Sigma^* \times \Delta_1^*$, $A_2$ de $(\mathbf{s_2}, \mathbf{t}_2) \in \Sigma^* \times \Delta_2^*$, ..., $A_N$ de $(\mathbf{s_N}, \mathbf{t}_N) \in \Sigma^* \times \Delta_N^*$ con $N \geq 1$ (varios corpus bilingües en donde las frases fuente pertenecen al mismo idioma pero no son las mismas):

1. Para conjunto de muestras $A_1$
   - Se infiere un transductor de estados finitos utilizando GIATI.
2. Para cada conjunto de muestras restante ($A_2$,...,$A_N$):
   *a*) Obtener el vocabulario extendido de cada muestra a partir del alineamiento de la misma al igual que en el primer paso en GIATI.
   *b*) Obtener el vocabulario extendido a partir del mejor camino obtenido en el último transductor inferido hasta el momento utilizando la frase fuente de cada muestra como entrada, es decir, el vocabulario extendido será el valor de las etiquetas de las aristas que recorre el mejor camino dada una muestra de entrada.
   *c*) Obtener el vocabulario extendido juntando los vocabularios extendidos de los pasos $2a$ y $2b$ por la palabra de entrada.
   *d*) Inferir un transductor de estados finitos a partir del vocabulario extendido del paso $2c$ de la misma manera que en el paso 2 y 3 de GIATI.

Este método aquí presentado lo llamaremos GIATIM3.

*Example 1.* Dado un conjunto de muestras finitas $A_1$(castellano-inglés) de $(\mathbf{s_1}, \mathbf{t}_1)$, $A_2$(castellano-italiano) de $(\mathbf{s_2}, \mathbf{t}_2)$ y $A_3$(castellano-alemán) de $(\mathbf{s_3}, \mathbf{t}_3)$:

1. Inferir un transductor $T_1$ de estados finitos con el corpus $A_1(\mathbf{s_1}, \mathbf{t}_1)$.
2. Para el corpus $A_2(\mathbf{s_2}, \mathbf{t}_2)$ :
   *a*) Obtener el vocabulario extendido $(s_2, t_2)$ como en el paso 1 de GIATI para cada muestra.
   *(mi,il mio) (nombre,nome) (es,e') (Eva,Eva) (Monferrer,Monferrer) (.,.)*
   *b*) Obtener el vocabulario extendido $(s_2, t_1)$ a partir del transductor $T_1$ para cada muestra.
   *(mi,my) (nombre,) (es,name is) (Eva,Eva) (Monferrer,Monferrer) (.,.)*
   *c*) Obtener el vocabulario extendido múltiple $(s_2, t_1, t_2)$.
   *(mi,my,il mio) (nombre,,nome) (es,name is,e') (Eva,Eva,Eva) (Monferrer,Monferrer,Monferrer) (.,.,.)*
   *d*) Inferir un transductor de estados finitos $T_2$ a partir del vocabulario extendido múltiple anterior.
3. Para el corpus $A_3(\mathbf{s_3}, \mathbf{t}_3)$ :
   *a*) Obtener el vocabulario extendido $(s_3, t_3)$ como en el paso 1 de GIATI.
   *(he,ich) (de,) (irme,muss) (el,am) (día,) (uno,1.) (de,) (septiembre,September wegfahren) (.,.)*
   *b*) Obtener el vocabulario extendido $(s_3, t_1, t_2)$ a partir del transductor $T_2$.

*(he,I,dovrei) (de,should,) (irme,leave,partire) (el,on,il) (día,,) (uno,,) (de,,) (septiembre,,) (.,.,.)*

*c*) Obtener el vocabulario extendido múltiple $(s_3, t_1, t_2, t_3)$.
*(he,I,dovrei,ich) (de,should,,) (irme,leave,partire,muss) (el,on,il,am) (día,,,) (uno,,,1.) (de,,,) (septiembre,,,September wegfahren) (.,.,.,.)*

*d*) Inferir un transductor de estados finitos $T_3$ a partir del vocabulario extendido múltiple anterior.

### 2.4. Búsqueda en transductores estocásticos de estados finitos mediante el algoritmo de Viterbi

La búsqueda de la traducción óptima $\hat{\mathbf{t}}$ en un tranductor estocástico de estados finitos $\mathcal{T}_P$ para una frase fuente $\mathbf{s}$,

$$\hat{\mathbf{t}} = \underset{\mathbf{t}\in\Delta^*}{\operatorname{argmax}} P_{\mathcal{T}_P}(\mathbf{s}, \mathbf{t}) \tag{1}$$

ha sido probado que es un problema computacionalmente difícil [12].

En la práctica, se puede obtener una solución aproximada [13] partiendo de la siguiente aproximación de la probabilidad de un par de frases (fuente, traducción):

$$P_{\mathcal{T}_P}(\mathbf{s}, \mathbf{t}) \approx V_{\mathcal{T}_P}(\mathbf{s}, \mathbf{t}) = \max_{\phi\in d(\mathbf{s},\mathbf{t})} P_{\mathcal{T}_P}(\phi) \tag{2}$$

Entonces la aproximación de la traducción quedaría como:

$$\tilde{\mathbf{t}} = \underset{\mathbf{t}\in\Delta^*}{\operatorname{argmax}} V_{\mathcal{T}_P}(\mathbf{s}, \mathbf{t}) = \underset{\mathbf{t}\in\Delta^*}{\operatorname{argmax}} \max_{\phi\in d(\mathbf{s},\mathbf{t})} P_{\mathcal{T}_P}(\phi), \tag{3}$$

que se puede llevar a cabo eficientemente [14] mediante programación dinámica.

Así que la traducción aproximada $\tilde{\mathbf{t}}$ se obtendrá como la concatenación de cadenas de traducciones asociadas con el formato de traducción

$$\phi = (q_0, s_1, \bar{\mathbf{t}}_1, q_1)(q_1, s_2, \bar{\mathbf{t}}_2, q_2)\dots(q_{I-1}, s_I, \bar{\mathbf{t}}_I, q_I) \tag{4}$$

que sería la secuencia óptima de estados en la solución, es decir,

$$\tilde{\mathbf{t}} = \bar{\mathbf{t}}_1\bar{\mathbf{t}}_2\dots\bar{\mathbf{t}}_I \tag{5}$$

En nuestro caso particular múltiple donde las muestras son del tipo *(fuente, traducción1,...,traducciónN)*, la búsqueda se realizaría de la misma manera que para dos pero, a la hora de traducir, se eligiría la traducciónN que queremos obtener. Por ejemplo, en el transdutor de la Figura 1, si el camino óptimo que se obtiene para la fuente "*una habitación doble*" es "*(una/a/una/ein) (habitación/λ/camera/λ) (doble/double room/doppia/Doppelzimmer)*", la traducción para el inglés sería "*a λ double room*", para el italiano "*a camera doppia*" y para el alemán "*ein λ Doppelzimmer*".

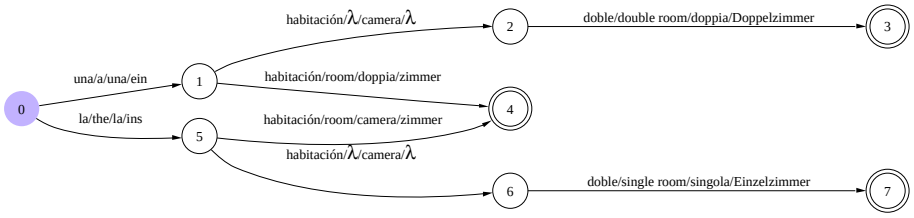



**Figura 1.** Transductor de estados finitos múltiple donde la fuente está en castellano y las traducciones al inglés, italiano y alemán.

## 3. Experimentación

Para la realización de los experimentos y que éstos sean lo más realistas posible, fueron creados varios corpus bilingües a partir de los del proyecto *EuTrans* [7]. En él se generaron varios corpus de 500, 000 pares de frases cada uno. Entre los se encuentran el *castellano-inglés* ($C1$), *castellano-italiano* ($C2$) y *castellano-alemán* ($C3$).

El proceso que se siguió para ello fue el que se expecifica a continuación. De $C1$, $C2$ y $C3$ (de cada uno de ellos), se eligieron aleatoriamente 3000 pares de frases para test y 10000 pares de frases para entrenamiento. La única restricción para los pares de frases para entrenamiento que se tuvo en cuenta fue que todos ellos tuvieran el mismo vocabulario de entrada, es decir, mismo vocabulario para la fuente. Con ello se obtuvieron tres corpus bilingües en donde las frases de entrada no eran las mismas, con lo cual no era un corpus múltiple.

Un resumen de los datos de estos nuevos corpus bilingües se pueden ver en el Cuadro 1.

**Cuadro 1.** Datos de los corpus bilingües castellano-inglés, castellano-italiano y castellano-alemán. No hay solapamiento entre el entrenamiento y el test. Además, el test no contiene palabras que estén fuera del vocabulario del entrenamiento y todos los corpus tienen el mismo vocabulario para el castellano en el entrenamiento.

| | | cast. | inglés | cast. | italiano | cast. | alemán |
|---|---|---|---|---|---|---|---|
| Entrenamiento | Pares de frases | 10000 | | | | | |
| | Pares distintos | 9704 | | 9707 | | 9671 | |
| | Palabras | 127701 | 128225 | 126023 | 117634 | 128634 | 120572 |
| | Vocabulario | 680 | 505 | 680 | 569 | 680 | 553 |
| Test | Pares de frases | 2870 | | 2865 | | 2969 | |
| | Palabras | 37147 | 37462 | 36094 | 33579 | 38108 | 35657 |
| | Vocabulario | 579 | 442 | 587 | 503 | 587 | 495 |
| | Perpl. bigrama | 7.9 | 5.7 | 8.3 | 6.9 | 7.7 | 6.0 |

Con dichos corpus realizamos los siguientes experimentos:

1. Obtener resultados utilizando GIATI [8] para los pares de frases *castellano-inglés*, *castellano-italiano* y *castellano-alemán* de los nuevos corpus para tener un valor de referencia. En [5] se ha demostrado que el método de GIATI es el que mejores resultados reporta.
2. Obtener resultados utilizando el método GIATIM2 expuesto en la Subsección 2.2 de *castellano-inglés-italiano-alemán*, *castellano-inglés-italiano*, *castellano-inglés-alemán* y *castellano-italiano-alemán* utilizando los nuevos corpus. Estos resultados se podrán comparar con los demás para ver si se obtienen mejoras con este nuevo método.
3. Obtener resultados utilizando el método GIATIM3 expuesto en la Subsección 2.3 de *castellano-inglés-italiano-alemán* utilizando los nuevos corpus. Estos resultados se podrán comparar con los anteriores para ver si se obtienen mejoras con este nuevo método.

Los resultados obtenidos se pueden ver en el Cuadro 2. Todos los resultados mostrados en él están evaluados con la medida de error WER[3].

**Cuadro 2.** Resultados obtenidos con GIATI, con GIATIM2 y con GIATIM3 al traducir del *castellano* al *inglés*, al *italiano* o al *alemán*. Los resultados están mostrados con la medida de error WER (ver nota a pie de página 3).

| | GIATI | GIATIM2 | | | | GIATIM3 |
|---|---|---|---|---|---|---|
| | | cast-ing-ita-ale | cast-ing-ita | cast-ing-ale | cast-ita-ale | |
| inglés | 7.17 | 7.41 | 7.35 | **6.97** | XXX | 20.51 |
| italiano | **2.37** | 2.58 | 2.47 | XXX | 2.43 | 12.10 |
| alemán | 10.50 | 10.81 | XXX | 10.71 | **10.46** | 10.69 |

A la vista de dichos resultados, podemos concluir lo siguiente para el método GIATIM2:

*Resultado buscado.*

- Dos lenguas germánicas pueden llegar a ayudarse entre si. Esto se puede observar en el caso del *castellano-inglés-alemán*, en donde el *alemán* beneficia al resultado final de las traducciones al *inglés*.

*Resultados previsibles.*

- La traducción de algunas frases que se realiza en el paso 2 del método GIATIM2, introduce errores que hacen que las mejoras obtenidas en GIATIM no sean tan buenas con este método.

[3] Número mínimo de operaciones de sustitución, borrado e inserción realizadas sobre las palabras de una frase traducida por el sistema para convertirla en la traducción de referencia dada por un traductor humano, dividido por el número total de palabras en la referencia.

- El italiano, al igual que el castellano, es una lengua románica. Por lo tanto, al pretender que la influencia de dos lenguas germánicas como son el inglés y el alemán, ayuden a mejorar los traducciones no da buenos resultados.

A la vista de dichos resultados, podemos concluir lo siguiente para el método GIATIM3:

- Los resultados se van degradando a medida que avanza el proceso de entrenamiento del método GIATIM3. Cómo se observa en el Cuadro 2, los peores resultados se obtienen para el inglés, que es el primer corpus bilingüe con el que se infiere el primer transductor de estados finitos de este método.
- Se pierde mucha información en cada iteración, ya que al inferir el siguiente transductor no se asegura que se pasa por todas las aristas del transductor.

## 4. Conclusión y trabajo futuro

Los resultados son muy similares para los métodos de GIATI y GIATIM2, aunque se puede apreciar en alguno de los casos que se llega a mejorar los resultados con GIATIM2. Sin embargo, con el método de GIATIM3 se observa que habría que hacer algún cambio en el método para solventar el problema de pérdida de información y así pudieran ofrecerse resultados competitivos.

Ésta parece ser una vía interesante de investigación, por lo que para futuros trabajos, se podrían investigar nuevas maneras de inferir transductores de estados finitos múltiples sin la necesidad de tener corpus múltiples a priori, es decir,con corpus bilingües, tal y como se ha realizado en este trabajo. Con ello, podríamos beneficiarnos de las mejoras que se obtienen con dichos transductores sin la necesidad de tener un corpus múltiple a priori.

## Agradecimientos

Este trabajo ha sido parcialmente financiado por el proyecto Itefte (CICYT TIC2003-08681-C02-02) y por la Agencia Valenciana de Ciencia y Tecnología (AVCiT) (Grupos I+D+I GRUPOS03/031).

## Referencias

1. Vidal, E., Casacuberta, F.: Learning finite-state models for machine translation. In: Grammatical Inference: Algorithms and Applications. Proceedings of the 7th International Coloquium ICGI 2004. Volume 3264 of Lecture Notes in Artificial Intelligence. Springer, Athens, Greece (2004) 16–27 Invited conference.
2. Fu, K.: Syntactic pattern recognition and applications. Pretice-Hall, Englewood Cliffs, MJ (1982)
3. Vidal, E., Casacuberta, F., García, P.: Grammatical inference and automatic speech recognition. In Rubio, A., ed.: New Advantages and Trends in Speech Recognition and Coding. Volume 147 of NATO-ASI Series F:Computer and Systems Sciences. Springer-Verlag (1995) 174–191

4. Mohri, M.: Finite-state transducers in language and speech processing. Computational Linguistics **23**(2) (1997) 269–311
5. Casacuberta, F., Ney, H., Och, F.J., Vidal, E., Vilar, J.M., Barrachina, S., Garcia-Varea, I., Llorens, D., Martinez, C., Molau, S., Nevado, F., Pastor, M., Pico, D., Sanchis, A.: Some approaches to statistical and finite-state speech-to-speech translation. Computer Speech and Language **18** (2004) 25–47
6. González, M.T., Casacuberta, F.: Traducción múltiple con transductores de estados finitos. In de la Blanca Capilla, N.P., Bañón, F.P., eds.: Actas del Primer Congreso Español de Informática (CEDI 2005), Simposio de Reconocimiento de Formas y Análisis de Imágenes (AERFAI). ISBN: 84-9732-445-5, Granada (España), Thomson (2005) 37–44
7. Instituto Tecnológico de Informática, Bordoni, F.U., für Informatik VI, R.W.T.H.A.L., Bochum, Z.G.: Example-based language transalation systems: Final report. Technical report (2000)
8. Casacuberta, F., Vidal, E.: Machine translation with inferred stochastic finite-state transducers. Computational Linguistics **30**(2) (2004) 205–225
9. Brown, P.F., Cocke, J., Pietra, S.D., Pietra, V.J.D., Jelinek, F., Lafferty, J.D., Mercer, R.L., Roossin, P.S.: A statistical approach to machine translation. Computational Linguistics **16**(2) (1990) 79–85
10. Brown, P.F., Pietra, S.D., Pietra, V.D., Mercer, R.: The mathematics of statistical machine translation: Parameter estimation. Computational Linguistics **19**(2) (1993) 263–312
11. Och, F.J., Ney, H.: Improved statistical alignment models. In: Proc. of ACL'00, Hong Kong, China (2000) 440–447
12. Casacuberta, F., de la Higuera, C.: Computational complexity of problems on probabilistic grammars and transducers. In Oliveira, A., ed.: Grammatical Inference: Algorithms and Applications. Volume 1891 of Lecture Notes in Computer Science. Springer-Verlag (2000) 15–24 5th International Colloquium Grammatical Inference -ICGI2000-. Lisboa. Portugal. Septiembre.
13. Casacuberta, F.: Inference of finite-state transducers by using regular grammars and morphisms. In Oliveira, A., ed.: Grammatical Inference: Algorithms and Applications. Volume 1891 of Lecture Notes in Computer Science. Springer-Verlag (2000) 1–14 5th International Colloquium Grammatical Inference -ICGI2000-. Lisboa. Portugal. Septiembre.
14. Casacuberta, F.: Maximum mutual information and conditional maximum likelihood estimations of stochastic syntax-directed translation schemes. In Miclet, L., de la Higuera, C., eds.: Grammatical Inference: Learning Syntax from Sentences. Volume 1147 of Lecture Notes in Artificial Intelligence. Springer-Verlag (1996) 282–291

# Algunas soluciones al problema del escalado en traducción automática estadística

Daniel Ortiz-Martínez[1], Ismael García-Varea[2] y Francisco Casacuberta[1]

[1] Universidad Politécnica de Valencia
[2] Universidad de Castilla-la Mancha

**Resumen** En este artículo se trata el problema de la estimación de modelos de secuencias a partir de corpus muy grandes, y su posterior aplicación en traducción automática estadística. La gran cantidad de pares de frases contenidos en corpus de reciente aparición como el bien conocido corpus EUROPARL, ha incrementado sobremanera los requerimientos de memoria a la hora de entrenar modelos de secuencias y aplicarlos más tarde en un proceso de búsqueda, haciendo inabordables estas tareas en determinados casos. En este artículo se proponen una serie de técnicas que permiten resolver estos problemas sin introducir sobrecargas temporales significativas. Las técnicas están basadas en el trabajo con cuentas en lugar de probabilidades, y en el concepto clásico de arquitecturas de memoria caché. Las explicaciones teóricas de las técnicas se acompañan de resultados empíricos.

## 1. Introducción

La existencia, cada día mayor, de grandes corpus paralelos (o bases de datos de frases traducidas a distintos idiomas) ha provocado durante las últimas dos décadas que la aproximación estadística a la traducción automática (TAE) sea una de las disciplinas de mayor estudio dentro del área del procesamiento del lenguaje natural. Bajo esta perspectiva, el proceso de traducción desde el punto de vista estadístico se puede formular del siguiente modo: dada una frase $f_1^J = f_1...f_j...f_J$ en un lenguaje de entrada, se pretende traducir a otra frase $e_1^I = e_1...e_i...e_I$. De todas las posibles frases en el lenguaje de salida, elegiremos aquella que nos devuelva la mayor probabilidad a posteriori $Pr(e_1^I|f_1^J)$. Aplicando la regla de decisión de Bayes tendremos que elegir la frase de salida $\hat{e}_1^I$ que maximice la expresión:

$$\hat{e}_1^I = \arg\max_{e_1^I}\{Pr(e_1^I) \cdot Pr(f_1^J|e_1^I)\} \quad (1)$$

donde $Pr(e_1^I)$ representa la probabilidad de generar la frase $e_1^I$ y $Pr(f_1^J|e_1^I)$ la probabilidad de obtener $f_1^J$ habiendo observado $e_1^I$. En la práctica se obtiene una estimación de $Pr(e_1^I)$ mediante un *modelo de lenguaje* y de $Pr(f_1^J|e_1^I)$ mediante un *modelo de traducción*.

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 830-842, 2006.

En los orígenes de la TAE los modelos de traducción (MT) se basaban en relaciones estructurales a nivel de palabra [1]. Desde hace solo unos años los MT se han extendido a modelos que intentan capturar relaciones entre grupos de palabras, dando lugar a los MT basados en secuencias (o grupos de palabras), como por ejemplo [2,3,4,5].

La disponibilidad de grandes corpus de datos (plurilingües) hace que los sistemas de traducción basados en este tipo de modelos sean mucho más competitivos que sus predecesores. Por contra conllevan grandes requisitos computacionales tanto a nivel espacial como temporal, provocados por la necesidad de manejar estructuras de datos que permitan utilizar dichos diccionarios estadísticos de manera apropiada. Precisamente en este aspecto es en el que se centra todo el trabajo desarrollado en este artículo.

Recientemente en [6] se ha propuesto una estructura de datos de reducido coste espacial para tratar eficientemente con este problema. Propuesta que, a nuestro juicio, tiene dos serios inconvenientes:

- Por un lado, en máquinas de 32 bits un proceso dispone como máximo de 2 Gbytes de espacio de almacenamiento. En el mencionado artículo, y según sus propios autores, el modelo de secuencias que se obtiene para el corpus bajo estudio requiere justo 2 GBytes de espacio de memoria. Por tanto, si usamos máquinas de 32 bits, ya habríamos alcanzado el techo que permite la técnica.
- Por otro lado, la técnica presentada es extremadamente lenta a la hora de recuperar la probabilidad para un par de secuencias, siendo más lenta cuanto más frecuente es el par (requiriendo varios segundos en determinados casos). En el citado artículo se propone una solución a este grave inconveniente con una técnica alternativa que permite recuperar las probabilidades con mayor rapidez, al precio de que dichas probabilidades no sean exactas, sino aproximadas. Esta solución, a nuestro juicio, no es satisfactoria.

