TPAL. Una herramienta de simulación de sistemas concurrentes con restricciones temporales *

J. José Pardo, Valentín Valero y M. Carmen Ruiz

Departamento de Informática
Escuela Politécnica de Albacete
Univ. de Castilla-La Mancha
Campus Univ. s/n. 02071. Albacete, Spain.
ph: +34 967 599200 fax +34 967 599224
{jpardo, valentin,carmenr}@info-ab.uclm.es

Resumen En este artículo presentaremos TPAL, una herramienta que permite la especificación y la evaluación de sistemas concurrentes incluyendo factores temporales y probabilísticos. Las especificaciones aceptadas por esta herramienta son escritas en un lenguaje similar a LOTOS, que incluye nuevos operadores para la especificación de comportamientos temporales y probabilisticos.

1 Introducción

Existe una amplia gama de sistemas para los cuales el tiempo adquiere una gran importancia (sistemas de tiempo real). En estos sistemas descubrimos que algunas acciones o tareas son críticas, en el sentido de que se deben ejecutar urgentemente cuando estén disponibles. Además en los últimos años, se han definido varias extensiones de los modelos clásicos usados para describir sistemas concurrentes, con el fin de incorporar este factor. Existen extensiones temporales de CSP [?,?,?], CCS [?], y LOTOS [?]. Bolognesi y Lucidi [?] han definido una extensión temporal de LOTOS en la que se incluye un operador para la especificación de interacciones urgentes. Hansson y Jonsson han definido una extensión de CCS [?,?] con probabilidades y tiempos que incluye un operador de *time-out*. En realidad, no existen todavía demasiadas propuestas que combinen ambos factores en un único modelo. Podemos citar [?], pero la sintaxis del modelo que se presenta es muy limitada, y el modelo se restringe a procesos finitos. En [?] se presenta una semántica de testing para una extensión de LOTOS con tiempo y probabilidades.

El modelo que presentamos se basa en LOTOS, añadiéndole operadores nuevos para especificar el comportamiento temporal y las interacciones urgentes. Siguiendo las ideas de Quemada, Azcorra y de Frutos [?], introducimos un operador prefijo temporal, $a < t_1, t_2 >; P$, el cual establece un intervalo de tiempo

* Este trabajo ha sido parcialmente financiado con el Subproyecto CICYT Evaluación de rendimientos de Sistemas Distribuidos, TIC2000-0701-C02-02.

para la ejecución de la acción correspondiente, y una vez agotado ese intervalo temporal, esa acción ya no puede ser ejecutada. Otros operadores que hemos incluido son el prefijo de acción interna, $\tau; P$ y el operador de espera, $wait(t); P$. El primero establece el carácter de urgencia de la acción τ, en el sentido de que debe ser ejecutada inmediatamente, sin ningún retraso (a menos que tengamos un conflicto con otra acción que pueda ser ejecutada inmediatamente), mientras que el segundo operador establece un retraso de t unidades de tiempo, y entonces se comporta como $\tau; P$. Además, mantenemos los operadores clásicos de LOTOS para la elección externa, composición paralela, ocultamiento y recursión.

En determinadas ocasiones, y para verificar algunas propiedades de un modelo, es más conveniente tener una representación gráfica, normalmente utilizando grafos o autómatas. Esto ha llevado a que la relación entre las álgebras de procesos y los grafos temporales se haya estudiado en varios trabajos anteriores. Dos herramientas clásicas para describir sistemas concurrentes son Modecharts [**?**,**?**] y Autómatas Temporizados [**?**,**?**]. También, en [**?**] se definen máquinas de comunicación en tiempo real (CRSM) para modelar sistemas de tiempo real. En este modelo, un sistema se modela mediante un conjunto de estas máquinas interconectadas una a una por una canal de comunicación. [**?**] proporciona un método para transformar estas máquinas en grafos cuyos nodos muestran la situación actual de cada CRSM, los valores de las variables y el consumo de tiempo de cada CRSM. Otros modelos temporales basados en grafos temporizados que se han propuesto son los Modelos de Transiciones Temporizadas [**?**], en los que el tiempo se modela mediante un reloj conceptual externo que es global, y un conjunto de variables que capturan la información temporal local de cada proceso. Así, cada transición va dotada de una condición de activación, una función de transformación y unos límites temporales para su ejecución.