En el trabajo que aquí presentamos no sólo se solucionan los inconvenientes mencionados acerca del trabajo [6], sino que se mejoran las prestaciones tanto a nivel de entrenamiento como a nivel de traducción propiamente dicha.

## 2. Modelos de traducción basados en secuencias

Como ya hemos comentado, éstos modelos surgen como alternativa a los modelos de palabras (o más comúnmente conocidos como modelos de IBM [1]) para superar las limitaciones que presentan, de modo que en lugar de trabajar con diccionarios estadísticos de palabras ($Pr(f_j|e_i)$), trabajan con diccionarios estadísticos de secuencias ($Pr(\tilde{f}_k|\tilde{e}_k)$).

La traducción, usando modelos de secuencias de palabras, de una frase de entrada $f_1^J$ en la frase destino equivalente $e_1^I$ consiste, desde un punto de vista generativo, en escoger la forma en que dicha frase de entrada es segmentada en $K$ secuencias $f_1^J = \tilde{f}_1^K$, seleccionar las secuencias en el lenguaje destino que

traducen las secuencias origen y, por último, reordenar; con lo que terminamos obteniendo $e_1^I = \tilde{e}_1^K$.

En general todos los trabajos propuestos para modelos de secuencias [2,3,4,5,7] basan sus sistemas en diccionarios estadísticos de secuencias, los cuales se estiman mediante la técnica de máxima verosimilitud, que normalmente consiste en calcular $p(\tilde{f}|\tilde{e})$ del siguiente modo:

$$p(\tilde{f}|\tilde{e}) = \frac{N(\tilde{f}, \tilde{e})}{N(\tilde{e})} \tag{2}$$

donde $N(\tilde{f}|\tilde{e})$ es el número de veces que $\tilde{f}$ se ha visto como traducción de $\tilde{e}$ en todo el corpus de entrenamiento.

En general los modelos de secuencias, como ya hemos comentado en la sección anterior, el gran inconveniente que poseen es la gran cantidad de espacio en memoria requerida por los parámetros del modelo [8].

## 3. Modelos de traducción basados en secuencias

Como ya hemos comentado, éstos modelos surgen como alternativa a los modelos de palabras (o más comúnmente conocidos como modelos de IBM [1]) para superar las limitaciones que presentan, de modo que en lugar de trabajar con diccionarios estadísticos de palabras ($Pr(f_j|e_i)$), trabajan con diccionarios estadísticos de secuencias ($Pr(\tilde{f}_k|\tilde{e}_k)$).

La traducción, usando modelos de secuencias de palabras, de una frase de entrada $f_1^J$ en la frase destino equivalente $e_1^I$ consiste, desde un punto de vista generativo, en escoger la forma en que dicha frase de entrada es segmentada en $K$ secuencias $f_1^J = \tilde{f}_1^K$, seleccionar las secuencias en el lenguaje destino que traducen las secuencias origen y, por último, reordenar; con lo que terminamos obteniendo $e_1^I = \tilde{e}_1^K$.

En general todos los trabajos propuestos para modelos de secuencias [2,3,4,5,7] basan sus sistemas en diccionarios estadísticos de secuencias, los cuales se estiman mediante la técnica de máxima verosimilitud, que normalmente consiste en calcular $p(\tilde{f}|\tilde{e})$ del siguiente modo:

$$p(\tilde{f}|\tilde{e}) = \frac{N(\tilde{f}, \tilde{e})}{N(\tilde{e})} \tag{3}$$

donde $N(\tilde{f}|\tilde{e})$ es el número de veces que $\tilde{f}$ se ha visto como traducción de $\tilde{e}$ en todo el corpus de entrenamiento.

En general los modelos de secuencias, como ya hemos comentado en la sección anterior, el gran inconveniente que poseen es la gran cantidad de espacio en memoria requerida por los parámetros del modelo [8].

## 4. Una estructura de datos eficiente para almacenar pares bilingües

Dado que se pretende almacenar en memoria un conjunto muy grande de pares bilingües, será fundamental reducir al mínimo posible el coste espacial de la estructura de datos que represente dichos pares. No obstante, no hay que olvidar que posteriormente, en un proceso se aplicación del modelo, necesitaremos acceder a ellos en tiempo razonable, por lo que el coste temporal tampoco debe descuidarse.

Suponiendo que, en lugar de tener que almacenar un conjunto de pares bilingües, hubiera de almacenarse un conjunto de secuencias en un único idioma, podrían plantearse distintas alternativas de representación, como por ejemplo vectores o árboles. Los vectores son eficientes en cuanto al consumo de espacio, al almacenar elementos en posiciones consecutivas de memoria. El coste de inserción es constante, mientras que por el contrario, el de recuperación es lineal. Los árboles, por su parte, tienen costes de inserción y recuperación logarítimicos pero son menos eficientes en cuanto al espacio por requerir el uso de punteros. Ambas estructuras tienen además el inconveniente de que no explotan la redundancia que suele darse en el caso concreto de las secuencias de palabras, que pueden tener multitud de prefijos, interfijos y sufijos comunes.

Una estructura que supone un compromiso entre velocidad y espacio requerido y que además explota la redundancia típica de las secuencias de palabras es el *trie*, que tiene complejidad logarítmica de recuperación y almacenamiento [9].

No obstante, el objetivo es almacenar un conjunto de pares de secuencias, siendo muy importante expresar la relación entre dichos pares (esto es, los alineamientos a nivel de secuencia) con la menor cantidad de memoria posible. Para conseguir esto, se propone un doble trie asimétrico como el mostrado en la Figura 1. En la parte izquierda se da un pequeño conjunto formado por pares de secuencias inglés-castellano, y en la parte derecha una representación gráfica de la estructura de datos propuesta.

El trie asociado al lenguaje origen (el inglés) se compone de una serie de nodos, cada uno de los cuales almacena una palabra de la secuencia y la cuenta de dicha secuencia $c(\tilde{e})$. Por otro lado, el trie asociado al lenguaje destino (el castellano), contiene nodos relacionados con las palabras de las secuencias destino y además una serie de punteros a nodos del trie para las secuencias origen, que sirven para expresar alineamientos a nivel de secuencia. Cada uno de estos punteros debe llevar asociado un dato adicional consistente en la frecuencia absoluta del par $c(\tilde{e}, \tilde{f})$. Asimismo, cada uno de los nodos del trie asociado a la lengua origen tienen punteros a sus nodos padre, de manera que, a partir de un puntero a un nodo origen, puede recuperarse la secuencia de palabras completa que representa.

Para recuperar la probabilidad de un par de secuencias $(\tilde{e}\#\tilde{f})$ , primero se busca la secuencia origen $\tilde{e}$, guardando el puntero que la representa y su cuenta $c(\tilde{e})$. Con esta información nos vamos al segundo trie y buscamos $\tilde{f}$. Una vez encontrado el nodo correspondiente, buscamos a su vez el puntero a $\tilde{e}$ que

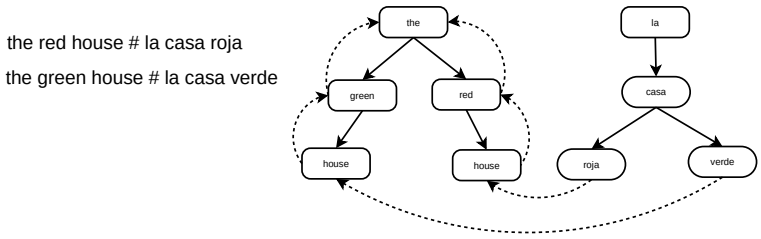



**Figura 1.** Estructura de datos para el almacenamiento de pares bilingües

nos habíamos guardado antes, lo que nos permitirá obtener $c(\tilde{e}, \tilde{f})$. Una vez recuperadas ambas cuentas, la probabilidad del par viene dada por $c(\tilde{e}, \tilde{f})/c(\tilde{e})$.

Otro posible uso que se puede dar a la estructura consiste en la obtención del conjunto de posibles traducciones de una secuencia destino $\tilde{f}$, que es básica en algoritmos de decodificación basados en pila o programación dinámica (PD). Para ello, se busca $\tilde{f}$ en el trie destino, con lo que se obtiene un conjunto de punteros que representan las posibles traducciones de $\tilde{f}$. Para obtener la secuencias representadas por los punteros nos serviremos de los punteros a los nodos padre del trie origen.

La estructura de datos anterior presenta además una serie de propiedades interesantes:

- Recuperación de probabilidades con complejidad logarítmica.
- Su organización y el uso de cuentas permite obtener eficientemente tanto $p(\tilde{f}|\tilde{e})$ como $p(\tilde{e}|\tilde{f})$, lo que resulta útil para no tener que cargar tablas de probabilidad inversa que se usan normalmente en algoritmos de pila y PD.
- Permite modificar dinámicamente las cuentas sin tener que renormalizar el modelo.

Dado que la estructura de datos propuesta está fuertemente centrada en el uso de tries, resulta de gran importancia profundizar en aspectos relacionados con la implementación de los mismos. Los tries pueden implementarse de diversas formas con distintas propiedades. En [9] se propone una implementación en la que cada nodo posee dos punteros, uno al primer hijo del nodo y otro al primer nodo hermano. No obstante, pueden proponerse implementaciones alternativas en las que los hijos de cada nodo pueden accederse mediante árboles o vectores ordenados. De entre estas tres alternativas, nos hemos decantado por la primera debido a que no introduce aumentos en el coste espacial y además provee unos costes de recuperación logarítmicos.

Con la estructura de datos que hemos definido, la cantidad de memoria en bytes que se requiere para almacenar un modelo de secuencias completo viene dado por la expresión:

$$\begin{aligned} &palabras\_orig \times factor\_compresion\_trie\_orig \times 5\, numbytes(int) + \\ &palabras\_dest \times factor\_compresion\_trie\_dest \times 3\, numbytes(int) + \\ &\qquad pares\_secuencias \times 2\, numbytes(int) \end{aligned}$$

Tanto el número de palabras origen como el número de palabras destino serán comprimidos en cierta medida al almacenarse en el trie, esta compresión se refleja con un factor entre 0 y 1 que dependerá de las características del corpus en cuestión. Para los corpus bajo estudio se ha encontrado que ese factor de compresión, tanto para el trie origen como para el trie destino nunca es superior a 0,3. Creemos que esto hace que la estructura de datos sea competitiva en términos de complejidad espacial con la propuesta en [6].

Cada nodo del trie requiere 3 enteros, uno para almacenar la palabra que contiene y otros dos como punteros a nodos hijo y hermano. El trie asociado a las secuencias origen requiere un entero adicional para los punteros a los nodos padre y otro más para la cuenta de la secuencia que representa. Por último, cada alineamiento a nivel de secuencia requiere un entero, y la cuenta del par de secuencias involucradas otro entero más.

## 5. Entrenamiento de modelos

Pese a que la estructura introducida en el apartado anterior presenta un reducido coste espacial, aparecerán problemas cuando sea preciso estimar un modelo a partir de un corpus muy grande, como por ejemplo el corpus Europarl. Estos problemas serán de especial importancia en máquinas de 32 bits, en donde es insalvable el límite de 2 GBytes para los datos.

Una posible solución a este problema, que permite el entrenamiento de corpus de tamaño arbitrario, viene dada por el Algoritmo 1. Dicho algoritmo se apoya en el trabajo con cuentas en lugar de probabilidades. Consiste en entrenar el corpus dividiéndolo en trozos de tamaño fijo ($tam\_fragm$) volcando cada uno de los submodelos que se obtienen en disco. Una vez volcados los modelos se juntan en un único fichero, se ordenan lexicográficamente y se fusionan las cuentas, dando lugar a un modelo idéntico al que se obtendría de haber entrenado el corpus completo de una sola vez.

**Algoritmo 1** algoritmo_entrenamiento (fichAlin)

```
partir(fichAlin,tam_fragm)
for all f fragmento de fichAlin do
  entrena(f) >> fichCuentas
end for
ordenarCuentas(fichCuentas) > fichCuentasOrdenado
fundirCuentas(fichCuentasOrdenado) > modeloSecuencias
```

La única sobrecarga temporal que introduce el algoritmo se debe a la necesidad de ordenar el fichero de cuentas resultante. Creemos que el algoritmo es claramente susceptible de ser paralelizado, por lo que la mencionada sobrecarga temporal puede ser contrarrestada.

## 6. Traducción con modelos de secuencias muy grandes

El problema de los elevados requerimientos de memoria que se ha expuesto en la sección anterior para el caso del entrenamiento de modelos de secuencias, persiste a la hora de acceder a dichos modelos en un proceso de búsqueda en traducción automática.

Hasta ahora, la solución más común a este problema ha consistido en buscar estructuras de datos de reducido coste espacial [6] aunque con contrapartidas negativas en lo relativo al coste temporal. No obstante, cualquier propuesta de este tipo pasa por cargar el modelo de secuencias completo en memoria, con todo lo que ello conlleva.

La solución que se propone para resolver este problema pasa por importar un concepto clásico en arquitectura de computadores: la *memoria caché*. La memoria caché se apoya en el *principio de localidad* que establece que los programas no acceden a memoria al azar, sino que los accesos están próximos unos de otros tanto en espacio como en tiempo. Este principio de localidad se manifiesta en el caso que nos ocupa de dos maneras distintas:

1. En los modelos de secuencias habituales se observa que la mayoría de los pares de secuencias tiene una frecuencia de aparición muy escasa. Por tanto, es previsible que una parte importante del modelo apenas se requiera o no se requiera en absoluto a la hora de traducir un conjunto de frases. En los modelos de secuencias que se generaron para HANSARDS, el 94 % de los pares habían aparecido sólo una vez. Para EUROPARL esa cifra es del 92 %.
2. En el caso concreto de la traducción con algoritmos de pila y programación dinámica, cuando se traduce cada frase, y dado que se trabaja con tablas de $N$ mejores traducciones inversas, es muy reducido el número de entradas del diccionario al que hay que acceder con respecto al total, y además el número de veces que se accede a esas entradas es enorme. Pueden identificarse perfectamente los principios de localidad temporal y espacial.

Lo anterior nos lleva a proponer una jerarquía de memoria con una caché de dos niveles. En el primero se van a almacenar los pares bilingües que se requieren para traducir una frase. Este nivel es local a la frase y se borra cada vez que empieza una búsqueda.

El segundo nivel contiene un determinado porcentaje de los pares más frecuentes. Este nivel estará cargado en memoria a lo largo de todo el proceso de traducción.

Por último, tendremos el modelo completo almacenado en disco, organizado de la forma que sea necesaria para poder recuperar las probabilidades de los pares de secuencias con coste logarítmico mediante búsqueda binaria.

Es muy importante resaltar que la unidad básica de información con la que se trabajará en la jerarquía de memoria propuesta es la secuencia destino $f$ junto con todas sus posibles traducciones en la lengua origen. Esto se hace así para favorecer la localidad espacial.

De esta forma, cada vez que el traductor necesita recuperar la probabilidad de un par de segmentos $(\tilde{e}\#\tilde{f})$ mira en la caché de primer nivel. Si está el

par devuelve la probabilidad, si no, busca las traducciones de $\tilde{f}$ en la caché de segundo nivel y las copia en la caché de primer nivel en caso de que las hubiera, devolviendo la probabilidad del par si $\tilde{e}$ está entre las posibles traducciones de $\tilde{f}$. Si no se encontrasen traducciones para $\tilde{f}$ en la caché de segundo nivel, entonces se accede a disco.

Cuando se accede a disco, se buscan las posibles traducciones de $\tilde{f}$. Tanto si existen como si no, se registra el resultado de la búsqueda en la caché de primer nivel (así, si el par no existe, no provoca pérdidas de tiempo innecesarias repitiendo esa búsqueda) y se devuelve la probabilidad del par.

Cuando termina de traducirse la frase, se borra el contenido de la caché de primer nivel, de manera que en todo momento, sólo tendremos almacenado en memoria el porcentaje del modelo que hayamos escogido. El porcentaje de pares de secuencias del modelo que se cargan en la caché de segundo nivel lo vamos a llamar $\alpha$, siendo un parámetro de esta arquitectura de acceso a modelos que se propone. Los pares de secuencias que se carguen en memoria serán precisamente aquellos que tengan una mayor frecuencia de aparición, de acuerdo con el primer principio de localidad expuesto al inicio de esta sección.

El parámetro $\alpha$ al tratarse de un porcentaje podrá tomar valores entre 0 y 100, ambos extremos se corresponden con casos particulares de interés:

$\alpha$=**0** : la caché de segundo nivel se queda vacía por lo que no se carga absolutamente ninguna entrada del diccionario en memoria. Esto hará que se eleve la tasa de fallos de caché, pero permite traducir sin cargar el modelo en memoria.

$\alpha$=**100** : se carga el modelo completo en la caché de segundo nivel. Esto permite, por un lado, traducir sin que se produzcan fallos de caché, lo que supone una gran diferencia con respecto a otro valores de $\alpha$. Por otro lado, es más rápida que el enfoque tradicional al beneficiarse de la velocidad de acceso de la caché de primer nivel.

## 7. Experimentos

En esta sección se dan una serie de experimentos tanto de entrenamiento como de traducción con modelos de secuencias estimados a partir de los corpus presentados en la sección **??**. Para el caso del corpus HANSARDS se utilizó un subconjunto de 1000 frases del conjunto estándar de test, con objeto de reducir el tiempo requerido por los experimentos.

### 7.1. Entrenamiento

El Cuadro 1 muestra los costes temporal y espacial requeridos por la estimación de un modelo de frase para los corpus HANSARDS y EUROPARL de la forma clásica y de la forma propuesta en la sección 5[3] . El tamaño máximo de segmento se estableció en 8 palabras para HANSARDS y en 6 para EUROPARL.

[3] Todos los experimentos fueron ejecutados en un PC con procesador Intel Pentium 4 con 2.6 Ghz de frecuencia de reloj y 2 GB de memoria RAM.

| | | tiempo (min.) | MBytes |
|---|---|---|---|
| **Hansards** | est. clásica | 18 | 305 |
| | est. a trozos | 31 | 128 |
| **EuroParl** | est. clásica | 181 | 1507 |
| | est. a trozos | 178 | 128 |

**Cuadro 1.** Costes temporal y espacial del entrenamiento para los corpus HANSARDS y EUROPARL

El coste espacial para el entrenamiento clásico llega a superar 1 GByte cuando se trabaja con el corpus EUROPARL[4], mientras que el coste para el entrenamiento a trozos es fijo e igual a 128 MBytes. Este valor es el tamaño máximo de memoria asignado al algoritmo de ordenación para su funcionamiento, y puede reducirse a costa de aumentar el tiempo requerido para que dicha ordenación se produzca.

Con respecto al coste temporal de los algoritmos, se observa algo muy llamativo, y es que mientras que la estimación clásica es más rápida que la estimación a trozos cuando se entrena el corpus HANSARDS, esta situación se invierte para el caso del corpus EUROPARL. La razón de ello está en que la sobrecarga introducida por la ordenación se ve contrarrestada por una mayor velocidad de manejo de las cuentas cuando se trabaja con submodelos pequeños, como es el caso del entrenamiento a trozos. Estos tiempos hacen pensar, asimismo, que la eficiencia de la estructura de datos presentada en la sección 4 se degrada cuando el modelo almacenado es muy grande.

### 7.2. Decodificación

Con objeto de establecer las prestaciones de la técnica de acceso a modelos propuesta en la sección 6 se ha llevado a cabo una serie de experimentos de traducción tanto para el corpus HANSARDS como para el corpus EUROPARL, con distintos valores del parámetro $\alpha$. Asimismo, y con vistas a determinar cual es el grado de sobrecarga temporal introducido por la nueva técnica, se define un caso base que consiste en traducir cargando en memoria el modelo completo de la manera convencional, usando la estructura de datos definida en la sección 4. Dicha estructura de datos ha sido también empleada para representar las cachés de primer y segundo nivel.

Como algoritmo de traducción se ha utilizado un algoritmo de pila monótono que aplica diversas podas a la traducción, entre ellas, limita a 8 el número máximo de traducciones inversas para cada secuencia de $f$.

Los Cuadros 2 y 3 muestran los resultados de traducción que se han obtenido con el corpus HANSARDS y EUROPARL. La primera fila de cada cuadro corresponde al caso base, y las siguientes, a experimentos en los que se trabaja con la arquitectura de caché con valores crecientes del parámetro $\alpha$. Para cada entrada

[4] La cantidad de memoria ha sido obtenida a partir de información en tiempo de ejecución que provee la plataforma en que fue lanzado el experimento, esta cifra puede exceder el coste teórico de la estructura dado en la sección 4 según las características de la política de asignación de memoria implementada por el sistema.

se dan: la cantidad de pares cargados en memoria, el coste temporal por frase en segundos, la cantidad de fallos de caché, el porcentaje de fallos de caché, la sobrecarga temporal producida por los accesos a disco a lo largo de *todo* el proceso de traducción, y por último, las medidas de calidad de la traducción $WER$ y *Bleu*.

| $\alpha$ | **secuencias** | **segsXfrase** | **FallosC** | **FallosC %** | **sobrec.** | **WER** | **Bleu** |
|---|---|---|---|---|---|---|---|
| **-** | 8750177 | 0.472 | - | - | - | 61.1 | 0.222 |
| **0** | 0 | 0.406 | 135400 | 1.334 | 205 | 61.1 | 0.222 |
| **10** | 2163029 | 0.696 | 89102 | 0.878 | 480 | 61.1 | 0.222 |
| **20** | 3349551 | 0.360 | 84574 | 0.833 | 144 | 61.1 | 0.222 |
| **30** | 4419806 | 0.349 | 83022 | 0.818 | 134 | 61.1 | 0.222 |
| **40** | 5298657 | 0.342 | 82101 | 0.809 | 125 | 61.1 | 0.222 |
| **50** | 6078112 | 0.342 | 81426 | 0.802 | 124 | 61.1 | 0.222 |
| **60** | 6764646 | 0.339 | 80993 | 0.798 | 122 | 61.1 | 0.222 |
| **70** | 7423550 | 0.337 | 80543 | 0.793 | 118 | 61.1 | 0.222 |
| **80** | 7900145 | 0.334 | 80135 | 0.789 | 117 | 61.1 | 0.222 |
| **90** | 8371058 | 0.331 | 79814 | 0.786 | 113 | 61.1 | 0.222 |
| **100** | 8750177 | 0.215 | 0 | 0.000 | 0 | 61.1 | 0.222 |

**Cuadro 2.** Experimentos de traducción variando el parámetro $\alpha$ para el corpus HANSARDS

Como puede observarse, la nueva técnica reduce el coste temporal por frase con respecto al caso base en todos los casos, siendo de especial interés aquél en el que $\alpha$ es igual a cero, debido a que no requiere cargar ningún par de secuencias en memoria (salvo, de forma temporal, los pares requeridos por cada frase en la caché de primer nivel). La razón de que se obtengan unos resultados tan satisfactorios reside en las reducidas tasas de fallos de caché, que salvo para $\alpha$ igual a cero, son inferiores al 1 %. No obstante, los fallos de caché producen una sobrecarga apreciable al acceder a disco (para $\alpha = 0$ el traductor pasa 205 segundos accediendo a disco para traducir mil frases), por tanto, las mejoras producidas por el uso de caché se deben a que este sobrecoste se ve compensado por un aumento en la eficiencia a la hora de acceder a los pares, debido a que la caché de primer nivel siempre contendrá una pequeñísima fracción del modelo completo. Creemos que estas mejoras siempre se producirán independientemente de cuál sea la estructura de datos escogida para implementar el caso base, ya que al tener que almacenar el modelo completo en memoria, siempre se tendrá que sacrificar en cierta medida algo de eficiencia temporal para reducir la complejidad espacial, ésto es, al menos, lo que ocurre con la estructura propuesta en [6], y en menor medida, con la que se presentaba en la sección 4.

Es importante destacar también que a medida que incrementamos el valor de $\alpha$ se va reduciendo la tasa de fallos de caché, repercutiendo favorablemente en el tiempo de ejecución. No obstante, se detecta una cierta variabilidad en el coste de los accesos a disco que hace que una disminución de la tasa de fallos de caché no se traduzca en una mejora del coste por frase, los motivos de ello no han sido aclarados. También es de interés resaltar que un determinado porcentaje

| $\alpha$ | **secuencias** | **segsXfrase** | **FallosC** | **FallosC %** | **sobrec.** | **WER** | **Bleu** |
|---|---|---|---|---|---|---|---|
| **-** | 40428266 | 2.877 | - | - | - | 68.454 | 0.171 |
| **0** | 0 | 0.515 | 336004 | 0.859 | 398 | 68.454 | 0.171 |
| **10** | 10917030 | 0.561 | 198723 | 0.508 | 437 | 68.454 | 0.171 |
| **20** | 16271419 | 0.556 | 185876 | 0.475 | 414 | 68.456 | 0.171 |
| **30** | 20915266 | 0.523 | 180227 | 0.461 | 345 | 68.456 | 0.171 |
| **40** | 24935576 | 0.521 | 177505 | 0.454 | 343 | 68.456 | 0.171 |
| **50** | 28404889 | 0.508 | 175655 | 0.449 | 316 | 68.456 | 0.171 |
| **60** | 31476767 | 0.506 | 174345 | 0.445 | 309 | 68.456 | 0.171 |
| **70** | 34017015 | 0.551 | 173346 | 0.443 | 394 | 68.456 | 0.171 |
| **80** | 36398738 | 0.642 | 172549 | 0.441 | 580 | 68.456 | 0.171 |
| **90** | 38413502 | 0.762 | 171756 | 0.439 | 760 | 68.456 | 0.171 |
| **100** | 40428266 | 0.356 | 0 | 0.000 | 0 | 68.458 | 0.171 |

**Cuadro 3.** Experimentos de traducción variando el parámetro $\alpha$ para el corpus EUROPARL

de los fallos de caché no se podrá nunca eliminar debido a que corresponde a la búsqueda de pares de secuencias no presentes en el modelo (para HANSARDS se buscaron 77968 pares no presentes en el modelo, mientras que para EUROPARL esa cifra era igual a 164726.)

Es particularmente interesante el hecho de que la reducción de la tasa de fallos de caché más importante, se produce al pasar de $\alpha = 0$ a $\alpha = 10$ (reducción del 65 %), lo que respalda empíricamente el primer principio de localidad expuesto en la sección 6 y justifica el uso de la caché de segundo nivel. El menor coste por frase se produce cuando $\alpha$ es igual a 100, porque eliminamos el sobrecoste de los fallos de caché, manteniendo la eficiencia de la caché de primer nivel.

## 8. Conclusiones

En el artículo se ha propuesto un conjunto de técnicas que dan una solución a problemas de escalado que están empezando a ser la tónica en el trabajo con modelos de secuencias. Dichas técnicas se basan en el trabajo con pequeñas porciones de los modelos tanto en entrenamiento como en traducción.

En el apartado del entrenamiento se ha presentado una técnica que permite entrenar el corpus EUROPARL con unos requerimientos de memoria RAM muy reducidos. Dicha técnica, basada en el procesado del corpus a trozos, permite incluso mejorar el coste temporal del caso base, lo que se debe al hecho de que los submodelos que se generan se pueden manejar más rápidamente que el modelo completo. Además de lo anterior, pensamos que es posible paralelizar el algoritmo.

En el apartado de la búsqueda, el trabajo con una arquitectura inspirada en memoria caché permite traducir sin necesidad de cargar el modelo de secuencias en memoria (o cargando sólo una pequeña parte). Los accesos a disco necesarios para que la técnica funcione, introducen una sobrecarga temporal moderada. No

obstante, y dado que la tasa de fallos de caché es extremadamente reducida, esta sobrecarga se ve contrarrestada por el hecho de que la parte del modelo que se carga en memoria, debido a su reducido tamaño, puede accederse mucho más rápidamente. Como consecuencia de lo anterior, la nueva técnica ha permitido *reducir* el tiempo de traducción por frase con respecto al caso base.

La arquitectura de modelos inspirada en caché se ha demostrado aplicable en traducción con algoritmos de pila, y pensamos que también debería serlo con algoritmos de programación dinámica. Además hay otras situaciones en las que también podría ser eficiente, por ejemplo, en la obtención de la mejor segmentación a nivel de secuencia [7], en donde parece que los principios de localidad descritos también son aplicables. También se podría exportar esta técnica a los modelos de lenguaje.

Se pueden plantear extensiones del trabajo presentado tanto en el apartado del entrenamiento como en el de traducción. En lo que respecta al entrenamiento, como mencionábamos más arriba, el algoritmo de entrenamiento a trozos es susceptible de ser paralelizado lo que permitiría alcanzar altas cotas de eficiencia. Con respecto a la traducción, sería de interés el trabajo con representaciones muy eficientes de la caché de primer nivel, dado que lo permite la escasa cantidad de pares que puede llegar a contener.

## Agradecimientos

Este trabajo ha sido parcialmente subvencionado por el proyecto CICYT TIC2003-08681-C02-02, la *Agencia Valenciana de Ciencia y Tecnología* dentro del contrato GRUPOS03/031, la *Generalitat Valenciana*, y el proyecto HERMES (Vicerrectorado de Investigación - UCLM-05/06).

## Referencias

1. Brown, P.F., Della Pietra, S.A., Della Pietra, V.J., Mercer, R.L.: The mathematics of statistical machine translation: Parameter estimation. Computational Linguistics **19**(2) (1993) 263–311
2. Tomás, J., Casacuberta, F.: Monotone statistical translation using word groups. In: Procs. of the MT Summit VIII, Santiago de Compostela, Spain (2001) 357–361
3. Marcu, D., Wong, W.: A phrase-based, joint probability model for statistical machine translation. In: Proc. of the EMNLP, Philadelphia, USA (2002) 1408–1414
4. Zens, R., Och, F., Ney, H.: Phrase-based statistical machine translation. In: Advances in artificial intelligence. 25. Annual German Conference on AI. Volume 2479 of Lecture Notes in Computer Science. Springer Verlag (2002) 18–32
5. Koehn, P., Och, F.J., Marcu, D.: Statistical phrase-based translation. In: Proceedings of the HLT/NAACL, Edmonton, Canada (2003)
6. Callison-Burch, C., Bannard, C., Schroeder, J.: Scaling phrase-based statistical machine translation to larger corpora and longer sentences. In: Proc. of the 43rd Annual Meeting of the ACL, Ann Arbor (2005) 255–262
7. García-Varea, I., Ortiz, D., Nevado, F., Gómez, P., Casacuberta, F.: Automatic segmentation of bilingual corpora: A comparison of different techniques. In: Procs.

of the 2nd IbPRIA. Volume 3523 of LNCS. Springer-Verlag, Estoril (Portugal) (2005) 614–621
8. Tomás, J.: Traducción automática de textos entre lenguas similares utilizando métodos estadísticos. PhD thesis, UPV, Valencia (Spain) (2003) (In spanish).
9. Knuth, D.E. In: Sorting and Searching. Second edn. Volume 3 of The Art of Computer Programming. Addison-Wesley, Reading, Massachusetts (1973)

# Búsqueda de alineamientos en traducción automática estadística: Un nuevo enfoque basado en un EDA

Luis Rodríguez, Ismael García-Varea y José A. Gámez

Departamento de Sistemas Infomáticos, UCLM
{luisr, ivarea, jgamez}@info-ab.uclm.es

**Resumen** Dentro de la traducción automática estadística, un alineamiento establece una correspondencia entre las palabras de la frase origen y las de la frase destino. Los alineamientos se utilizan, por un lado, durante el proceso de aprendizaje de los modelos estadísticos y, por otro, durante la búsqueda de la mejor traducción posible para una frase de entrada concreta. En ambos casos, la calidad del proceso final de traducción se ve notablemente influida por la calidad de dichos alineamientos. En el presente artículo, se propone un algoritmo para llevar a cabo la búsqueda de alineamientos entre dos frases. Dicho algoritmo está basado en los algoritmos de estimación de distribuciones de probabilidad. El algoritmo aquí propuesto se ha aplicado sobre distintas tareas considerando distintos pares de lenguas. En concreto, se ha experimentado con las tareas propuestas en los congresos internacionales HLT-NAACL 2003 y ACL-2005, en los cuales este algoritmo ha conseguido mejorar los resultados presentados por los diferentes sistemas participantes en dichas tareas.

## 1. Introducción

Actualmente, la aproximación estadística a la traducción automática constituye uno de los enfoques más prometedores dentro de este campo. Dicha aproximación se basa fundamentalmente en el aprendizaje de un modelo estadístico a partir de un corpus paralelo. Un corpus paralelo es un conjunto de pares de frases, cada uno de los cuales contiene una frase en un lenguaje origen junto con su correspondiente traducción en otro lenguaje destino. Los alineamientos a nivel de palabra son necesarios para establecer una correpondencia entre las palabras de ambas frases. La importancia de estos alineamientos radica en el hecho de que constituyen la base de los modelos estadísticos de traducción, específicamente de los ampliamente utilizados modelos de de IBM [1]. En los últimos años, se han propuesto diferentes competiciones internacionales donde se planteaban tareas compartidas de búsqueda de alineamientos ([2],[3]).

En el presente artículo, se propone una nueva aproximación al problema de la búsqueda de alineamientos. Concretamente, nos centraremos en la búsqueda del alineamiento óptimo, a nivel de palabra, entre una frase origen y una

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 843-854, 2006.

frase destino. Debido al hecho de que no existe un método eficiente que garantice la consecución de dicho alineamiento óptimo (llamado alineamiento de *Viterbi*) cuando se trabaja con modelos de IBM superiores al modelo 2 (que son precisamente los modelos que mejores resultados proporcionan en traducción automática con modelos de palabra), en el presente trabajo proponemos el uso de una metaheurística de reciente aparición. Dicha metaheurística es conocida como *Algoritmos de estimación de distribuciones (EDA, Estimation of Distribution Algorithms)*. A pesar de que el empleo de estos algoritmos no puede garantizar la obtención del alineamiento óptimo, esperamos que las soluciones que pueda alcanzar la propuesta que aquí se describe presenten unos niveles de calidad considerables en la mayoría de los casos. Esta expectativa está basada en los resultados obtenidos cuando se ha aplicado dicha técnica en problemas similares y ampliamente descritos en la literatura [4], además de en los resultados presentados aquí.

El presente trabajo se estructura de la siguente forma. En primer lugar, se presenta el concepto de alineamientos estadísticos en la sección 2. En la sección 3 se describen los algoritmos de estimación de distribuciones (EDAs). La implementación de un algoritmo basado en un EDA para la búsqueda de alineamientos se discute en la sección 4. Los experimentos llevados a cabo son descritos en la sección 5 . Finalmente, la sección 6 está dedicada a describir las conclusiones así como el trabajo futuro a realizar.

## 2. Traducción automática estadística. Alineamientos

En esta sección, introduciremos en primer lugar la aproximación estadística a la traducción automática, con el objeto de encuadrar posteriormente y de manera apropiada el problema del alineamiento dentro de dicho marco.

La aproximación estadística a la traducción automática puede describirse de la siguiente forma. Dada una frase $\mathbf{f} = f_1^J = f_1 \dots f_J$ en un lenguaje fuente se desea encontrar una traducción $\mathbf{e} = e_1^I = e_1 \dots e_I$ a dicha frase en un lenguaje destino. Cada posible frase $\mathbf{e}$ del lenguaje destino se considera una posible traducción de $\mathbf{f}$, con probabilidad $Pr(\mathbf{e}|\mathbf{f})$. Como traducción final, se escogerá aquella frase que maximice esta probabilidad. Aplicando el teorema de *Bayes*, tenemos:

$$\hat{\mathbf{e}} = \underset{e}{\operatorname{argmax}}\, Pr(\mathbf{e}|\mathbf{f}) = \underset{e}{\operatorname{argmax}}\{Pr(\mathbf{e}) \cdot Pr(\mathbf{f}|\mathbf{e})\} \tag{1}$$

donde $Pr(\mathbf{f}|\mathbf{e})$ representa el modelo de traducción, que establece la probabilidad de que la frase origen $\mathbf{f}$ se haya generado a partir de $\mathbf{e}$ (en este modelo, el proceso de traducción se enfoca desde la perspectiva inversa, es decir es $\mathbf{e}$ quien genera a $\mathbf{f}$) , y $Pr(\mathbf{e})$ representa el modelo de lenguaje, que indica la probabilidad de $e$ en el lenguaje destino.

Pueden establecerse diferentes modelos para estimar la probabilidad $Pr(\mathbf{f}|\mathbf{e})$. Dentro de la literatura, los más referenciados son los modelos de IBM[5]. Dichos modelos se basan en establecer relaciones estructurales entre las palabras de las frases origen y destino, utilizando para ello el concepto de alineamiento. Un

alineamiento es una correspondencia $a : j \rightarrow i = a_j$, que se establece entre posiciones $j$ en la frase de origen y posiciones $i$ en la frase destino. Los modelos estadísticos de alineamiento son similares al concepto de Modelos Ocultos de Markov (HMM) empleados en reconocimiento automático del habla.

La introducción del concepto de alineamiento permite reescribir la probabilidad del modelo de traducción obtenido en la ecuación (1) como $Pr(\mathbf{f}, \mathbf{a}|\mathbf{e})$. La forma en la que se tratan los alineamientos dentro del modelo varía según el modelo en sí que se esté considerando. En el caso de los modelos 1 y 2 de IBM, el alineamiento es considerado una variable oculta del modelo, obteniéndose así:

$$Pr(\mathbf{f}|\mathbf{e}) = \sum_{\mathbf{a} \in A} Pr(\mathbf{f}, \mathbf{a}|\mathbf{e}) \tag{2}$$

siendo $A$ el conjunto de todos los posibles alineamientos entre $\mathbf{f}$ y $\mathbf{e}$.

En los modelos IBM 1 y 2, el cálculo de todos los posibles alineamientos puede realizarse de forma eficiente, por lo que es factible seguir la aproximación mostrada en la fórmula (2). En el caso de modelos más complejos, como los modelos IBM 3, 4 y 5 [1] , no existe un algoritmo que calcule el conjunto completo de alineamientos de forma eficiente, por lo que la estimación de $Pr(\mathbf{f}, \mathbf{a}|\mathbf{e})$ se realiza a partir de algoritmos que aproximan el alineamiento óptimo entre las dos frases (realmente, en estos modelos se utiliza un conjunto de alineamientos obtenidos a partir de dicha aproximación).

Una vez introducido el concepto de alineamiento y su relación con los modelos estadísticos de traducción, podemos definir el problema tratado en el presente trabajo. Dada una frase origen $\mathbf{f}$ y una frase destino $\mathbf{e}$, el problema a resolver consiste en encontrar el alineamiento óptimo entre ambas frases:

$$\hat{\mathbf{a}} = \underset{\mathbf{a} \in A}{\operatorname{argmax}} Pr(\mathbf{a}|\mathbf{f}, \mathbf{e}) = \underset{\mathbf{a} \in A}{\operatorname{argmax}} Pr(\mathbf{f}, \mathbf{a}|\mathbf{e}) \tag{3}$$

siendo $A$ el conjunto de todos los alineamientos posibles entre $\mathbf{f}$ y $\mathbf{e}$.

La transformación llevada a cabo en la ecuación (3) nos permite afrontar el problema del alineamiento partiendo de los modelos de traducción de IBM previamente descritos. Antes de continuar, es necesario comentar que estos modelos imponen una fuerte restricción sobre el conjunto de posibles alineamientos. Esta restricción consiste en que cada palabra de la frase origen puede ser alineada como máximo con una palabra en la frase de destino. En la figura 1 se muestra un ejemplo de alineamiento a nivel de palabra siguiendo estas restricciones.

[1] En estos modelos, la probabilidad $Pr(\mathbf{f}, \mathbf{a}|\mathbf{e})$ se descompone en una serie de submodelos, cada uno de los cuales se estima a partir de los alineamientos entre frases. El primer submodelo es el modelo léxico, que relaciona dos palabras de los lenguajes origen y destino, asignando una probabilidad de traducción entre ellas. Por otra parte, el modelo de fertilidad modela el número de palabras de la frase $\mathbf{f}$ a partir de las cuales se ha generado una palabra concreta de la frase $\mathbf{e}$. Por último, el modelo de distorsión establece restricciones respecto a las posiciones que pueden ocupar en las frases $\mathbf{f}$ y $\mathbf{e}$ dos palabras entre las que se establezca un alineamiento. Además, dichos modelos introducen el concepto de la palabra vacía $e_0$ (NULL) en la frase $\mathbf{e}$ lo que permite la existencia de palabras en $\mathbf{f}$ no alineadas con ninguna palabra en $e$.

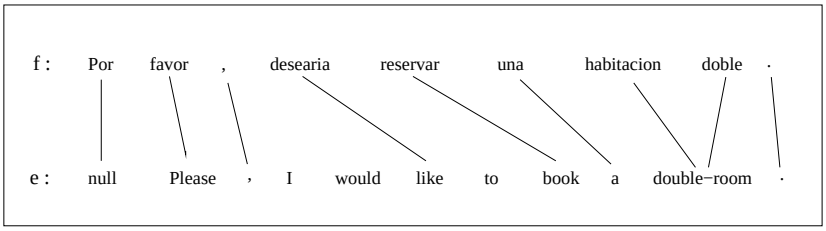



**Figura 1.** Ejemplo de alineamiento . De acuerdo con las restricciones impuestas por el modelo, cada palabra de la frase **f** está alineada a lo sumo con una palabra de la frase **e**. Además se introducen alineamientos con la palabra vacía $e_0$ (NULL) para indicar aquellas palabras de la frase **f** que no tienen correspondencia con palabras de la frase **e** (palabras espurias). Asimismo, puede observarse que existen palabras en la frase **e** que no están alineadas con ninguna de la frase **f** (palabras de fertilidad cero).

Debido a la limitación antes mencionada, suele ser necesario extender los alineamientos obtenidos con estos modelos (alineamientos unidireccionales) con el objetivo de alcanzar alineamientos más naturales (alineamientos bidireccionales). Esta extensión se lleva a cabo normalmente, mediante el cómputo de los alineamientos en ambas direcciones (esto es, de **f** a **e** y luego de **e** a **f**) y combinando convenientemente ambos conjuntos en uno solo.

## 3. Algoritmos de Estimación de Distribuciones

Los algoritmos de estimación de distribuciones (EDAs) [4] constituyen una metaheurística que ha suscitado gran interés durante los últimos años debido a su buen comportamiento en la resolución de problemas de optimización combinatoria. Los EDAs, al igual que los algoritmos genéticos, son algoritmos evolutivos basados en poblaciones pero, en vez de utilizar operadores genéticos, están basados en la estimación y posterior muestro de una distribución de probabilidad, la cual relaciona las variables o genes que constituyen la representación de las soluciones al problema. De esta forma, las relaciones de dependencia/independencia entre las variables del problema pueden ser modeladas explícitamente dentro de este marco. El esquema básico de un EDA canónico se muestra en la figura 2.