A la par, con estas extensiones de modelos de especificación han aparecido diversas herramientas que facilitan el manejo y verificación de especificaciones de sistemas concurrentes. Entre ellas podemos destacar UPPAAL [**?**,**?**], creada por las universidades de Upsala y Alborg, y que permite la especificación de sistemas de forma gráfica y textual, CADP [**?**], y LOLA (LOTOS Laboratory) [**?**], creada por la Universidad Politécnica de Madrid, que utiliza LOTOS como lenguaje de especificación.

La estructura de este artículo es la siguiente: En la sección 2 presentamos la sintaxis del álgebra de procesos aceptada por la herramienta. En la sección 3 presentaremos de forma intuitiva la definición de los grafos. A continuación en la sección 4 se describe el algoritmo para el cálculo de cotas de tiempo. Para finalizar con la descripción de la interfaz de usuario de la herramienta.

2 Sintaxis de las especificaciones aceptadas por TPAL

Las especificaciones aceptadas por TPAL están formadas por un conjunto de descripciones de procesos que atienden a la sintaxis siguiente:

proceso ::= PROCESS < id_proceso > [<lista_acciones>];
BEHAVIOUR [PPROBABILISTIC | NON_DETERMINISTIC]
<cuerpo>
ENDPROC

Entre todos ellos existe uno que será el proceso principal de la especificación, para el cual la sintaxis de su descripción variará algo siendo la siguiente:

SPECIFICATION < id_proceso > [< lista_acciones >];
BEHAVIOUR [PROBABILISTIC | NON_DETERMINISTIC]
< cuerpo >
[WHERE < lista_procesos >]
ENDSPEC

donde para la especificación de todos los procesos *id_proceso* es el nombre asignado al proceso en cuestión, *lista_acciones* es una lista con los nombres de todas las acciones utilizadas dentro de la especificación del proceso y *cuerpo* es la especificación en sí del comportamiento del proceso.

En la especificación del proceso principal *lista_procesos* será la descripción de todos los procesos que intervienen en la descripción del sistema completo:

< lista_procesos >::= { T}

Podemos observar que la indicación del tipo de proceso (PROBABILISTIC o NON_DETERMINISTIC) es opcional, ya que la propia herramienta puede identificar el mismo en la mayoría de los casos.

El cuerpo de cada uno de los procesos será escrito de acuerdo a la siguiente expresión BNF, que se encuentra basada en el álgebra de procesos descrita formalmente en [**?**]

T::= N | P
N::= N1 | N2
N1::= stop | a< t_1, t_2 >;T | i;T | wait(t);T
N2::= N or N | N [A] N | N\a | X | rec X.N
P::= $[q_1]$N3 + ... + $[q_n]$N3 | P [A] P | P\a | X | rec X.P
N3::= N1 | rec X.N1

donde $t, t_1 \in \mathbb{R}$, $t_2 \in \mathbb{R} \cup \{*\}$, $a \in$ *lista_acciones*, $q_i \in (0,1)$ y $A \subseteq$ *lista_acciones*.

A continuación se hace una descripción informal de los operadores:

stop .- Representa un deadlock.

$a < t_1, t_2 >; P$.- La acción $'a'$ puede ser ejecutada después de un retraso dentro del intervalo $[t_1, t_2]$. Una vez que el intervalo ha expirado, se pierde la posibilidad de realizar esa acción, es decir, no se fuerza su ejecución cuando su time-out va a expirar. Una vez que la acción a se ha ejecutado, el proceso se comporta como P.

$\tau; P$.- La acción τ se ejecuta inmediatamente y el proceso pasa a comportarse como P.

$wait(t);P$.- El proceso espera $'t'$ unidades de tiempo, y después se comporta como $\tau;P$.

$N_1\,or\,N_2$.- Representa el operador de elección externa clásico. Cuando se requiere que se ejecute una acción, ésta es ejecutada por el proceso que pueda realizarla, y si ambos pueden ejecutarla, entonces la elección del proceso es realizada de forma interna por el sistema.

$N_1[A]N_2$.- Representa la ejecución paralela de los procesos N_1 y N_2 sincronizando en las acciones del conjunto A.