Como puede apreciarse, el algoritmo mantiene una población de $m$ individuos durante la búsqueda. Un individuo representa una solución potencial al problema a resolver. En este caso, cada uno de los individuos representa un posible alineamiento. Normalmente, en problemas de optimización combinatoria, un individuo se codifica mediante un vector de enteros $\mathbf{a} = \langle a_1, \ldots, a_J \rangle$, donde cada posicion $a_j$ puede tomar un conjunto finito de valores $\Omega_{a_j} = \{0, \ldots, I\}$. El primer paso dentro de un algoritmo evolutivo consiste en generar la población inicial $D_0$. A pesar de que $D_0$ es generada normalmente de forma aleatoria (para asegurar diversidad), la incorporación de conocimiento a priori acerca del problema específico puede ser de utilidad en este paso.

1. $D_0 \leftarrow$ Generar la población inicial ($m$ individuos)
2. Evaluar la población $D_0$
3. $k = 1$
4. Repetir
   (a) $D_{tra} \leftarrow$ Seleccionar $s \leq m$ individos de $D_{k-1}$
   (b) Estimar un nuevo modelo $\mathcal{M}$ a partir de $D_{tra}$
   (c) $D_{aux} \leftarrow$ Muestrear $m$ individuos de $\mathcal{M}$
   (d) Evaluar $D_{aux}$
   (e) $D_k \leftarrow$ Seleccionar $m$ individuos de $D_{k-1} \cup D_{aux}$
   (f) $k = k + 1$

   Hasta alcanzar el criterio de finalización

**Figura 2.** Esquema de un EDA canónico

Una vez que se ha generado la población inicial, es necesario evaluarla, esto es, medir la calidad de la solución representada por cada individuo con respecto al problema que estamos resolviendo. Para ello se utiliza como función de *fitness*, $f(\mathbf{a}) = log(Pr(\mathbf{f}, \mathbf{a}|\mathbf{e}))$ (ver Eq. 4), para asignar una puntuación o *score* a los individuos. Los algoritmos evolutivos en general, y los EDAs en particular, tratan de mejorar la calidad de los individuos dentro de la población durante el proceso de búsqueda. Con este propósito, en los algoritmos genéticos, se construye la nueva población copiando ciertos individuos de la actual y generando algunos nuevos a partir de éstos, de manera que aquellos que contengan soluciones de mayor calidad tendrán más probabilidad de ser escogidos para formar parte de esta nueva población.

En los EDAs, la transición entre poblaciones se realiza de forma completamente diferente. En este caso, el proceso consiste en resumir las propiedades de los individuos de la población, mediante el aprendizaje de una distribución de probabilidad que los describa lo más fielmente posible. Debido al hecho de que se intenta mejorar la calidad de la población en cada paso, sólo los $s$ mejores individuos (aquellos que presenten un mejor valor de la función de *fitness*) serán utilizados durante la estimación de dicha distribución de probabilidad $Pr(a_1, \ldots, a_J)$, tratando de esta forma de descubrir patrones comunes entre los mejores individuos. Una vez estimada dicha distribución de probabilidad, se procede a su muestreo para obtener nuevos individuos. Estos nuevos individuos, serán evaluados mediante la función de *fitness* y se sumarán a los que conforman la población actual, extendiéndose así dicha población. Finalmente, la nueva población se obtendrá mediante la selección de individuos de esta población extendida. Dicha selección se lleva acabo normalmente aplicando algún tipo de criterio *elitista* que asegure que las soluciones prometedoras se mantendrán durante el proceso de búsqueda.

El principal problema a afrontar durante este proceso consiste precisamente en la estimación de la distribución de probabilidad, debido al hecho de que la estimación de la probabilidad conjunta $Pr(a_1, \ldots, a_J)$ no es factible en la

mayoría de los casos. En la práctica, lo que verdaderamente se estima es un modelo probabilístico consistente en una aproximación de la factorización de la distribución anterior. Llegados a este punto, se pueden considerar diferentes niveles de complejidad en dicha factorización, desde distribuciones univariadas hasta distribuciones $n$-variadas o Redes Bayesianas [4, Capítulo 3]. En el presente artículo, al ser la primera aproximación al problema de los alineamientos mediante EDAs, amén de otras cuestiones que se discutirán más adelante, se ha optado por el modelo más simple de EDA: el conocido como UMDA (*Univariated Marginal Distribution Algorithm*, [6]). En el UMDA se asume que todas las variables son marginalmente independientes. Por ello, la distribución de probabilidad $n$-dimensional $Pr(a_1, \ldots, a_J)$ se factoriza como el producto de $J$ distribuciones unidimensionales (marginales): $\prod_{j=1}^{J} Pr(a_j)$. Entre las ventajas del UMDA pueden citarse: la posibilidad de utilizar conjuntos de datos reducidos para estimar la distribución, la no necesidad de aprendizaje estructural, la facilidad de muestreado de la distribución, etc.

## 4. Diseño de un EDA para la búsqueda de alineamientos

En esta sección, se describirá el diseño de un EDA para afrontar el problema de la búsqueda del alineamiento óptimo entre dos frases.

### 4.1. Representación

Uno de los aspectos más importantes en la definición de un algoritmo de búsqueda consiste en la representación del espacio de soluciones del problema. En el problema aquí considerado, la búsqueda de un alineamiento óptimo entre una frase origen $\mathbf{f}$ y una frase destino $\mathbf{e}$, el espacio de búsqueda puede ser descrito como el conjunto de todos los posibles alineamientos entre dos frases. Debido a las restricciones impuestas por los modelos de IBM (consistentes básicamente en que una palabra de $\mathbf{f}$ puede estar alineada como mucho con una palabra de $\mathbf{e}$), la manera más natural de representar una solución al problema consiste en almacenar cada posible alineamiento en un vector $\mathbf{a} = a_1 \ldots a_J$, siendo $J$ la longitud de $\mathbf{f}$. Cada posición del vector puede tomar el valor de "0" para respresentar un alineamiento con la palabra NULL (es decir, una palabra de la frase origen que no está alineada con ninguna palabra de la frase destino) o un índice que represente cualquier posición de la frase destino (por ejemplo, el alineamiento mostrado en la figura 1, se representaría mediante el vector [0 1 2 5 7 8 9 9 10]).

### 4.2. Función de evaluación

Durante el proceso de búsqueda, cada individuo (hipótesis de búsqueda) es evaluado utilizando la función de *fitness* definida de la siguiente forma. Sea $\mathbf{a} = a_1 \cdots a_J$ el alineamiento representado por un individuo. Este alineamiento se evalúa mediante el cálculo de la probabilidad $Pr(\mathbf{f}, \mathbf{a}|\mathbf{e})$. Esta probabilidad se obtiene a partir del modelo 4 de IBM como:

$$p(\mathbf{f},\mathbf{a}|\mathbf{e}) = \sum_{(\tau,\pi)\in\langle\mathbf{f},\mathbf{a}\rangle} p(\tau,\pi|\mathbf{e})$$
$$\prod_{i=1}^{I} n(\phi_i|e_i) \times \prod_{i=1}^{I}\prod_{k=1}^{\phi_i} t(\tau_{ik}|e_i) \times$$
$$\prod_{i=1,\phi_i>0}^{I} d_{=1}(\pi_{i1} - c_{\rho_i}|\mathcal{E}_c(e_{\rho_i}), \mathcal{F}_c(\tau_{i1})) \times$$
$$\prod_{i=1}^{I}\prod_{k=2}^{\phi_i} d_{>1}(\pi_{ik} - \pi_{i(k-1)}|\mathcal{F}_c(\tau_{ik})) \times$$
$$\binom{J-\phi_0}{\phi_0} p_0^{J-2\phi_0} p_1^{\phi_0} \times \prod_{k=1}^{\phi_0} t(\tau_{0k}|e_0) \quad (4)$$

donde los factores separados por los símbolos × denotan las probabilidades de fertilidad, traducción y distorsión.[2].

Este modelo se entrenó mediante la herramienta GIZA++ [7] utilizando como datos de entrenamiento los corpus suministrados en las diferentes tareas de alineamiento descritas en la sección 5.1.

### 4.3. Búsqueda

En esta sección, se describirán algunos de los detalles específicos de la búsqueda. Como se mencionó en la sección 3, el algoritmo comienza generando un conjunto de hipótesis iniciales (población inicial). En este caso, lo que se genera es un conjunto de alineamientos (individuos) aleatorios entre las frases origen y destino. Una vez hecho esto, se procede a evaluar dichos individuos mediante la función de *fitness* descrita en la sección 4.2. En este instante, la búsqueda comienza siguiendo el esquema descrito en la sección 3.

El proceso de búsqueda finaliza cuando se cumple cierto criterio de finalización. En la implementación aquí considerada, el algoritmo finaliza la búsqueda cuando se alcanza un número determinado de iteraciones sin haber conseguido ninguna mejora en las hipótesis de búsqueda. Una vez finalizada la búsqueda, el mejor individuo de la última población obtenida es devuelto como solución final.

En cuanto al esquema de EDA utilizado, como se comentó anteriormente, se ha optado por la elección del UMDA debido principalmente al tamaño del espacio de búsqueda. Considerando que la longitud de cada individuo (solución) es $J$, y que cada posición puede tomar $(I+1)$ valores posibles, en el caso del UMDA,

[2] Los símbolos de esta fórmula denotan: $J$ (longitud de $\mathbf{e}$), $I$ (longitud de $\mathbf{f}$), $e_i$ (la $i$-ésima palabra de $e_1^I$), $e_0$ (la palabra NULL), $\phi_i$ (la fertilidad de $e_i$), $\tau_{ik}$ (la $k$-ésima palabra generada por $e_i$ en $\mathbf{a}$), $\pi_{ik}$ (la posición de $\tau_{ik}$ en $\mathbf{f}$), $\rho_i$ (la posición de la primera palabra fértil a la izquierda de $e_i$ en $\mathbf{a}$), $c_{\rho_i}$ (el máximo de la media de todas las $\pi_{\rho_i k}$ para $\rho_i$, o 0 si $\rho_i$ no está definido).

el número de parámetros a estimar para cada posición es $I$, por lo que el número total de parámetros a estimar es de $J * I$ (p.e, en la tarea Inglés-Francés descrita en la sección 5.1, *media*$(J)$=15 y *media*$(I)$=17.3). Si se hubiese optado por la adopción de modelos más complejos, el tamaño de las tablas de probabilidad hubiese crecido exponencialmente. Por ejemplo, al considerar un modelo bivariado, cada variable (posición) está condicionada por otra variable y, por tanto, las tablas de probabilidad $P(.|.)$ a estimar tendrían $I(I+1)$ parámetros. Para poder estimar convenientemente las distribuciones de probabilidad, el tamaño de las poblaciones debería incrementarse de forma notable. Como resultado de esto, los recursos computacionales requeridos por el algoritmo se incrementarían de forma dramática.

Además, como se describió en la sección 3, es necesario fijar ciertos parámetros a la hora de diseñar el EDA. Por una parte, debe definirse el tamaño de la población. En este caso, se ha escogido este tamaño de forma que sea proporcional a la longitud de las frases a ser alineadas. Concretamente, el tamaño de población escogido es igual a la longitud de la frase de entrada **f** multiplicada por un factor de diez.

Por otra parte, como ya se mencionó en la sección 3, la distribución de probabilidad sobre los individuos no se estima de la población completa. En la presente tarea, la estimación se realiza sobre el 20 % de la población global (evidentemente, este 20 % está formado por los mejores individuos de la población).

Por último, es necesario mencionar que, debido a que la función de *fitness* utilizada en el algoritmo permite solamente la obtención de alineamientos unidireccionales, los alineamientos finales se obtuvieron como se describe a continuación. En primer lugar, la búsqueda se llevó a cabo en las dos direcciones (esto es, de **f** a **e** y de **e** a **f**) para posteriormente combinar los resultados de ambos conjuntos de alineamientos, consiguiendo así alineamientos bidireccionales. Para ello, se experimentó con diferentes técnicas de combinación de alineamientos, obteniéndose los mejores resultados con el método propuesto en [8] que consiste básicamente en considerar todos los alineamientos pertenecientes a la intersección de ambos conjuntos, incluyendo además aquellos alineamientos de la unión que cumplan una serie de condiciones.

## 5. Experimentos

Se han llevado a cabo diferentes experimentos para comprobar el comportamiento del algoritmo propuesto sobre diferentes tareas. A continuación, se describirá la metodología empleada así como los resultados obtenidos.

### 5.1. Corpus y evaluación

Los experimentos se han realizado utilizando dos corpus y tres conjuntos de test diferentes. Todos ellos han sido obtenidos de las dos tareas propuestas en HLT/NAACL 2003 [2] y en ACL 2005 [3]. En estas dos tareas estaban involucrados cuatro pares de lenguajes, Inglés-Francés, Rumano-Inglés, Inglés-Inuktitut e

Inglés-Hindi. En el presente trabajo se han considerado los pares Inglés-Francés y Rumano-Inglés (los lenguajes Inuktitut e Hindi no se pudieron abordar debido a la falta de tiempo suficiente par realizar preprocesos específicos sobre dichos lenguajes, lo que se antoja indispensable para obtener resultados suficientemente competitivos). En la tabla 1 se muestran las características de los corpus utilizados.

**Cuadro 1.** Características de los corpus utilizados en las diferentes tareas de alineamiento

| | **In-Fr** | **Ru-En 03** | **Ru-En 05** |
|---|---|---|---|
| **Frases de entrenamiento** | 1M | 97K | 97K |
| **Vocabulario** | 68K / 86K | 48K / 27K | 48K / 27K |
| **Palabras totales** | 20M / 23M | 1.9M / 2M | 1.9M / 2M |
| **Tamaño del conjunto de test** | 447 | 248 | 200 |

A la hora de evaluar la calidad de los alineamientos obtenidos, se emplearon diferentes métricas: *Precision* ($P_T$), *Recall* ($R_T$), *F-measure* ($F_T$)[7] , and *Alignment Error Rate* ($AER$) [2]. Dados un alineamiento $A$ y un alineamiento de referencia $G$ (ambos $A$ y $G$ pueden ser descompuestos en dos conjuntos $A_S$, $A_P$ y $G_S$, $G_P$, que representan alineamientos *Seguros* y *Probables* respectivamente). Estas medidas se definen de la siguiente manera:

$$\begin{aligned} P_T &= \frac{|A_T \bigcap G_T|}{|A_T|} \\ R_T &= \frac{|A_T \bigcap G_T|}{|G_T|} \\ F_T &= \frac{|2P_T R_T|}{|P_T + R_T|} \\ P_T &= \frac{|A_P \bigcap G_S| + |A_P \bigcap G_P|}{|A_P| + |G_S|} \\ AER &= \frac{1 - |A_S \bigcap G_S| + |A_P \bigcap G_P|}{|A_P| + |G_S|} \end{aligned} \tag{5}$$

donde $T$ hace referencia al tipo de alineamiento, pudiendo ser éste $S$ (seguro) o $P$ (probable).

Es importante destacar, por último, que debido al hecho de que los EDAs son algoritmos no deterministas, los resultados presentados en la sección 5.2 representan la media de diez ejecuciones del algoritmo de búsqueda.

### 5.2. Resultados

En los cuadros 2, 3 y 4 se muestran los resultados obtenidos con las diferentes tareas antes descritas. Los resultados obtenidos con la técnica propuesta en el presente artículo se comparan con los mejores resultados presentados en las tareas descritas en [2] y [3]. En dichos cuadros, se muestran la media y la varianza de diez ejecuciones del algoritmo. A tenor de las reducidas varianzas observadas en los resultados, puede concluirse que la naturaleza no determinista del algoritmo no es estadísticamente significativa.

**Cuadro 2.** Calidad de los alineamientos ( %) para la tarea Inglés-Francés

| System | $P_s$ | $R_s$ | $F_s$ | $P_p$ | $R_p$ | $F_p$ | AER |
|---|---|---|---|---|---|---|---|
| EDA | **74.13** | 84.57 | **79.01** | **80.70** | 33.50 | 47.35 | **14.88** ±0,03 |
| Ralign.EF1 | 72.54 | 80.61 | 76.36 | 77.56 | **36.79** | **49.91** | 18.50 |
| XRCE.Nolem.EF.3 | 55.43 | **93.81** | 69.68 | 72.01 | 36.00 | 48.00 | 21.27 |

**Cuadro 3.** Calidad de alineamientos ( %) para la tarea Rumano-Inglés 2003

| System | $P_s$ | $R_s$ | $F_s$ | $P_p$ | $R_p$ | $F_p$ | AER |
|---|---|---|---|---|---|---|---|
| EDA | **94.22** | 49.67 | 65.05 | **76.66** | 60.97 | **67.92** | **32.08** ±0,05 |
| XRCE.Trilex.RE.3 | 80.97 | 53.64 | 64.53 | 63.64 | **61.58** | 62.59 | 37.41 |
| XRCE.Nolem-56k.RE.2 | 82.65 | **54.12** | **65.41** | 61.59 | 61.50 | 61.54 | 38.46 |

**Cuadro 4.** Calidad de alineamientos( %) para la tarea Rumano-Inglés 2005

| System | $P_s$ | $R_s$ | $F_s$ | $P_p$ | $R_p$ | $F_p$ | AER |
|---|---|---|---|---|---|---|---|
| EDA | **95.37** | 54.90 | 69.68 | 80.61 | 67.83 | **73.67** | **26.33** ±0,044 |
| ISI.Run5.vocab.grow | 87.90 | 63.08 | **73.45** | 87.90 | 63.08 | 73.45 | 26.55 |
| ISI.Run4.simple.intersect | 94.29 | 57.42 | 71.38 | **94.29** | 57.42 | 71,38 | 28.62 |
| ISI.Run2.simple.union | 70.46 | **71.31** | 70.88 | 70.46 | **71.31** | 70.88 | 29.12 |

De acuerdo con lo que muestran estos resultados, podemos concluir que la técnica de búsqueda propuesta basada en un EDA se muestra muy competitiva con respecto a los mejores resultados presentados en las diferentes tareas de alineamiento propuestas.

Como complemento a estos resultados, se realizó una experimentación adicional con el objeto de evaluar el comportamiento del algoritmo desde una pers-

pectiva de búsqueda únicamente. Para ello, se estableció una distinción entre los errores producidos por el propio proceso de búsqueda y los errores producidos por el modelo que guía la búsqueda (esto es, los errores provocados por la función de *fitness*). Con este fin, se siguió la siguiente aproximación. En primer lugar, los alineamientos de referencia (bidireccionales) fueron descompuestos en dos conjuntos diferentes de alineamientos unidireccionales (esta operación se realizó con el objeto de poder evaluar los alineamientos de referencia con los modelos de IBM, que, recordemos, no permiten alineamientos bidireccionales). Debido a que no existe un método exacto para llevar a cabo esta descomposición, se adoptó el método que se describe a continuación. Por cada alineamiento de referencia, se calcularon todas las descomposiciones posibles de dicho alineamiento en alineamientos unidireccionales, puntuando cada uno de ellos con la función de evaluación $F(\mathbf{a}) = p(\mathbf{f}, \mathbf{a}|\mathbf{e})$ descrita en la sección 4.2 y seleccionando el de mayor puntuación ($\mathbf{a}_{ref}$). A continuación, este alineamiento se comparó con la solucion obtenida por el EDA ($\mathbf{a}_{eda}$). Este proceso se llevó a cabo sobre todas las frases del test, midiendo el AER de ambos alineamientos así como el valor de la función de *fitness*. En este punto, consideramos que se produjo un error del modelo si $F(\mathbf{a}_{eda}) > F(\mathbf{a}_{ref})$. Por otra parte, consideramos que se produjo un error de búsqueda si $F(\mathbf{a}_{eda}) < F(\mathbf{a}_{ref})$. En el cuadro 5, se muestra un resumen para ambos tipos de errores en la tarea Rumano-Inglés 2005.

**Cuadro 5.** Compativa entre los alineamientos de referencia y los alineamientos obtenidos mediante el EDA. Se muestran los errores de búsqueda y del modelo. Además se muestran el AER para los alineamientos de referencia y del EDA. Estos resultados se han obtenido sobre la tarea Rumano-Inglés 05

| | **Rumano-Inglés** | **Inglés-Rumano** |
|---|---|---|
| **errores de búsqueda** | 35 (17.5 %) | 18 (9 %) |
| **errores del modelo** | 165 (82.5 %) | 182 (91 %) |
| **AER-EDA** | 29.67 % | 30.66 % |
| **AER-referencia** | 12.77 % | 11.03 % |

Estos experimentos muestran que la mayoría de los errores no fueron debidos al proceso de búsqueda en sí, sino a otros factores. De aquí puede concluirse que el modelo que guía la búsqueda es susceptible de ser mejorado, por un lado, y que se deberían explorar nuevas técnicas de simetrización, por otro.

## 6. Conclusiones y trabajo futuro

En el presente trabajo, se ha presentado una nueva aproximación al problema de la búsqueda de alineamientos en traducción automática, basada en el uso de un algoritmo de estimación de distribuciones (EDA). Los resultados obtenidos con esta técnica han demostrado ser suficientemente competitivos.

La naturaleza no determinista del algoritmo no constituye un verdadero problema, tal y como muestran los resultados aquí mostrados, demostrando que uno de los mayores inconvenientes de este tipo de técnicas no se traduce en un impacto significativo en los resultados obtenidos en esta tarea.

Finalmente, el trabajo aquí desarrollado está siendo aplicado al aprendizaje de los modelos de IBM descritos en la sección 2 con el objeto de comprobar el impacto de la mejora de los alineamientos sobre los modelos. Asimismo, se plantea la utilización de los alineamientos obtenidos mediante la técnica aquí descrita en el aprendizaje de modelos de segmentos. En ambos casos se plantea la realización de experimentos de traducción que reflejen la mejora de los alineamientos sobre las traducciones finales obtenidas.

## Agradecimientos

Este trabajo se ha desarrollado en el marco de los proyectos JCCM (PBI-05-022) y HERMES 05/06 (Vic. Inv. UCLM).

## Referencias

1. Brown, P.F., Cocke, J., Pietra, S.A.D., Pietra, V.J.D., Jelinek, F., Lafferty, J.D., Mercer, R.L., Roosin, P.S.: A statistical approach to machine translation. Comp. Linguistics **16**(2) (1990) 79–85
2. Mihalcea, R., Pedersen, T.: An evaluation exercise for word alignment. In Mihalcea, R., Pedersen, T., eds.: HLT-NAACL 2003 Workshop: Building and Using Parallel Texts: Data Driven Machine Translation and Beyond, Edmonton, Alberta, Canada, Association for Computational Linguistics (2003) 1–10
3. Joel Martin, Rada Mihalcea, T.P.: Word alignment for languages with scarce resources. In Mihalcea, R., Pedersen, T., eds.: Proceedings of the ACL Workshop on Building and Exploiting Parallel Texts: Data Driven Machine Translation and Beyond, Michigan, USA, Association for Computational Linguistics (2005) 1–10
4. Larrañaga, P., Lozano, J.: Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation. Kluwer Academic Publishers (2001)
5. Brown, P.F., Pietra, S.A.D., Pietra, V.J.D., Mercer, R.L.: The mathematics of statistical machine translation: Parameter estimation. Comp. Linguistics **19**(2) (1993) 263–311
6. Muhlenbein, H.: The equation for response to selection and its use for prediction. Evolutionary Computation **5**(3) (1997) 303–346
7. Och, F.J., Ney, H.: A systematic comparison of various statistical alignment models. Computational Linguistics **29**(1) (2003) 19–51
8. Och, F.J., Ney, H.: Improved statistical alignment models. In: ACL00, Hongkong, China (2000) 440–447

# Análisis teórico sobre las reglas de traducción directa e inversa en traducción automática estadística

Jesús Andrés Ferrer[1], Ismael García-Varea[2] y Francisco Casacuberta[1]

[1] Universidad Politécnica de Valencia
[2] Universidad de Castilla-la Mancha

**Resumen** En la formulación original de la traducción automática como un proceso estadístico se utiliza la regla de la traducción inversa, que modeliza la traducción con dos modelos: un *modelo de traducción inverso* y un *modelo de lenguaje destino*. No obstante, la mayoría de los sistemas de traducción automática estadística actuales hacen uso de la regla de traducción directa, es decir, utilizan un *modelo traducción directo* y un *modelo de lenguaje destino*. Según sus autores, el motivo principal de ese cambio ("heurístico") no es otro que simplificar el problema de la decodificación o búsqueda de la traducción óptima. Hasta ahora hubo un intento para explicar esta regla de traducción directa dentro del marco teórico de la *máxima entropía*, no obstante esta explicación no analiza las repercusiones estadísticas de estas reglas.
En este articulo presentamos un marco teórico basado en la teoría de la decisión, que permite formalizar la utilización de una regla u otra así como sus repercusiones estadísticas. Adicionalmente, este marco general proporciona otras reglas más potentes. Finalmente, la teoría y reglas propuestas se analizan en la práctica con una tarea de traducción simple y con un modelo también simple, con el objetivo de demostrar la validez de la teoría.

## 1. Introducción

La traducción automática (TA) aborda el problema de traducir automáticamente una frase $\mathbf{f}$ en un lenguaje origen ($\mathbf{F}^*$) a otra frase $\mathbf{e}$ en un lenguaje destino ($\mathbf{E}^*$). Debido a la complejidad de esta tarea, normalmente, no se puede definir un sistema experto (o sistema basado en reglas puramente lingüísticas) que lo resuelva. Por este motivo, los sistemas estadísticos son una opción en auge desde que a principios de los años 90 [1] se presentara el primer modelo puramente estadístico para resolver el problema de la traducción automática.

En el primer modelo estadístico de traducción automática (TA) (ver [1]), el problema de la traducción se analizaba como un problema clásico de reconocimiento de formas, utilizando para ello la extendida regla de clasificación de Bayes [2]. Por lo tanto, el problema de la traducción automática estadística

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 855-867, 2006.

(TAE) se debe analizar como un problema de clasificación donde el conjunto de clases son todas las frases posibles de la lengua destino ($\mathbf{E}^*$) sin tener en cuenta su estructura sintáctica. Es decir, cualquier frase compuesta por palabras del vocabulario destino ($\mathbf{e} \in \mathbf{E}^*$) puede ser una posible traducción de una frase de la lengua origen ($\mathbf{f}$). El objetivo consiste en buscar la traducción ($\hat{\mathbf{e}}$) que maximice la probabilidad a posteriori $p(\mathbf{e}|\mathbf{f})$:

$$\hat{\mathbf{e}} = \arg \max_{\mathbf{e} \in \mathbf{E}^*} \{p(\mathbf{e}|\mathbf{f})\} \tag{1}$$

donde $p(\mathbf{e}|\mathbf{f})$ se puede aproximar con un modelo estadístico de traducción *directo*. La Ec. (1) se conoce con el nombre de regla de clasificación óptima de Bayes[3]y es además la regla de clasificación correcta bajo ciertas asunciones.

La *regla de traducción inversa (RTI)* se obtiene aplicando el teorema de Bayes a la ecuación (1):

$$\hat{\mathbf{e}} = \arg \max_{\mathbf{e} \in \mathbf{E}^*} \{p(\mathbf{e}) \cdot p(\mathbf{f}|\mathbf{e})\} \tag{2}$$

La ecuación (2), inicialmente propuesta en Brown y colaboradores [1], implica que la frase destino seleccionada ($\hat{\mathbf{e}}$) es aquella que maximiza los dos modelos: el modelo de lenguaje destino ($p(\mathbf{e})$) y el modelo de traducción inverso ($p(\mathbf{f}|\mathbf{e})$). Obviamente, esta regla también es óptima si se conociesen las distribuciones de probabilidad reales; sin embargo, en la práctica las distribuciones se aproximan con modelos. La ventaja de esta aproximación subyace precisamente en este enfoque, puesto que permite modelar el proceso de traducción con dos modelos (el modelo inverso de traducción y el modelo de lenguaje) en lugar de uno sólo.

En la práctica, esta aproximación tiene un problema considerable: *el problema de la búsqueda*[4]. Esta búsqueda normalmente es un problema complejo, llegando incluso a ser un problema NP-duro[5] [3] (y por lo tanto, con un coste exponencial). Motivo por el cual se suele resolver de manera aproximada [4,5].

Con el objetivo de reducir este coste, muchos de los sistemas de traducción estadística actuales [6,7,8,9,10] proponen utilizar la *regla de traducción directa (RTD)*:

$$\hat{\mathbf{e}} = \arg \max_{\mathbf{e} \in \mathbf{E}^*} \{p(\mathbf{e}) \cdot p(\mathbf{e}|\mathbf{f})\} \tag{3}$$

Hasta este momento, esta regla, ec. (3), se ha analizado como una versión heurística de la RTI (Ec. (2)), donde $p(\mathbf{f}|\mathbf{e})$ se ha sustituido por $p(\mathbf{e}|\mathbf{f})$; o equivalentemente, donde se ha añadido un modelo de lenguaje a la regla óptima de Bayes, Ec. (1). La ventaja de esta regla consiste en el uso de un algoritmo de traducción que puede resolverse en tiempo polinómico. Precisamente, esto ha motivado su uso aunque sus consecuencias y/o ventajas desde un punto de vista estadístico se desconozcan; asumiendo que no se utiliza un modelo de traducción

[3] Asumiendo el conocimiento de la verdadera distribución de probabilidad de traducción $p(\mathbf{e}|\mathbf{f})$

[4] La forma de resolver la maximización de la optimización $\hat{\mathbf{e}}$ en el conjunto $\mathbf{E}^*$, es decir, $\arg\max_{\mathbf{e} \in \mathbf{E}^*}$

[5] Si se utilizan modelos asimétricos, y en especial modelos de alineamientos de IBM y modelos de $n$-gramas para modelizar el lenguaje destino.

simétrico[6]. En este sentido, algunos trabajos proporcionan argumentos en favor de esta regla, como por ejemplo [11].

Actualmente, los modelos de TAE más prácticos [10,12], utilizan una combinación lineal de modelos estadísticos para aproximar el modelo de traducción directo del siguiente modo:

$$p(\mathbf{e}|\mathbf{f}) \approx \frac{\exp\left[\sum_{m=1}^{M} \lambda_m h_m(\mathbf{f},\mathbf{e})\right]}{\sum_{\mathbf{e}'\in\mathbf{E}^*} \exp\left[\sum_{m=1}^{M} \lambda_m h_m(\mathbf{f},\mathbf{e}')\right]} \tag{4}$$

donde $h_m$ es un modelo estadístico logarítmico que aproxima una distribución de probabilidad (en este caso, una probabilidad de un modelo de traducción o de lenguaje).

Esta explicación es válida pero no aclara ninguna propiedad, solo define la ecuación como una instanciación de una técnica (como se hace en [10]). De hecho, es extraño debido a que las funciones de características ($h_m$) son aproximaciones a distribuciones de probabilidad, esto es *modelos estadísticos*. Es decir, una probabilidad $p(\mathbf{e}|\mathbf{f})$ se modeliza como la combinación de otros modelos que modelizan probabilidades, llegando incluso alguno de ellos a ser un modelo de la propia distribución $p(\mathbf{e}|\mathbf{f})$; circunstancia que no deja de extrañar desde un punto de vista estadístico. Por ejemplo, la probabilidad directa de traducción podría aproximarse por una mixtura logarítmica compuesta por un modelo directo y un modelo inverso, esto es $p(\mathbf{e}\,|\,\mathbf{f}) \propto \exp(\lambda_1 \log \tilde{p}(\mathbf{e}\,|\,\mathbf{f}) + \lambda_2 \log \tilde{p}(\mathbf{f}\,|\,\mathbf{e}))$. En definitiva, esto no deja de ser sorprendente, puesto que se supone que $\tilde{p}(\mathbf{e}\,|\,\mathbf{f})$ modeliza $p(\mathbf{e}\,|\,\mathbf{f})$.

Desde esta perspectiva, la RTD resulta ser una instanciación de la técnica de Máxima Entropía (ME) [13]. No obstante, en [13] sólo se explica la idoneidad de la técnica de ME para entrenar los pesos de una combinación log-lineal de modelos estadísticos. Es decir, esta aproximación no tiene relación directa con la regla de clasificación y no explica las consecuencias de aplicar un modelo log-linear, la RTI o la RTD.

En definitiva un sistema de traducción estadístico clásico se compone de tres módulos:

- un modelo de traducción: que puede ser directo ($p(\mathbf{e}|\mathbf{f})$) pudiéndose obtener una búsqueda resoluble en tiempo polinómico; o que puede ser inverso ($p(\mathbf{f}|\mathbf{e})$) complicando generalmente el problema de la búsqueda llegando incluso a transformase en un algoritmo exponencial.
- un Modelo de Lenguaje destino: que normalmente es un modelo de $n$-gramas ($p(\mathbf{e})$);
- y un algoritmo de búsqueda: encargado de resolver el problema de la maximización ($\arg\max_{\mathbf{e}\in\mathbf{E}^*}$), que en principio es la razón más importante para usar la RTD en lugar de la RTI.

[6] Dadas dos frases $\mathbf{e}$ y $\mathbf{f}$ de una lengua destino y origen respectivamente: un modelo *simétrico* asigna la misma probabilidad a $p(\mathbf{e}|\mathbf{f})$ y a $p(\mathbf{f}|\mathbf{e})$; y análogamente un modelo *asimétrico* no lo hace.

El objetivo de este trabajo, es formalizar y explicar cómo se obtienen estas reglas bajo una visión estadística. Es más, desde este enfoque, se puede proponer un marco generalizado replanteando el problema como un problema de optimización de funcional.

El trabajo se organiza de la siguiente manera: la sección 2 resume la teoría de la decisión de Bayes obteniéndose la regla oprima de clasificación de Bayes. En esta sección se analiza la traducción estadística como un caso específico, obteniéndose las diferentes reglas de traducción. La sección 4, donde se muestran algunos experimentos preliminares, demuestra de forma empírica la teoría expuesta. Finalmente, se resumen las conclusiones más importantes en la sección 5.

## 2. Teoría de la decisión de Bayes

Un problema de clasificación, como la traducción automática estadística, es una instancia de un problema de decisión. En esta sección abordaremos los problemas de clasificación desde la perspectiva de la teoría de decisión de Bayes.

En primer lugar analizaremos en que consiste un problema de clasificación bajo el marco de la teoría de decisión. En general, un problema de clasificación se compone de tres elementos:

1. Un conjunto de *objetos* ($\mathcal{X}$) que el sistema puede observar en algún momento y que tendrá que clasificar.
2. Un conjunto de *clases* ($\Omega = \{\omega_1, \ldots, \omega_C\}$) en las que el sistema deberá clasificar cada objeto observado $\mathbf{x} \in \mathcal{X}$.
3. Una *función de pérdida* ($\mathrm{l}(\omega_k|\mathbf{x}, \omega_j)$). Esta función evalúa la pérdida cometida al clasificar un objeto observado $\mathbf{x}$ en una clase, $\omega_k \in \Omega$, de entre todas las posibles clases ($\Omega$), sabiendo que la *clase óptima* para el objeto $\mathbf{x}$ es $\omega_j \in \Omega$.

Cuando el sistema de clasificación observa un objeto $x \in \mathcal{X}$ elige la clase que considera correcta; en el sentido de aquella clase para la cual la función de pérdida sea mínima, es decir, aquella que minimice la pérdida en que el sistema incurriría si se produjese un error. Una función de pérdida ampliamente utilizada debido a su simplicidad, es la *función* 0-1, la cual se define como:

$$\mathrm{l}(\omega_k|\mathbf{x}, \omega_j) = \begin{cases} 0 & \omega_k = \omega_j \\ 1 & \text{en otro caso} \end{cases} \tag{5}$$

Esta función no penaliza la clase correcta, sin embargo, tampoco distingue entre la importancia de clasificar un objeto en una u otra clase incorrecta; es decir, la penalización que recibe el sistema al clasificar el objeto $\mathbf{x}$ en la clase $\omega_i$ o $\omega_j$ es la misma independientemente del propio objeto, de la clase correcta o de la clase propuesta. Esta aproximación sólo tiene sentido en casos sencillos con un número de clases finito y pequeño. Por ejemplo, si se tratase de un sistema de diagnóstico de enfermedades, no tiene las mismas consecuencias considerar a una persona sana como enferma que viceversa. Las repercusiones de esta decisión errónea

depende de una serie de factores como el tipo de enfermedad, la mortalidad de la enfermedad, la agresividad del posible tratamiento para dicha enfermedad, etc.

El elemento primordial en el diseño de un sistema de clasificación es *la función de clasificación*, es decir $\mathrm{c} : \mathcal{X} \rightarrow \Omega$. Esta función determina la clase en la que el sistema debe clasificar un objeto observado $\mathbf{x}$, y por supuesto, debe tener en cuenta la pérdida o riesgo que puede cometer dependiendo de la función de pérdida. En consecuencia, el sistema de clasificación produce un riesgo, denominado *el riesgo global*, que se define como:

$$R(\mathrm{c}) = \mathrm{E}_{\mathbf{x}}[R(\mathrm{c}(\mathbf{x})|\mathbf{x})] = \int_{\mathcal{X}} R(\mathrm{c}(\mathbf{x})|\mathbf{x})\, \mathrm{p}(\mathbf{x}) d\mathbf{x} \tag{6}$$

donde $R(\omega_k|\mathbf{x})$ (con $\omega_k = \mathrm{c}(\mathbf{x})$) es el *riesgo condicional dado* $\mathbf{x}$, es decir, la pérdida esperada al clasificar según el criterio de la función de clasificación. El mencionado *riesgo condicional* se define como:

$$R(\omega_k|\mathbf{x}) = \mathrm{E}[\mathrm{l}(\omega_k|\mathbf{x}, \omega_j)] = \sum_{\omega_j \in \Omega} \mathrm{l}(\omega_k|\mathbf{x}, \omega_j)\, \mathrm{p}(\omega_j|\mathbf{x}) \tag{7}$$

La regla de clasificación de Bayes es aquella regla que minimiza el riesgo global. Dado que minimizar el riesgo condicional para cada objeto ($\mathbf{x}$) es condición suficiente para minimizar el riesgo global, sin perdida de generalidad se puede afirmar que la regla óptima de clasificación de Bayes es aquella que minimiza el riesgo condicional para cada objeto, es decir:

$$\hat{\mathrm{c}}(\mathbf{x}) = \arg\min_{\omega \in \Omega} R(\omega|\mathbf{x}) \tag{8}$$

Sin embargo, existen funciones de pérdida más apropiadas que la función 0-1 en especial para problemas con un gran número de clases. El objetivo es proponer aquellas funciones que mantengan la simplicidad de la regla de clasificación óptima para el caso de la función 0-1, ec.(8). Sea $\omega_k$ la clase propuesta por el sistema y $\omega_j$ la clase correcta que el sistema debería haber propuesto (utópicamente se espera que $\omega_k = \omega_j$); se puede definir la función de pérdida $\mathrm{l}(\omega_k|\mathbf{x}, \omega_j)$ siguiente:

$$\mathrm{l}(\omega_k|\mathbf{x}, \omega_j) = \begin{cases} 0 & \omega_k = \omega_j \\ \epsilon(\mathbf{x}, \omega_j) & \text{en otro caso} \end{cases} \tag{9}$$

donde $\epsilon(\cdot)$ es una función que depende del objeto($\mathbf{x}$) y la clase correcta ($\omega_j$) pero no depende de la clase incorrecta propuesta por el sistema ($\omega_k$), hecho que mantendrá la simplicidad de la regla de clasificación. Esta función debe cumplir la propiedad de convergencia, es decir $\sum_{\omega_j \in \Omega} \mathrm{p}(\omega_j|\mathbf{x})\, \epsilon(\mathbf{x}, \omega_j) < \infty$. El objetivo de la nueva función $\epsilon(\cdot)$ es evaluar la pérdida en caso de que el sistema falle, puesto que en caso contrario no existe pérdida alguna.

Se puede demostrar (ver apéndice A) que el valor esperado del riesgo condicional para la función de pérdida de la ec. (9) es:

$$R(\omega_k|\mathbf{x}) = \mathrm{S}(\mathbf{x}) - \mathrm{p}(\omega_k|\mathbf{x})\, \epsilon(\mathbf{x}, \omega_k) \tag{10}$$

donde $\mathrm{S}(\mathbf{x}) = \mathrm{E}[\epsilon(\mathbf{x},\omega)] = \sum_{\omega_j \in \Omega} \mathrm{p}(\omega_j|\mathbf{x})\, \epsilon(\mathbf{x},\omega_j)$ y $\mathrm{S}(\mathbf{x}) < \infty$, es decir, la suma ponderada sobre todas las posibles clases converge a un número finito que sólo depende de $\mathbf{x}$. Obviamente, $\epsilon(\cdot)$ debe restringirse a las funciones que cumplan la propiedad de finitud mencionada.

Finalmente, la regla de clasificación óptima para la función de pérdida (9) es muy similar a la regla de clasificación de Bayes para la función de pérdida 0-1 (ver apéndice A),y se define como:

$$\hat{\mathrm{c}}(\mathbf{x}) = \arg\max_{\omega \in \Omega} \{\mathrm{p}(\omega|\mathbf{x})\, \epsilon(\mathbf{x},\omega)\} \tag{11}$$

## 3. Traducción automática estadística

La traducción automática estadística (TAE) es un caso específico de un problema de clasificación donde el conjunto de clases es el conjunto de todas las posibles frases que se podrían componer en la lengua destino, por lo que $\Omega = \mathbf{E}^*$. Del mismo modo, los objetos que van a ser clasificados[7] son frases de la lengua destino, es decir, $\mathbf{f} \in \mathbf{F}^*$.

En un sistema de traducción estadístico, la regla de clasificación de Bayes es la de la Ec. (2). Como se ha comentado antes, esta regla se obtiene usando una función de pérdida 0-1 , es decir:

$$\hat{\mathbf{e}} = \hat{\mathrm{c}}(\mathbf{f}) = \arg\max_{\omega_k \in \Omega} \{p(\mathbf{f}|\omega_k) p(\omega_k)\} \tag{12}$$

donde $\omega_k = \mathbf{e}_k$ Esta función es inapropiada cuando el número de clases es extenso (especialmente si es infinito) como ocurre en traducción estadística. En concreto, si la traducción correcta para la frase de origen $\mathbf{f}$ es $\mathbf{e}_j$, y la hipótesis del sistema es $\mathbf{e}_k$; usar una función de pérdida 0-1 (Ec. (5)) tiene la consecuencia de penalizar el sistema con el mismo criterio independientemente de cuál es la traducción ($\mathbf{e}_k$) propuesta por el sistema, cuál es la traducción correcta ($\mathbf{e}_j$) y cuál es la frase origen ($\mathbf{f}$). Es decir, el sistema no tiene ningún criterio para errar.