$T\backslash a$.- Todas las instancias de la acción a en el proceso T son ocultadas y se convierten en urgentes (τ).

X .- Representa un proceso del lenguaje.

$rec\ X.P$.- Es el clásico operador de recursión.

$\sum_i[q_i]N_i$. Representa la elección interna probabilística entre los procesos. El proceso N_i será seleccionado con una probabilidad q_i.

Los términos obtenidos mediante esta sintaxis los denotaremos como TPA. Sin embargo, para definir los grafos de estados dinámicos nos restringimos a términos regulares, en los que $\mu X.P$ debe ser guardada (todas las apariciones de X en P deben estar prefijadas, y no puede haber ni un operador paralelo ni de ocultamiento en P). Estos términos los denotaremos como RTPA.

Ejemplo 1 Consideremos el problema de la exclusión mutua en el acceso a un recurso compartido. De acuerdo con la sintaxis comentada anteriormente la especificación de este problema será:

```
System = (Proc1 [] Proc2) [w, s] Recurso) \ w )\ s
  Proc1 = rec X.a1<12.5,20.7>; w<0, ∞ > ;u1<2.3,4.1>;s<0, 0>;X
  Proc2 = rec X. a2<2.75,9.7> w<0, ∞ >;u2<0.3,1.23>;s<0, 0>; X
Recurso = rec X.w<0, ∞> ; s<0, ∞>;X
```

□

3 Grafo de Estados Dinámicos

Una vez comprobado que la especificación introducida se ajusta a la sintaxis del lenguaje definida en la sección anterior, la herramienta genera el grafo de estados asociado a la especificación. Para poder entender el funcionamiento de esta parte de la herramienta, en esta sección describiremos brevemente la estructura y el funcionamiento de los grafos.

Estos grafos de estados dinámicos son esencialmente autómatas extendidos con relojes, los cuales capturan información temporal de cada uno de los procesos. Podemos encontrar en la literatura trabajos relacionados con éste [?], donde los grafos temporizados son definidos para el lenguaje ATP (Algebra of Timed Processes).

Los nodos son etiquetados por términos del álgebra, mientras que los arcos están etiquetados por tuplas (a, r, S) donde a es una acción, r es un conjunto de intervalos de tiempo que especifican restricciones sobre algunos relojes, y S es un conjunto de relojes que deben sincronizar con el reloj global una vez que la transición es ejecutada.

La definición formal de los grafos es la siguiente:

Definición 1 Un grafo de estados dinámicos es una tupla $G = (N, E, \lambda, n_0, \mathit{Clocks})$, donde:

- N es el conjunto de nodos, los cuales están etiquetados con un término (que representamos por $\mathit{term}(n) \in \mathit{TPA}$, $n \in N$)
- E es el conjunto de arcos del grafo $E \subseteq N \times N$, $n_0 \in N$,
- $\mathit{Clocks} = \{R_1, \ldots R_r\}$. Es el conjunto de relojes asociados al grafo. Entre ellos uno es el global que en adelante lo representaremos como R_g),
- $\lambda : E \longrightarrow \mathit{Act} \times \mathit{Times} \times \mathcal{P}_F(\mathit{Clocks})$, donde $\mathit{Times} : \mathit{Clocks} \hookrightarrow \mathbb{R}_0^+ \times \mathbb{R}_0^+ \cup \{\infty\}$ es una función parcial que asocia un intervalo de tiempo a cada uno de los relojes sobre los que está definida. Cada una de las componentes de $\lambda(e)$ la representaremos como $\lambda_i(e)$, $i = 1, 2, 3$.

□

3.1 Evolución del grafo

Dado un grafo de estados dinámicos $G{=}(N, E, \lambda, n_0, \mathit{Clocks})$, podemos definir el estado en el que se encuentra como la tupla (n, C, t) donde $n \in N$, $C : \mathit{Clocks} \longrightarrow \mathbb{R}_0^+$ (función que da los valores de los relojes) y $t \in \mathbb{R}_0^+$ es el instante actual, $t \geq C(R_g)$.