Se puede proponer una función que tenga en cuenta la probabilidad de la traducción correcta ($p(\mathbf{e}_j)$). Si definimos la función de pérdida de la Ec. (9) con $\epsilon(\mathbf{f},\mathbf{e}_j) = p(\mathbf{e}_j)$, se obtiene la siguiente función de pérdida:

$$\mathrm{l}(\mathbf{e}_k|\mathbf{x},\mathbf{e}_j) = \begin{cases} 0 & \mathbf{e}_k = \mathbf{e}_j \\ p(\mathbf{e}_j) & \text{en otro caso} \end{cases} \tag{13}$$

Esta función de pérdida parece ser más apropiada que la 0-1, puesto que el sistema "prefiere" errar en la frase origen ($\mathbf{f}$) cuya traducción correcta[8]($\mathbf{e}_j$) tiene menor probabilidad en la lengua destino. Entonces, el riesgo global se reducirá puesto que el sistema se equivocará en las traducciones menos probables.

[7] En este contexto la acción de clasificar un objeto $\mathbf{f}$ en la clase $\omega_k$ (o $\mathbf{e}_k$) es una forma de expresar que $\mathbf{e}_k$ es la traducción de $\mathbf{f}$.

[8] Aquí reside la importancia de distinguir entre la traducción propuesta por el sistema ($\mathbf{e}_k$) y la traducción correcta ($\mathbf{e}_j$) para la frase origen ($\mathbf{f}$).

Se puede demostrar (mediante la Ec. (12)) que esta función de pérdida produce como regla de clasificación la regla de traducción directa, que se corresponde con la Ec. (3). Por todo el análisis anterior, si analizamos teóricamente (ignorando errores de modelado) las reglas de traducción directa e inversa asumiendo que se conocen las verdaderas distribuciones de probabilidad, la regla de traducción directa debería obtener mejores resultados que la regla de traducción inversa por provenir de una función de pérdida más apropiada.

Puede ocurrir que algunas aproximaciones estadísticas usadas para modelizar probabilidades de traducción no sean simétricas (como lo son los modelos de IBM [1]) o incluso estar diseñadas para ser utilizadas sólo en un sentido de traducción. Por lo tanto, *el error de modelado*, que se produce por no utilizar el mismo modelo en su forma directa o inversa, puede ser más importante que la ventaja obtenida por utilizar una función de pérdida más adecuada. Para medir esta asimetría se podría utilizar, la regla directa en su forma inversa de manera que el modelo de traducción será el mismo y la influencia de la asimetría se reduce o incluso se elimina. Volviendo a utilizar el teorema de Bayes a la ecuación (3), obtenemos *la forma inversa de la regla directa (FIRTD)*:

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e}\in\mathbf{E}^*} \left\{ p(\mathbf{e})^2 p(\mathbf{f}|\mathbf{e}) \right\} \tag{14}$$

De cualquier forma, esta aproximación asume que el modelo de lenguaje de la lengua destino es un modelo muy próximo al real, puesto que el "peso" de la asimetría pasa de los modelos de traducción ($p(\mathbf{e}|\mathbf{f})$,$p(\mathbf{f}|\mathbf{e})$) al modelo de lenguaje ($p(\mathbf{e})$). La pregunta de cuál de las dos formas (directa o inversa) es más apropiada para modelizar el proceso de traducción depende del las propiedades del modelo de traducción y del modelo de lenguaje, así como de la mejor estimación de estos modelos que se pueda obtener con la muestra disponible. No obstante, en la práctica, la diferencia entre las ecuaciones (14) y (3) se puede utilizar como una medida de la asimetría del modelo.

Análogamente a la Ec. (13), se puede definir una función de pérdida que contemple la influencia de $\mathbf{f}$. Esta función de pérdida se deriva a partir de la Ec. (9) usando $\epsilon(\mathbf{f}, \mathbf{e}_j) = p(\mathbf{f}, \mathbf{e}_j)$:

$$\mathrm{l}(\mathbf{e}_k|\mathbf{f}, \mathbf{e}_j) = \begin{cases} 0 & \mathbf{e}_k = \mathbf{e}_j \\ p(\mathbf{f}, \mathbf{e}_j) & \text{en otro caso} \end{cases} \tag{15}$$

La correspondiente regla de traducción a la función de pérdida de la Ec. (15) es:

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e}\in\mathbf{E}^*} \{ p(\mathbf{f}, \mathbf{e}) p(\mathbf{e}|\mathbf{f}) \} \tag{16}$$

De esta regla se obtienen varias reglas de traducción óptimas, dependiendo de la aproximación que se utilice para modelizar la tarea de traducción. La regla más importante producida por esta función es la *regla de traducción inversa y directa (RTI&D)*, que se expresa como:

$$\hat{\mathbf{e}} = \arg\max_{\mathbf{e}\in\mathbf{E}^*} \{ p(\mathbf{e}) p(\mathbf{f} \,|\, \mathbf{e}) p(\mathbf{e} \,|\, \mathbf{f}) \} \tag{17}$$

La interpretación de esta regla es un refinamiento de la regla directa de traducción. En este caso, el sistema intenta fallar en los pares $(\mathbf{f}, \mathbf{e})$ menos probables en términos de $p(\mathbf{e}, \mathbf{f})$, siendo $\mathbf{e}$ la traducción correcta de $\mathbf{f}$.

En conclusión, la diferencia teórica más importante entre todas las funciones de pérdida propuestas (y por extensión sus reglas óptimas de traducción asociadas) es en qué frases tenderá a fallar el sistema. Por ejemplo, comparando la función de pérdida 0-1 con la de la Ec. (15), la diferencia es que la 0-1 errará independientemente de la probabilidad de la frases, mientras que la RTI&D fallará en frases con una probabilidad (de modelos de traducción y lenguaje) muy pequeña.

## 4. Resultados experimentales

El objetivo de esta sección es mostrar con resultados preliminares y basándonos en modelos sencillos, cómo se puede utilizar la teoría desarrollada para mejorar las prestaciones de un sistema de traducción. Por lo tanto, los experimentos que se muestran tienen como objetivo demostrar la utilidad práctica de la teoría anterior y no la obtención de un sistema de traducción competitivo.

Antes de comenzar debemos definir lo que entendemos por *un error de búsqueda* y un *error de modelo*. Cuando el traductor produce una traducción errónea, esta se puede deber a dos factores diferentes: la búsqueda subóptima no ha encontrado una buena hipótesis ó el modelo de traducción no es capaz de proponer una buena traducción (y por lo tanto, encontrar la traducción es imposible). En caso de que la traducción propuesta por el sistema tenga una mayor probabilidad que la traducción de referencia, diremos que se trata de *un error de modelo (ErrM)*. Análogamente, en el caso contrario, esto es, la traducción propuesta tiene menor probabilidad que la traducción de referencia, diremos que se trata de un *error de búsqueda (ErrB)*.

Para llevar a cabo la experimentación práctica se seleccionó el modelo de IBM 2 [1]. Esta elección responde a la simplicidad del modelo, a que muchos modelos se inicializan con los alineamientos y diccionarios de este modelo, y al hecho de que la búsqueda directa se puede puede llevar a cabo de forma exacta.

Los parámetros del modelo de IBM 2, se entrenaron con la herramienta estándar *GIZA++* [14]. Para realizar la búsqueda inversa implementamos el algoritmo expuesto en [5]. Aunque este algoritmo no es óptimo, hemos ajustado los parámetros expuestos en [5] para minimizar los errores de búsqueda y por lo tanto minimizar las dependencias del algoritmo de búsqueda. Es importante resaltar, que más tarde este algoritmo puede ser extendido a modelos más complejos de IBM. Para llevar a cabo la búsqueda directa se implementó un algoritmo óptimo basado en programación dinámica. En el caso de la regla de traducción inversa y directa (RTI&D) se implementaron dos versiones subóptimas: una dirigida por el modelo directo (más rápida pero menos precisa), que llamaremos "RTI&D-D"; y otra dirigida por el algoritmo inverso que es más lenta e intenta aproximarse a la óptima, siendo por ello más precisa y que llamaremos "RTI&D-I". La descripción detallada de estos algoritmos queda fuera del objetivo de este artículo.

| | Test | | Entrenamiento | |
|---|---|---|---|---|
| | Español | Inglés | Español | Inglés |
| Núm. Pares de Frases | 1K | | 170K | |
| Longitud media | 12,7 | 12,6 | 12,9 | 13,0 |
| Tamaño del Vocabulario | 518 | 393 | 688 | 514 |
| Núm. Palabras con 1 aparición | 107 | 90 | 12 | 7 |
| Núm. de ocurrencia de palabras | 12,7K | 12,6K | 2193K | 2206K |
| Perplejidad | 3,62 | 2,95 | 3,50 | 2,89 |

**Cuadro 1.** Estadísticas básicas de la tarea del TURISTA.

En cuanto a la tarea de traducción, se seleccionó la tarea del TURISTA [15] que es una tarea bilingüe del Español al Inglés. Los pares de frases se corresponden con las interacciones entre individuos que se producen en la recepción de un hotel. El corpus consiste en 171,352 pares de frases diferentes, donde 1,000 frases se seleccionaron aleatoriamente para para test, y el resto (en conjuntos que crecían exponencialmente: 1K, 2K, 4K, 8K, 16K, 32K, 64K, 128K y 170K pares de frases) para el entrenamiento del modelo. El cuadro 1 muestra las estadísticas básicas del corpus. Para evaluar la calidad de las traducciones, hemos utilizado las siguientes medidas calculables automáticamente: WER, SER[9], y el BLEU [16].

En la figura 1 se puede apreciar las diferencias en términos del WER para todas las formas de la regla de traducción directa: la forma inversa 'FIRTD" (Ec. 14), "RTD" (Ec. 3), y "RTD-N" (la versión con normalización de longitud de la RTD). Destacar la importancia de la asimetría del modelo en los resultados obtenidos. La forma inversa de la regla directa obtiene los mejores resultados de dicha figura. La versión normalizada se creó debido a que el modelo de IBM 2 pese a tener un modelo de longitud, en su forma directa tiende a dar traducciones cortas. De cualquier modo, la "RTD" y la "RTD-N" se comportan peor que la "RTI" (ver cuadro 2).

En el cuadro 2 se comparan las prestaciones de los algoritmos Aunque los resultados obtenidos con el WER no mejoran el ratio, es muy importante tener en cuenta que la función de pérdida optimiza el SER y no el WER. Es decir, lo que se está intentando minimizar es el SER, lo que en principio podría también suponer un descenso del WER, aunque no tiene porqué ser necesariamente de esta forma. Remarcar también las diferencias existentes entre los tiempos de ejecución.

Dando firmeza a esta idea, la figura 2 muestra el SER obtenido para diferentes tamaños de entrenamiento y reglas. En ella se observa que conforme aumenta el tamaño del corpus de entrenamiento existe una diferencia significativa en el

[9] Del inglés "Word Error Rate" que es el mínimo número (en porcentaje) de borrados, inserciones y sustituciones que son necesarias para transformar la traducción propuesta por el sistema en la traducción de referencia. SER (Sentence Error Rate) se corresponde con el número (en porcentaje) de traducciones propuestas que no son iguales a las traducciones de referencia.

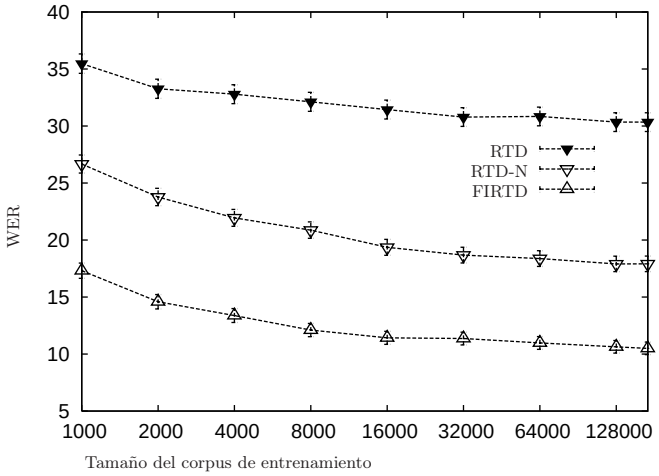



**Figura 1.** Medición de la asimetría del modelo 2 de IBM. Medido mediante el WER (intervalo de confianza al 95 %) cometido para la tarea de el *Turista* con diferentes tamaños de corpus de entrenamiento.

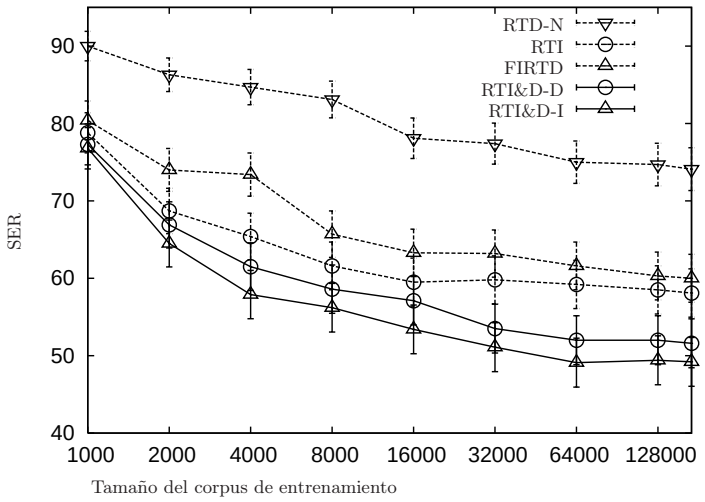



**Figura 2.** Resultados en términos del SER (intervalo de confianza al 95 %) para el conjunto de test de la tarea del Turista con diferentes reglas de traducción y diferentes tamaños de entrenamiento.

| Modelo | WER | SER | BLEU | ErrB | Tiempo |
|---|---|---|---|---|---|
| RTI&D-I | **10.0** | **49.2** | **0.847** | 1.3 | 34 |
| RTI&D-D | 10.6 | 51.6 | 0.844 | 9.7 | 2 |
| FIRTD | 10.5 | 60.0 | 0.837 | 2.7 | 35 |
| RTI | 10.7 | 58.1 | 0.843 | 1.9 | 43 |
| RTD-N | 17.9 | 74.1 | 0.750 | 0.0 | 2 |
| RTD | 30.3 | 92.4 | 0.535 | 0.0 | 2 |

**Cuadro 2.** La calidad de la traducción con diferentes reglas de traducción para el conjunto de evaluación de la tarea del TURISTA con un conjunto de entrenamiento de 170K frases. El tiempo de cómputo esta expresado en segundos.

comportamiento entre RTI y las dos RTI&D. Es decir, el uso de estas reglas reduce el SER y además esta diferencia es estadísticamente significativa conforme aumenta el tamaño del corpus y así la estimación de estos parámetros. En el caso de la forma inversa de la regla directa ("FIRTD"), conforme aumenta el tamaño del corpus de entrenamiento el error tiende a decrementarse y aproximarse a las prestaciones de la RTI. No obstante, las diferencias no son significativas estadísticamente hablando y no se puede concluir nada determinante. En conclusión, hay dos conjuntos de reglas: el primer grupo esta compuesto por la FIRTD y la RTI; y el segundo grupo esta compuesto por las dos versiones de RTI&D. El primer conjunto de reglas obtiene un error más elevado en términos del SER que el segundo grupo. No obstante, la RTD&I dirigida por el algoritmo directo ("RTI&D-D") tiene muy buenas propiedades en la práctica, principalmente la velocidad de cómputo.

## 5. Conclusiones

Para cada función diferente $\epsilon(\mathbf{x}, \omega_j)$ en la función de pérdida general de la Ec. (9), existe una regla de Bayes óptima distinta. La cuestión de usar una u otra es una cuestión principalmente práctica, que depende de los modelos estadísticos utilizados.

En concreto, algunas de estas funciones sobresalen por motivos históricos o motivos prácticos. Por ejemplo, en este trabajo se exploran las reglas directa, inversa y las reglas directa e inversa. Por lo tanto, hemos proporcionado un marco teórico que permite explicar las diferencias y puntos en común de las distintas reglas. Este marco teórico predice una mejora en términos del SER, que ha sido comprobada en la práctica. Sin embargo, debido a motivos prácticos, estas propiedades podrían no llegar a cumplirse en la práctica para ciertos modelos y corpus (como ocurre con el modelo de IBM 2 y las reglas de traducción Directa e Inversa). De entre las reglas analizadas en la práctica la regla RTI&D dirigida por el modelo directo ("TRI&D-D") es un compromiso muy bueno entre las propiedades prácticas y teóricas.

Las consecuencias que en la práctica se podrán obtener de este marco teórico son evidentes, puesto que el objetivo será optimizar la función $\epsilon(\cdot)$ (en la Ec. (9)) en un dominio funcional. Este será precisamente el trabajo futuro que analiza-

remos. Además, también pretendemos analizar estas propiedades con modelos más competitivos.

## Agradecimientos

Este trabajo ha sido respaldado parcialmente por "La Conselleria d'Empresa Universitat i Ciència" de la "Generalitat Valenciana" bajo la beca CTBPRA/2005/004; por la Comsión Interministerial de Ciencia y Tecnología española (CICYT) bajo el proyecto ITEFTE (TIC2003-08681-C02-02); y por el proyecto HERMES (Vicerrectorado de Investigación de la Universidad de Castilla-La Mancha).

## Referencias

1. Brown, P.F., Della Pietra, S.A., Della Pietra, V.J., Mercer, R.L.: The mathematics of statistical machine translation: Parameter estimation. Computational Linguistics **19**(2) (1993) 263–311
2. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis. John Wiley, New York, NY (1973)
3. Knight, K.: Decoding complexity in word-replacement translation models. Computational Linguistics **25**(4) (1999) 607–615
4. Jelinek, F.: A fast sequential decoding algorithm using a stack. IBM Journal of Research and Development **13** (1969) 675–685
5. García-Varea, I., Casacuberta, F.: Search algorithms for statistical machine translation based on dynamic programming and pruning techniques. In: Proc. of Machine Translation Summit VIII, Santiago de Compostela, Spain (2001) 115–120
6. Och, F.J., Tillmann, C., Ney, H.: Improved alignment models for statistical machine translation. In: Proc. of the Joint SIGDAT Conf. on Empirical Methods in Natural Language Processing and Very Large Corpora, University of Maryland, MD (1999) 20–28
7. Tomás, J., Casacuberta, F.: Monotone statistical translation using word groups. In: Procs. of the Machine Translation Summit VIII, Santiago de Compostela, Spain (2001) 357–361
8. Zens, R., Och, F., Ney, H.: Phrase-based statistical machine translation. In: Advances in artificial intelligence. 25. Annual German Conference on AI. Volume 2479 of Lecture Notes in Computer Science. Springer Verlag (2002) 18–32
9. Koehn, P., Och, F.J., Marcu, D.: Statistical phrase-based translation. In: Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference (HLT/NAACL), Edmonton, Canada (2003)
10. Och, F., Ney, H.: The Alignment Template Approach to Statistical Machine Translation . Computational Linguistics **30**(4) (2004) 417–449
11. Och, F.J.: Statistical Machine Translation: From Single-Word Models to Alignment Templates. PhD thesis, Computer Science Department, RWTH Aachen, Germany (2002)
12. Marino, J., Banchs, R., Crego, J., de Gispert, A., Lambert, P., Fonollosa, J., Ruiz, M.: Bilingual n-gram statistical machine translation. In: Proc. of Machine Translation Summit X, Phuket, Tailand (2005) 275–282
13. Berger, A.L., Della Pietra, S.A., Della Pietra, V.J.: A maximum entropy approach to natural language processing. Computational Linguistics **22**(1) (1996) 39–72

14. Och, F.J.: GIZA++: Training of statistical translation models (2000) http://www-i6.informatik.rwth-aachen.de/\~och/software/GIZA++.html.
15. Amengual, J., Benedí, J., Castaño, M., Marzal, A., Prat, F., Vidal, E., Vilar, J., Delogu, C., di Carlo, A., Ney, H., Vogel, S.: Definition of a machine translation task and generation of corpora. Technical report d4, ITI (1996) ESPRIT, EuTrans IT-LTR-OS-20268.
16. Papineni, K.A., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. Technical Report RC22176 (W0109-022), IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY (2001)

## A. La regla de clasificación de mínimo riesgo global

En este apéndice se explica como obtener la regla de clasificación para distintas funciones de pérdida. Por lo tanto, esta sección es una demostración de las reglas que se han utilizado a lo largo del trabajo.

Si el sistema de clasificación define una función de pérdida como la de la ecuación (9), entonces, el riesgo condicional cometido se puede calcular como:

$$\begin{aligned} R(\omega_k \,|\, \mathbf{x}) &= \sum_{\omega_j \in \Omega} l(\omega_k|\omega_j, \mathbf{x})\, \mathrm{p}(\omega_j|\mathbf{x}) \\ &= \sum_{\substack{\omega_j \in \Omega \\ \omega_k \neq \omega_j}} \epsilon(\omega_j, \mathbf{x})\, \mathrm{p}(\omega_j|\mathbf{x}) \\ &= (-1)\left(\epsilon(\omega_k, \mathbf{x})\, \mathrm{p}(\omega_k|\mathbf{x}) - S(\mathbf{x})\right), \end{aligned} \tag{18}$$

donde: $S(\mathbf{x}) = \sum_{\omega_j \in \Omega} \epsilon(\omega_j, \mathbf{x})\, \mathrm{p}(\omega_j|\mathbf{x})$.

Teniendo en cuenta que $\mathrm{S}(\mathbf{x})$ es constante para un valor fijo de $\mathbf{x}$ y no depende de la clase $(\omega_k)$, la regla del mínimo riesgo global se simplifica de la siguiente forma:

$$\begin{aligned} \mathrm{c}^\star(\mathbf{x}) &= \arg\min_{\omega_k \in \Omega} R(\omega_k|\mathbf{x}) \\ &= \arg\min_{\omega_k \in \Omega} \left\{(-1)\left(\epsilon(\omega_k, \mathbf{x})\, \mathrm{p}(\omega_k|\mathbf{x}) - S(\mathbf{x})\right)\right\} \\ &= \arg\max_{\omega_k \in \Omega} \left\{\epsilon(\omega_k, \mathbf{x})\, \mathrm{p}(\omega_k|\mathbf{x}) - S(\mathbf{x})\right\} \\ &= \arg\max_{\omega_k \in \Omega} \left\{\epsilon(\omega_k, \mathbf{x})\, \mathrm{p}(\omega_k|\mathbf{x})\right\} \end{aligned} \tag{19}$$

Esta última ecuación (19) es en la que se basa todo el documento. Las ecuaciones (2), (3), (12), y (16) son casos específicos de esta función de clasificación.

# Interfaces de usuario inteligentes: Pasado, presente y futuro

Víctor López Jaquero, Francisco Montero, José Pascual Molina
y Pascual González

Laboratorio de Interfaces de Usuario e Ingeniería del Software (LoUISE)
Instituto de Investigación en Informática (I3A)
Universidad de Castilla-La Mancha, 02071 Albacete, España
{victor, fmontero, jpmolina, pgonzalez}@info-ab.uclm.es

**Resumen.** Cuando en 1956 en la conferencia de Dartmouth se lanzó el reto de crear máquinas que fueran capaces de simular las distintas facetas de la inteligencia humana, sin duda una de las facetas más destacables en las que se pensaba era la de la comunicación. La simulación de las capacidades de comunicación por parte de una máquina implica la capacidad de realizar actos de comunicación, no sólo con otras máquinas, sino también con seres humanos. En este artículo se analiza la evolución de las interfaces de usuario con un cierto grado de inteligencia, donde la máquina trata de comunicar a través de la interfaz la información que considera de relevancia. De igual manera, se analizan las tendencias actuales dentro del campo de investigación de la Inteligencia Artificial (IA) aplicada a la Interacción Persona-Ordenador (IPO), así como hacia dónde nos encaminan las actuales propuestas en dicho campo.

*Palabras clave:* Interfaces de usuario inteligentes, adaptatividad, interacción persona-ordenador

## 1 Introducción

La IA trata de simular las capacidades humanas mediante el uso de máquinas. Este reto lanzado en la famosa conferencia de Dartmouth en 1956 ha concentrado una enorme cantidad de esfuerzo dentro de la comunidad investigadora en los últimos 50 años. Aunque después de estos años no se han alcanzado las eufóricas predicciones lanzadas tras dicha conferencia, múltiples avances en la simulación de las capacidades humanas en máquinas avalan la evolución del campo científico de la IA.

Un campo imprescindible para la simulación de las capacidades humanas es la comunicación. La comunicación en este escenario puede realizarse entre distintas máquinas o entre máquinas y seres humanos. Es en este segundo caso donde surge la disciplina IPO, la cual estudia el análisis, diseño, implementación y evaluación de la interfaz de usuario (IU), prestando especial atención a la comunicación entre la máquina y el ser humano. Una IU es una faceta muy importante de cualquier sistema, si ésta es *apropiada* hará que el usuario se sienta cómodo con la máquina. Del mismo modo, aunque una máquina realice correctamente las tareas para las que fue diseñada,

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 868-873, 2006.

el sistema no será acogido satisfactoriamente si no es capaz de comunicar sus logros de forma inteligible y usable por el usuario. Existe un verdadero interés dentro de la IPO para conseguir una buena comunicación entre el usuario y la máquina, expresada en la enorme comunidad de investigadores dedicados al estudio de técnicas que mejoren la usabilidad [12] de una interfaz de usuario.

En la actualidad, un ejemplo del interés que suscita el proceso de comunicación interpersonal y el estudio de las posibilidades de extrapolar el mismo *modus operandi* a la comunicación entre personas y máquinas lo constituye la red de excelencia SIMILAR[1].

En la búsqueda de crear interfaces de usuario con alto grado de usabilidad, la comunidad de investigadores IPO trata de crear IUs que produzcan mejores sensaciones en el usuario, lo cual implica mejorar la *naturalidad de la interacción*. Dicha mejora pasa principalmente por un aumento en la calidad de la comunicación entre la máquina y el usuario, lo cual ha llevado a la acogida y consideración de técnicas de IA dentro de la interacción que permiten, entre otras muchas cosas, proveer respuestas adecuadas a las acciones del usuario. El uso de distintos elementos de la IA dentro de la IPO ha dado lugar a que se acuñara el concepto de *interfaz de usuario inteligente* (IUI).

A lo largo de este artículo se analiza la evolución de las IUs con un cierto grado de inteligencia, donde la máquina trata de comunicar a través de la interfaz la información que considera más relevante de forma adecuada. De igual manera, se analizan las tendencias actuales dentro del campo de investigación de la IA aplicada a la IPO, para más tarde inferir hacia dónde nos encaminan las actuales investigaciones en dicho campo.

## 2 Las interfaces de usuario inteligentes

Las IUIs son interfaces hombre-máquina cuyo objetivo es mejorar la eficiencia, efectividad, y naturalidad de la interacción hombre máquina representando, razonando o actuando de acuerdo a una serie de modelos (usuario, dominio, tareas, discurso, contenidos, etc). Ello hace del desarrollo de las IUI un tarea multidisciplinar, donde las aportaciones pueden venir desde muy diversas direcciones.

Las IUIs, para lograr su principal objetivo y posibilitarlo en diferentes contextos, deben ser capaces de modelar al usuario, a sus tareas, al entorno en el que el usuario realice su labor, a la interacción y tener capacidad para analizar las entradas y generar las salidas de la forma más adecuada [17]. El contexto de uso se suele describir en función de el usuario que esté actualmente usando la aplicación, la plataforma que esté usando para interaccionar (tanto desde el punto de vista hardware como software), y el entorno físico donde se realiza la interacción (condiciones de luminosidad, ruido, etc). En [15] también se incluye la tarea actual que el usuario intenta alcanzar en cada momento dentro de la descripción del contexto de uso.

[1] http://www.similar.cc

### 2.1 La labor de una interfaz de usuario inteligente

Son muchos los campos en los que la IA aporta técnicas, métodos y propuestas que resultan muy interesantes en el desarrollo de IUI. En IA hay disponibles métodos que permiten que un sistema aprenda (por ejemplo, las redes neuronales, las redes Bayesianas o los algoritmos genéticos), tenga capacidad para representar el conocimiento (redes semánticas y marcos) o para tomar decisiones (lógica difusa, sistemas expertos, razonamiento basado en casos, sistemas multi-agente, árboles de decisión, etc). Con ellos es posible actuar sobre la IU adaptando (ConCall [24]), facilitando su uso (Deja Vu [9], STARZoom [2], VITE [11]), evaluando la interacción (SmartCanvas [18], TaskTracer [4]), simulando comportamientos (Adele) [13], guiando al usuario (Alife-WebGuide) [8] o ayudando al diseñador de IU (Moby-D [21], U-TEL [3], EDEM [10], FUSE [14], SUPPLE [7]).

En el desarrollo de productos software, y las IU no son una excepción, la tendencia predominante desde hace más de una década vienen siendo los entornos de desarrollo basados en modelos (MB-UIDE) [21]. La idea que subyace en esta tendencia es la de especificar, de forma declarativa, una serie de modelos en los que se describen las características del usuario, las tareas, el dominio, la propia interfaz a distintos niveles de detalle, los dispositivos, etc. Dichos modelos se almacenan en archivos con formato XML, lenguaje que se ha convertido en verdadera *lingua franca* de las IUIs. Las IUs y los sistemas hipermedia se diseñan y desarrollan de esta forma en estos momentos.

La IPO influye en el desarrollo de los IUI incluyendo consideraciones relacionadas con la usabilidad y la calidad de uso [1] que necesariamente debe ofrecer la IU [20]. Las metáforas, reseñadas en el segundo apartado de este artículo, el aumento de la capacidad de procesamiento de los sistemas y las nuevas formas de interactuar hacen necesaria nuevas metodologías que permitan considerar, llegado el momento, objetos de interacción diferentes a los tradicionales [19].

## 3 Mirando al futuro: retos actuales

El desarrollo de IU presenta en la actualidad desafíos comunes a los que encontramos en la IA. En el campo de la interacción persona-ordenador es bien sabido que la interfaz ideal es aquella que no existe, pero también se asume que, hoy en día, resulta ineludible la presencia de un intermediario entre las intenciones del usuario y su ejecución. No importa cuán intuitiva sea la interfaz, estará allí e impondrá una carga cognitiva al usuario.

El ordenador del futuro, según A. van Dam o T. Furness [5], será un mayordomo perfecto, quien conoce mi entorno, mis gustos y mi manera de ser, y de forma discreta se adelanta a mis necesidades sin precisar órdenes explícitas. Cuando el usuario se comunica con este mayordomo, lo hace principalmente mediante el habla, gestos, expresiones faciales y otras formas de comunicación humana, como el dibujo de bosquejos. Este hecho de poder ofrecer artefactos que puedan aprender, crear y comunicarse de igual a igual con una persona es un anhelo que la IA tiene desde sus inicios, hace ahora 50 años.

Para hacer realidad esta visión, algunos agentes necesitarán interpretar los gestos y expresiones a través de técnicas de visión por computador, o procesar la voz con técnicas de reconocimiento y comprensión del lenguaje natural. La inteligencia artificial y los sistemas basados en el conocimiento serán la base del motor de inferencia de los agentes, que determinará sus salidas. Al dirigirse al usuario, el agente deberá hacerlo de manera educada, y posiblemente deberá modular sus acciones de acuerdo al estado y humor del usuario en cada momento.

Los retos de esta tecnología pueden encuadrarse, por lo tanto, en las áreas de entrada, inferencia y respuesta, y más concretamente en la interpretación de los lenguajes de expresión humana, en la representación y gestión del conocimiento del entorno y, finalmente, la comprensión del ser humano como ser social.

## 4 Conclusiones

En la actualidad los productos software tienen la capacidad de ofrecernos información, entretenernos y facilitar nuestro trabajo, pero también pueden entorpecerlo si la IU que nos ofrecen es limitada o difícil de utilizar. En este artículo se ha pretendido mostrar entonces cómo diferentes técnicas de distintas disciplinas, concretamente la IA, la IS y la IPO, se han ido agregando a lo largo de estos años para contribuir en conjunto a crear una experiencia satisfactoria para el usuario, aumentando el grado de inteligencia, sofisticación y usabilidad de las mismas.

Para ilustrar las exigencias del desarrollo actual de interfaces inteligentes, en este artículo se ha presentado también una arquitectura que permite la especificación y desarrollo de IUs capaces de tener en cuenta al usuario, a las tareas que tiene que realizar, el entorno en el que el usuario trabaja y el dispositivo con el que el usuario desempeña su labor. El principal objetivo de la arquitectura presentada es facilitar la comunicación teniéndose en cuenta para ello consideraciones relacionadas con la usabilidad de la interfaz finalmente proporcionada y ofreciéndose la posibilidad de que la IU se adapte en tiempo de ejecución considerándose los factores antes reseñados.

En el futuro puede que la interfaz del ordenador sea tan natural como conversar con otra persona. No debe olvidarse en ese camino que el objetivo que se persigue en la IPO es reducir al mínimo la distancia cognitiva entre el usuario y su tarea, haciendo que la interfaz se diluya hasta hacerse invisible. Pero, sin duda, este camino no lo podrá recorrer la IPO sin el avance de la IA.

## 5 Agradecimientos

Este trabajo está financiado en parte por los proyectos de la Junta de Comunidades de Castilla-La Mancha PBC-03-003 y PCC05-005-1, y el proyecto CICYT TIN2004-08000-C03-01. También deseamos agradecer le apoyo recibido de la red de excelencia SIMILAR (http://www.similar.cc), el grupo de trabajo europeo que persigue creación de una interacción persona-ordenador similar a la comunicación humana

dentro del sexto programa marco.

## 6 Bibliografía


1. Bevan, N. Quality in Use: Meeting user needs for quality. Journal of Systems and Software 49(1): 89-96, 1999.
2. Bruno, P., Ehrenberg, V., Holmquist. STARzoom: an interactive visual database interface. Proceedings of the 4th international conference on Intelligent user interfaces. Los Angeles, California, United States, 188, 1998.
3. Chung-Man, R., Maulsby, D., Puerta, A. U-TEL: a tool for eliciting user task models from domain experts. Proceedings of the 3rd international conference on Intelligent user interfaces. San Francisco, California, United States. 77 – 80. 1997.
4. Dragunov, A., Dietterich, T., Johnsrude, K., McLaughlin, M., Li, L., Herlocker, J. TaskTracer: a desktop environment to support multi-tasking knowledge workers. Proceedings of the 10th international conference on Intelligent user interfaces. San Diego, California, USA. 75 – 82, 2005.
5. Earnshaw, R., Guedj, R., van Dam, A. and Vince, J.. Frontiers in Human-Centred Computing, Online Communities and Virtual Environment. Springer-Verlag, 2001.
6. Fernández-Caballero, A., López Jaquero, V., Montero, F. , González, P. Adaptive Interaction Multi-agent Systems in E-learning/E-teaching on the Web. International Conference on Web Engineering, ICWE 2003. In Web Engineering: International Conference, ICWE 2003, Oviedo, Spain, July 14-18, 2003. Proceedings. J.M. Cueva Lovelle, B.M. González Rodríguez, L. Joyanes Aguilar, J.E. Labra Gayo, M. del Puerto Paule Ruiz (Eds.). Springer Verlag, LNCS 2722, pp. 144-154. ISSN:0302-9743. Oviedo, Spain, June, 2003.
7. Gajos, K., Weld, D. SUPPLE: automatically generating user interfaces. Proceedings of the 9th international conference on Intelligent user interface. Funchal, Madeira, Portugal
8. Gaudiano, P., Kater, K. ALife-WebGuide: an intelligent user interface for Web site navigation. Proceedings of the 5th international conference on Intelligent user interfaces. New Orleans, Louisiana, United States. 121 – 124. 2000
9. Gordon, A., Domeshek, E. Deja Vu: a knowledge-rich interface for retrieval in digital libraries. International Conference on Intelligent User Interfaces. Proceedings of the 3rd international conference on Intelligent user interfaces. San Francisco, California, United States. 127 – 134. 1997
10. Hilbert, D., Robbins, J., Redmails, D. EDEM: intelligent agents for collecting usage data and increasing user involvement in development. Proceedings of the 3rd international conference on Intelligent user interfaces. San Francisco, California, United States. 73 – 76. 1997.
11. Hsieh, H., Shipman, F. VITE: a visual interface supporting the direct manipulation of structured data using two-way mappings. Proceedings of the 5th international conference on Intelligent user interfaces. New Orleans, Louisiana, United States. 141 – 148. 2000
12. ISO 9241-11. Ergonomic requirements for office work with visual display terminals (VDTs) - Part 11: Guidance on usability, 1998.
13. Johnson, W. L., Shaw, E. Marshall, A., LaBore, C. Evolution of user interaction: the case of agent adele. Proceedings of the 8th international conference on Intelligent user interfaces. Miami, Florida, USA. 93 – 100. 2003
14. Lonczewski, F. Providing user support for interactive applications with FUSE. Proceedings of the 2nd international conference on Intelligent user interfaces. Orlando, Florida, United States. 253 – 256. 1997.

15. López Jaquero, V., Montero, F., Molina, J.P., González, P. Fernández Caballero, A. A Seamless Development Process of Adaptive User Interfaces Explicitly Based on Usability Properties. Proc. of 9th IFIP Working Conference on Engineering for Human-Computer Interaction jointly with 11th Int. Workshop on Design, Specification, and Verification of Interactive Systems EHCI-DSVIS'2004 (Hamburg, July 11-13, 2004). LNCS, Vol. 3425, Springer-Verlag, Berlin, Germany, 2005.
16. López Jaquero, V., Montero, F., Molina, J.P., González, P., Fernández-Caballero, A. A Multi-Agent System Architecture for the Adaptation of User Interfaces. 4th International Central and Eastern European Conference on Multi-Agent Systems (CEEMAS 2005). 15-17 September 2005, Budapest, Hungary. In Multi-Agents Systems and Applications IV. M. Pechoucek, P. Petta, L. Z. Varga (Eds.) LNAI 3690, Springer-Verlag, Germany, 2005.
17. Maybury, M. Intelligent user interfaces: an introduction. In Proceedings of the 4th international Conference on intelligent User interfaces (Los Angeles, California, United States, January 05 - 08, 1999). IUI '99. ACM Press, New York, NY, 3-4, 1999.
18. Mo, Z., Lewis, J.P., Neumann, U. SmartCanvas: a gesture-driven intelligent drawing desk system. International Conference on Intelligent User Interfaces Proceedings of the 10th international conference on Intelligent user interfaces. San Diego, California, USA, 2005.
19. Molina, J. P., García, A.S., López Jaquero, V. and González, P. Developing VR applications: the TRES-D methodology. First International Workshop on Methods and Tools for Designing VR applications (MeTo-VR), in conjuntion with 11th International Conference on Virtual Systems and Multimedia (VSMM'05), Ghent, Belgium, 2005.
20. Montero, F., López Jaquero, V., Ramírez, Y., Lozano, M., González. P. Patrones de Interacción: para usuarios y para diseñadores. VI Congreso de Interacción Persona-Ordenador. 13-16 de septiembre, Granada, España. 2005.
21. Puerta, A.R. A Model-Based Interface Development Environment. IEEE Software,1997.
22. Vanderdonckt, J. Advice-giving systems for selecting interaction objects, in Proc. of 1st Int. Workshop on User Interfaces to Data Intensive Systems UIDIS'99 (Edinbourg, 5-6 septembre 1999), IEEE Computer Society Press, Los Alamitos, 1999, pp. 152-157.
23. Vanderdonckt, J., Bodart, F. Encapsulating Knowledge for Intelligent Automatic Interaction Objects Selection. In ACM Proc. of the Conf. On Human Factors in Computing Systems INTERCHI'93 (Amsterdam, 24-29 April 1993), S. Ashlund, K. Mullet, A. Henderson, E. Hollnagel & T. White (Eds.), ACM Press, New York, 1993.
24. Waern, A., Tierney, M., Rudsstrom, A., Laaksolahti, J. ConCall: edited and adaptive information filtering. Proceedings of the 4th international conference on Intelligent user interfaces. Los Angeles, California, United States 185, 1998.
25. Wahlster, W., Maybury, M. An Introduction to Intelligent User Interfaces. In: RUIU, San Francisco: Morgan Kaufmann, 1998, pp. 1- 13.

# Un algoritmo on-line para un scheduler en Internet

Carlos Gomez[1], Mauricio Solar[1], Fernanda Kri[1], Víctor Parada[1], Lorna Figueroa[1] y Mauricio Marin[2]

[1] Department of Computer Engineering, University of Santiago de Chile
Av. Ecuador 3659, Santiago, Chile
{cgomez, fdakri}@diinf.usach.cl
{msolar, vparada, lorfig}@usach.cl
[2] Department of Computer Engineering, University of Magallanes
Casilla 113-D, Punta Arenas, Chile
Mauricio.Marin@umag.cl

**Resumen.** This paper refers to a study of the different assignment strategies or scheduling algorithms for the m-machine problem, determining the most efficient of the chosen algorithms. In addition to the algorithms selected in the literature, a new algorithm called limit is proposed. Every algorithm is analyzed with respect to four metrics that are recorded when a task arrives and is assigned to a processing element. The metrics are presented in the paper and are analyzed according to two perspectives in the dynamic scheduling problem: one global and the other individual. In the simulations made with real cases there are clear differences that place the limit algorithm as the most efficient.

## 1 Introducción

Las aplicaciones reales de scheduling son un problema complejo en el cual se ha realizado una gran cantidad de investigación en las áreas de ciencia de la computación, inteligencia artificial e investigación operativa. Algunos ejemplos recientes son aplicaciones reales en una compañía de imprenta mostrado en [1], una solución al problema de scheduling en trenes sobre una vía presentado en [2], y una comparación de modelos paralelos de algoritmos genéticos y búsqueda tabú para el problema de scheduling de procesos concurrentes sobre una máquina paralela con procesadores homogéneos mostrado en [3].

En este trabajo se estudia y mejora un modelo de computación paralela aplicado en los buscadores de Internet, que son portales o sitios virtuales donde los usuarios buscan información. La respuesta que recibe el usuario del buscador es en forma de enlaces o *links* a documentos que más se acercan a sus necesidades de información.

En el modelo a estudiar, se tiene un conjunto de *n* elementos de procesamiento (EP) que forman el servidor del sistema y que acceden en forma paralela a la información almacenada en una base de datos. Las consultas y/o requerimientos de los usuarios constituyen un conjunto de tareas que llegan arbitrariamente, es decir, no se tiene conocimiento del tiempo de llegada al sistema [4], a un EP central denominado *broker* o scheduler, el cual se encarga de distribuir las tareas en cada EP del servidor.

A. Fernández-Caballero, M.G. Manzano, E. Alonso & S. Miguel (Eds.): CMPI-2006, pp. 874-883, 2006.

Una parte importante de la gestión de los *brokers* está relacionada con la forma de asignar los procesos a cada una de las máquinas que conforman el servidor. El criterio o algoritmo de asignación de tareas está dentro del ámbito de los algoritmos de *scheduling*, donde la decisión de asignación se realiza de manera dinámica. Bajo algunas suposiciones, tales como el Principio de Localidad, este modelo se transforma en el utilizado en el problema de las *m-máquinas* introducido por Graham.

Se realizará un estudio de las distintas estrategias de asignación o algoritmos de *scheduling* para el problema de las *m-máquinas*, determinando el más eficiente de los algoritmos seleccionados. Además de los algoritmos encontrados en la literatura, se propone un nuevo algoritmo denominado *límite*.

Cada algoritmo será analizado con respecto a cuatro métricas de medición que son registradas según llega una tarea y es asignada a alguno de los EP. Estas métricas de medición serán analizadas según dos puntos de perspectivas en el problema de *scheduling* dinámico: la perspectiva global y la perspectiva particular.

En el segundo capítulo de este trabajo se describe el modelo utilizado en el problema de las *m-máquinas*. En el capítulo tres se definen las métricas de medición y la importancia de cada una en el análisis. En el cuarto capítulo se enuncian de una forma breve los algoritmos utilizados en la comparación, incluyendo el propuesto. En el capítulo cinco se mencionan las series de tareas utilizadas en tiempo de ejecución. En el sexto capítulo se realiza un análisis de los resultados obtenidos, para finalizar con las conclusiones del trabajo.

## 2 Descripción del modelo

Graham [5] estudia el problema de *scheduling* en su forma básica, denominado problema de las *m-máquinas*. Este problema pertenece a la clase *NP-duro* [6].

El problema de las *m-máquinas* consiste en que dado *m* máquinas y un conjunto de *n* trabajos (o jobs), se deben asignar los *n* trabajos a las *m* máquinas minimizando el tiempo de finalización del trabajo que termine último.

Para el modelo *broker*-servidor, las tareas llegan de manera arbitraria al sistema, i.e. sus tiempos de llegada son desconocidos al momento de ejecución. Luego, las tareas deben ser clasificadas como *tareas aperiódicas*. Para efectos de simplificación, se asume que cada vez que llega una tarea ésta está lista para ejecutarse, razón por la cual no se utilizará el tiempo de llegada de la tarea en el modelo.

En el modelo existe una *cola de llegada* en la cual se insertan todas las nuevas tareas que llegan al sistema. Esta cola está en el *broker* o *scheduler*. La comunicación entre el *broker* y cada uno de los EP es a través de una *cola de tareas* en cada uno de los EP. La *cola de llegada* y la *cola de tareas* siguen un orden FIFO (*first in first out*). El modelo se observa en la figura 2.1.

Con respecto a las restricciones, en el modelo *broker*-servidor estudiado cada trabajo consiste exactamente de una tarea (trabajo atómico), cada trabajo tiene idéntico tiempo de ejecución en cualquiera de las *m* máquinas, no existe retardo de comunicación entre el *broker* y el servidor, y la tolerancia a fallas no es tomada en cuenta. Además se asume que en la *cola de llegada* del sistema siempre existe al menos una tarea para asignar a los EP, es decir, se asegura un flujo continuo de tareas.

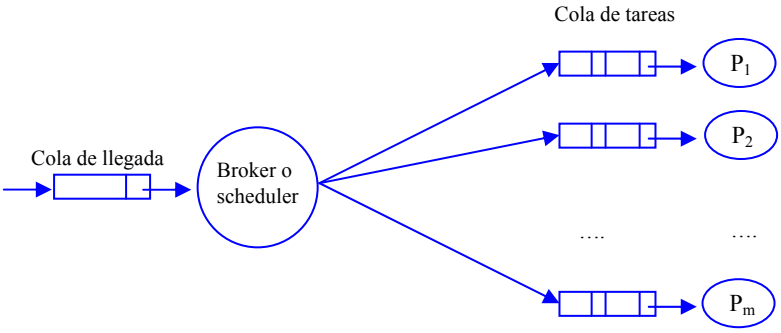



**Figura 2.1**: Modelo broker-servidor.

## 3 Métricas de medición

En este trabajo se utilizan cuatro métricas. La primera (*MS*) muestra el comportamiento del *makespan* actual (tiempo de finalización de la última tarea o máxima carga de trabajo del sistema); la segunda métrica (*MAO*) compara el *makespan* actual en relación al *makespan* óptimo; la tercera (*LB*) muestra como está la distribución de la carga de trabajo; y la cuarta (*WT*) muestra el comportamiento del tiempo de espera de cada tarea en el sistema. Métricas similares se pueden encontrar en [7]:

- **Métrica *MS:*** Esta métrica evalúa el *makespan* del sistema cada vez que es asignada una tarea. El *makespan* del sistema está dado por la máxima carga de trabajo en todas las máquinas. Los valores de esta métrica no tienen cota inferior, pero lo importante es que el algoritmo asigne las tareas de forma tal que esta métrica sea lo más pequeña posible. Un algoritmo que tenga esta medida más baja que los otros algoritmos significa que el conjunto de tareas es terminado antes que los otros algoritmos.
- **Métrica *MAO* (*Makespan* Actual-Óptimo):** Esta métrica está representada por la ecuación, que busca mostrar el comportamiento del *makespan* actual en relación con el *makespan* óptimo.

$$MAO = \frac{makespan\_actual}{makespan\_optimo} \qquad \text{(ec. 3.1)}$$

  El *makespan* óptimo actual se obtiene sumando las cargas de trabajo de todas las máquinas y dividiéndola por el número de máquinas del sistema, según la ecuación 3.2.

$$makespan\_optimo = \frac{\sum_{i}^{m} L_i}{m} \qquad \text{(ec. 3.2)}$$

- **Métrica *LB* (BALANCEO DE CARGA):** Esta métrica muestra el comportamiento del balanceo de carga realizado por el algoritmo. Los mejores valores son aquellos que más se acercan a uno, y entre más alejados estén del valor uno, peor

es la asignación realizada por el algoritmo en términos del balanceo de carga. Esta métrica está dada por la ecuación 3.3.

$$LB = \frac{makespan\_actual}{mínima\_carga} \quad \text{(ec. 3.3)}$$

- **Métrica *WT* (*WAIT TIME*):** Esta métrica no considera el tiempo de espera en la cola de llegada del sistema, sino que considera el tiempo de espera en la cola de tareas del procesador a la cual fue asignada (ec. 3.4):

$$WT = \sum_{i=1}^{n} tiempo\_finalizacion(J_i) \quad \text{(ec. 3.4)}$$

## 4. Algoritmos utilizados en la comparación

Se describen los algoritmos seleccionados de la literatura y el nuevo algoritmo.

**Algoritmo de Albers*:*** El algoritmo *M2* [8] considera la minimización del *makespan,* a través de la asignación eficiente de un conjunto de tareas en *m* máquinas idénticas. El tiempo de llegada de las tareas es desconocido en tiempo de ejecución, pero el tiempo de procesamiento de la tarea es conocido.

**Algoritmo de Avidor:** Este algoritmo [9] considera el balanceo de carga *on-line* en *m* máquinas idénticas dado una secuencia de trabajos. Los trabajos llegan uno a la vez y deben ser asignados a las máquinas en forma dinámica. La meta que proponen los autores es minimizar el retardo promedio de todas las tareas asignadas, lo que es equivalente a minimizar la suma de todos los cuadrados de las cargas de las máquinas, ya que el tiempo de respuesta de una tarea es proporcional a la carga de trabajo de la máquina. Según los autores, minimizar la carga de trabajo máxima a veces no es conveniente. Los autores denominan a su algoritmo *A(t,α)*, ya que se necesitan de los parámetros *t* y $\alpha$ para su funcionamiento.

**Algoritmo de Bartal:** La idea principal del algoritmo en [10] es prevenir que en todo momento la carga de trabajo de máquinas sea aproximadamente igual. Este algoritmo mantiene una lista ordenada de máquinas según carga de trabajo y divide la lista en dos grupos, manteniendo un 44,5% de las máquinas con una carga de trabajo baja. Los autores llaman al algoritmo *m-máquinas*($\delta$,$\xi$) que requiere los parámetros $\delta$ y $\xi$ para funcionar.

**Algoritmo de Galambos:** [11] llama a su algoritmo Lista Refinada, que trata de mantener las cargas de trabajo de las máquinas tan distintas como sea posible. Utiliza dos parámetros denominados $\alpha$ y $\beta$ que son números reales, y que dependen de *m*.

**Algoritmo de Graham:** El algoritmo *List* [5] trabaja principalmente asignando una tarea (no importando su tiempo de procesamiento) a la máquina con menor carga de trabajo. Mantiene, durante la ejecución, una lista de máquinas ordenadas en forma no decreciente por su carga de trabajo.

**Algoritmo de Karger:** [12] presenta el algoritmos $ALG_\alpha$ que utiliza un parámetro $\alpha$ que afecta su comportamiento. Según los autores, la mejor elección de $\alpha$ depende del número de máquinas *m*, además este parámetro está relacionado con el grado de

no balanceo de las máquinas. Para el funcionamiento del algoritmo se tiene una lista de máquinas en orden no decreciente según la carga de trabajo.

## 5. Algoritmo propuesto

El algoritmo propuesto en [13] se basa en una observación al algoritmo de Graham. Existen tres criterios de asignación de un elemento temporal a una serie de espacios de tiempos tomando en cuenta un límite: *donde peor quepa*, *donde mejor quepa* y *donde primero quepa*. Cabe señalar que, en este caso, el elemento temporal es el tiempo de procesamiento de una tarea que llega al sistema y los espacios de tiempo están representados por las cargas de trabajo de las máquinas. Desde este punto de vista, el algoritmo de Graham asigna una tarea a *donde peor quepa*, es decir, se tienen las máquinas ordenadas según su carga de trabajo y se asigna la tarea a la máquina de menor carga (la primera de la lista). El algoritmo propuesto se basa en la estrategia de asignación *donde mejor quepa*, definiendo un término no presente en la literatura, llamado *límite,* que es la máxima carga de trabajo.

El algoritmo *límite* mantiene una lista de máquinas ordenadas por carga de trabajo en orden no decreciente. En todo momento se calcula el *límite* del sistema. El algoritmo busca una máquina cuya diferencia entre el *límite* menos su carga y el tiempo de procesamiento de la tarea que llega sea mínimo. Un caso especial es cuando llega una tarea y no se encuentra una combinación de máquina y tarea que no sobrepase el *límite*, entonces se asigna a la máquina con menor carga de trabajo. El objetivo del algoritmo *límite* es retrasar el crecimiento del *límite* del sistema asignando de manera eficiente las tareas que llegan en forma arbitraria.

Algunas definiciones importantes son:

- *límite* $K^t$: máxima carga de trabajo presente en todas las máquinas.
- $M_i^t$: carga de trabajo de la máquina *i* cuando se han asignado exactamente *t* tareas.
- Función *w*($J_t$): retorna el tiempo de procesamiento de la tarea $J_t$.

***Algoritmo límite***

1. Ordenar las máquinas según carga de trabajo en forma no decreciente.
2. Esperar a que arribe una tarea $J_{t+1}$.
3. Calcular el *límite* $K^t$. Sea $p = 1$.
4. Para $i = 1$ hasta $m$ hacer
   Si ( $K^t$ - $M_i^t$ - $w(J_{t+1}) > 0$ ) entonces $p = i$
   Sino Break
5. Asignar $J_{t+1}$ a la máquina $p$.
6. Volver al paso (1).

## 6. Análisis de resultados

Considerando la implementación de los algoritmos anteriormente descritos y bajo la suposición de que las operaciones elementales, tales como asignación, suma, resta, multiplicación, división y comparación son de orden constante, se describe la complejidad de tiempo de cada algoritmo descrito en la Tabla 1.

**Tabla 1**: Complejidad en tiempo de los algoritmos.

| **Algoritmo** | **Tiempo de Ejecución** |
|---|---|
| *M2* [8] | $TE(m,n) = m + n\left(3m + \frac{m^2}{2} + k_1\right) + k_2$ |
| *A(t,α)* [9] | $TE(m,n) = m + n\left(\frac{3m}{2} + \frac{m^2}{2} + k_1\right) + k_2$ |
| *m-máquinas*(**δ,ξ**) [10] | $TE(m,n) = m + n\left(\frac{5m}{2} + \frac{m^2}{2} + k_1\right) + k_2$ |
| *Refined List Scheduling* (RLS) [11] | $TE(m,n) = m + n\left(\frac{11m}{2} + \frac{m^2}{2} + k_1\right) + k_2$ |
| *Límite* [13] | $TE(m,n) = m + n\left(\frac{5m}{2} + \frac{m^2}{2} + k_1\right) + k_2$ |
| *List* [5] | $TE(m,n) = m + n\left(\frac{3m}{2} + \frac{m^2}{2} + k_1\right) + k_2$ |
| $alg_\alpha$ [12] | $TE(m,n) = m + n\left(2m + m^2 + k_1\right) + k_2$ |

La complejidad de tiempo de los algoritmos dependen de dos variables: ***m***, el número de máquinas y, ***n***, el número de tareas. Por lo tanto la complejidad de cada algoritmo es O($n$) cuando $m$ es constante, u O($m^2$) cuando $n$ es constante.

Como entrada a los distintos algoritmos evaluados, se seleccionó una serie de conjuntos de tareas; estos conjuntos (Tabla 2) se basan en casos reales de archivos *logs* de supercomputadores: *Cornell Theory Center SP2* (CTC), *NASA Ames iPSC/860* (NASA) y *San Diego Supercomputer Center Paragon* (SDSC) del año 1995 y 1996.

El estudio de los algoritmos fue realizado en forma exhaustiva y evaluados bajo diversas circunstancias para determinar el mejor de ellos.

**Tabla 2**: Características de las series de tareas según casos reales.

| Característica | CTC | NASA | SDSC95 | SDSC96 |
|---|---|---|---|---|
| Número de tareas | 79.285 | 42.029 | 76.527 | 38.702 |
| tiempo máx. ejecución | 71.998 | 62.643 | 525.980 | 448.070 |
| tiempo mín. ejecución | 1 | 1 | 1 | 1 |
| Mediana tiempo de ejecución | 946 | 20 | 25 | 175 |
| Media de tiempo de ejecución | 10.986 | 348 | 3.168 | 7.426 |
| Desviación estándar | 18.130 | 1.816,4 | 10.987 | 15.071 |
| Varianza | 3,28E+8 | 3,29E+6 | 1,20E+8 | 2.27E+8 |

Las pruebas realizadas consideran sistemas compuestos por 4, 16, 32, 64, 128, 256 y 512 máquinas. Por cada caso real se realizaron siete pruebas (una por cada simulación de máquinas). La métrica *MS* es la métrica fundamental en el análisis, ya que una menor medida significa que las tareas serán finalizadas en menos tiempo.

Los resultados dan las condiciones necesarias para que se observen diferencias verdaderas entre los algoritmos. En las pruebas realizadas se observan diferencias de desempeño entre los algoritmos, siendo en general, el algoritmo *límite* el con mejor desempeño. La métrica *LB* muestra el comportamiento del balanceo de carga a través de la división de la carga máxima y la mínima.

La métrica *WT* es la única métrica que es utilizada para evaluar los algoritmos desde una perspectiva particular. Esta medida toma en cuenta principalmente el tiempo de espera de las tareas, pero sólo considerando lo que tendrá que esperar cada tarea para finalizarse una vez asignada. En las pruebas realizadas se observa que el algoritmo de Albers tiene mejor desempeño que los restantes, seguido por los algoritmos de Galambos, Gómez, Graham y Karger.

En el anexo A se muestran las gráficas obtenidas a partir de resultados de las simulaciones para el supercomputador CTC con 256 máquinas para las métricas *MS*, *MAO*, *LB*, y *WT*. En el anexo B se muestran las gráficas obtenidas a partir de resultados de las simulaciones en los supercomputadores CTC, NASA, SDSC95 y SDSC96 con 512 máquinas para la métrica *MS*.

## 7. Conclusiones

Fueron analizados distintos algoritmos buscando el más eficiente para un *broker* en Internet. Según lo planteado en la sección tres, los algoritmos más eficientes para el modelo *broker*-servidor son los mejores desde una perspectiva global, es decir, bajo las métricas *MS*, *MAO* y *LB*. Con respecto a estas métricas, los algoritmos más eficientes son los de Galambos, Gómez, Graham y Karger. En las simulaciones realiza-

das se observan diferencias que dan como algoritmo más eficiente a *límite* seguido por el algoritmo *List* de Graham. Desde una perspectiva particular, el algoritmo con mejor desempeño es el de Albers seguido por el de Graham. Con respecto al conjunto de métricas, éstas evalúan de una forma precisa a cada uno de los algoritmos seleccionados.

Como futuros trabajos de investigación se dan tres fuentes: la eliminación del posible sesgo de las métricas, la creación de métricas para la perspectiva particular y la generalización del algoritmo *límite* a otras instancias del problema de *scheduling*.

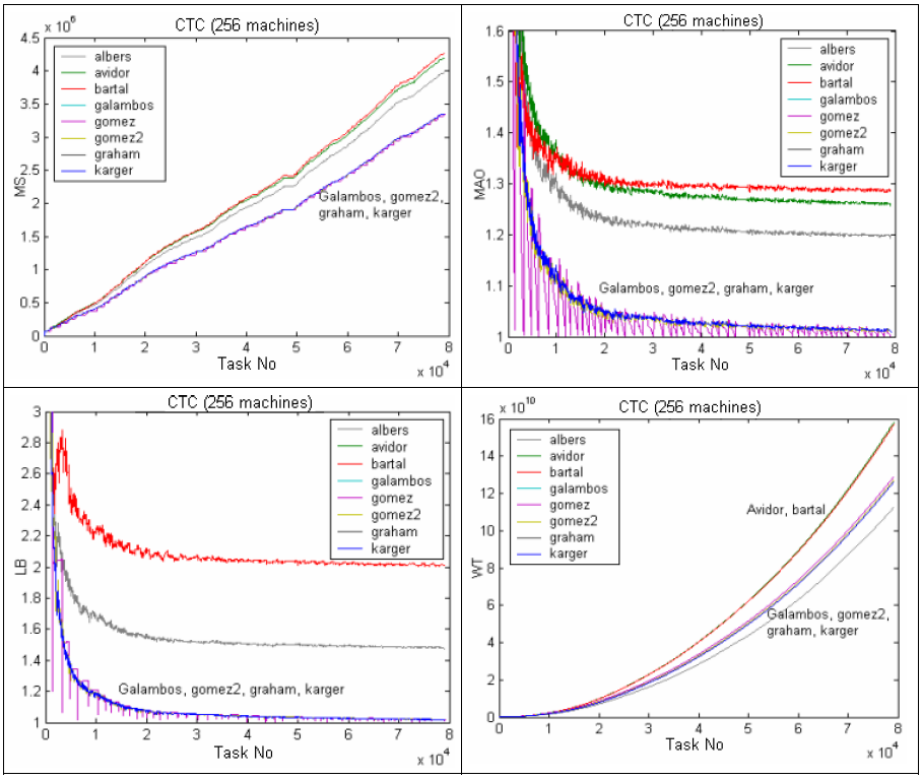
## Agradecimientos

Este trabajo fue parcialmente financiado por el proyecto FONDECYT 1030775.

## Referencias

1. Geiger, M.J. and Petrovic, S. (2004): An Interactive Multicriteria Optimisation Approach to Scheduling. In M. Bramer and V. Devedzic (Eds.). Artificial Intelligence Applications and Innovations. Kluwer Academic Publishers, pp. 475-484.
2. Ingolotti, L, Tormos, P., Lova, A., Barber, F., Salido, M.A. and Abril, M. (2004): A Decision Support System (DSS) for the Railway Scheduling Problem. In M. Bramer and V. Devedzic (Eds.). Artificial Intelligence Applications and Innovations. Kluwer Academic Publishers, pp. 465-474.
3. Pinacho, P.; Solar, M.; Inostroza, M. and Muñoz, R. (2004): Using Genetic Algorithms and Tabu Search Parallel Models to Solve the Scheduling Problem. In M. Bramer and V. Devedzic (Eds.). Artificial Intelligence Applications and Innovations. Kluwer Academic Publishers, pp. 343-358.
4. Andersson, B.; Abdelzaher, T. and Johnson, J. (2003): Partitioned aperiodic scheduling on multiprocessors. In Proc. 17th Int. Parallel & Distributed Processing Symposium. April.
5. Graham, R.L. (1966): Bounds for certain multiprocessing anomalies. Bell System Technical Journal. 45, 1563-1581.
6. Garey, M.R. and Johnson, D.S. (1979): Computer and Intractability, A guide of the theory of NP-completeness. Freeman, San Francisco, USA.
7. Feitelson, D. G. (2003): Metric and Workload Effects on Computer Systems Evaluation. IEEE Computer. Sept, pp. 18-25.
8. Alberts, S. (1999): Better bounds for online scheduling. SIAM Journal of Computing. 29 (2), 459-473.
9. Avidor, A; Assar, Y. and Sgall, J. (2001): Ancient and new algorithms for load balancing in the Lp norm. Algorithmica. 29 (3), 422-441.
10. Bartal, Y.; Fiat, A.; Karloff, H. and Vohra, R. (1995): New algorithms for an ancient scheduling problem. Journal of Computer and System Sciences. 51 (3), 359-366.
11. Galambos, G.. and Woeginger, G. (1993): An on-line heuristic with better worst case ratio than Graham's List scheduling. SIAM Journal of Computing. 22 (2), 332-355.
12. Karger, D.R.; Phillips, S.J. and Torng, E. (1996): A better algorithm for an ancient scheduling problem. Journal of Algorithms. 20 (2), 400-430.
13. Gomez, C. (2004): Broker management in parallel and distributed databases. Master Thesis in Computer Engineering. University of Santiago de Chile, Chile.

**Anexo A**. Gráficas obtenidas a partir de resultados de las simulaciones de los casos reales para el supercomputador CTC compuesto por 256 máquinas, para las métrica *MS*, *MAO*, *LB*, y *WT*.



**Anexo B**. Gráficas obtenidas a partir de resultados de las simulaciones de los casos reales para los supercomputadores CTC, NASA, SDSC95 y SDSC96 con 512 máquinas, para la métrica MS.