Dados, un estado $s = (n, C, t)$ y un arco e que sale del nodo n y que está etiquetado como $\lambda(e) = (a, r, S)$, definimos:

La cantidad de tiempo mínima que debe transcurrir para que la acción a puede ser ejecutada de acuerdo con este arco, como:

$\mathit{Min}(n, C, e) = \max\{(C(R_j) + \alpha_j) \dot{-} C(R_g) \ \ |R_j \in \mathit{Clocks}, \, r(R_j)$ definido, $r(R_j) = [\alpha_j, \beta_j]\}$

Y la cantidad de tiempo máxima que puede transcurrir para que esa acción pueda ser ejecutada, como:

$\mathit{Max}(n, C, e) = \min\{(C(R_j) + \beta_j) - C(R_g) \ \ |R_j \in \mathit{Clocks}, r(R_j)$ definido, $r(R_j) = [\alpha_j, \beta_j]\}$

Si obtenemos un valor negativo significará que el tiempo para ejecutar la acción ha expirado y no podrá ser ejecutada.

Además definimos el máximo retraso del que se dispone para realizar una acción visible, debido a las restricciones impuestas por las acciones urgentes, como:

$$\mathit{Min}_\tau(n, C) = \begin{cases} \infty & \text{si } \nexists e \; n \xrightarrow{e}, \, \lambda_1(e) = \tau \\ \min\{\mathit{Min}(n, C, e) \,|\, n \xrightarrow{e}, \; \lambda_1(e) = \tau\} & \text{ecc} \end{cases}$$

De acuerdo con estas definiciones, establecemos la condición de ejecución de la acción a en el estado $s = (n, C, t)$ en la existencia de un arco e etiquetado con $\lambda(e) = (a, r, S)$ que sale del nodo n y para el cual se cumple que:

$$\mathit{Min}(n, C, e) \leq \Delta t \leq \mathit{Max}(n, C, e) \wedge \Delta t \leq \mathit{Min}_\tau(n, C)$$

donde $\Delta t = t - C(R_g)$.

Representaremos este predicado como $act(s,e)$.

Por tanto, podemos establecer las siguientes reglas, que capturan las dos posibles evoluciones del grafo:

$$\textbf{G1)} \quad \frac{s \quad , \quad act(s,e)}{s \xrightarrow{e} (m, adjust(C,e,t), t)}$$

$$\textbf{G2)} \quad \frac{s \quad , \quad t' \geq t \quad , \quad t' - C(R_g) \leq Min_\tau(n,C)}{s \underset{t'-t}{\longmapsto} (n, C, t')}$$

con $s = (n,C,t)$, $e = (n,m) \in E$ y

$$adjust(C,e,t)(R_i) = \begin{cases} t & \text{si } R_i \in \lambda_3(e) \\ C(R_i) & ecc \end{cases}$$

3.2 Definición del grafo

Una vez definida la estructura de los grafos y su evolución habrá que definir la construcción del mismo para cada uno de los operadores de álgebra de procesos. En este artículo únicamente mostramos las ideas clave de esta construcción, de forma intuitiva. Esta construccion ya formalizada puede encontrarse en [**?**].

- *STOP*
  El grafo del proceso STOP tendrá un único nodo y un único reloj que será el reloj global.
- *Prefijo Temporizado*
  Si consideramos un proceso $P = a < t_1, t_2 >; P_1$. El grafo de P se obtendrá a partir del grafo de P_1 añadiendo un nuevo nodo, que será el inicial, y una nueva transición desde ese nodo al inicial del grafo de P_1 etiquetado de acuerdo a la transición a. El conjunto de relojes no se verá modificado.
- τ y Wait(t).
  La definición para estos dos operadores es similar a la del prefijo, diferenciándose únicamente en la acción que etiqueta el nuevo arco y en la restricción temporal.
- *Elección Externa*
  Para definir el grafo asociado con este operador impondremos la condición de que ambos componentes tengan un estado inicial sin arcos entrantes, para lo cual necesitamos *desplegar* la recursión más externa que puede aparecer en los componentes de la elección externa.
  Una vez transformados los procesos para que cumplan estas condiciones la construcción del nuevo grafo se hará fusionando los grafos de ambos procesos en uno nuevo en el que los nodos iniciales de ambos grafos se habrán sustituido por un único nodo que será el inicial, modificando el nodo de origen de los arcos que partiesen de esos nodos.
- *Composición Paralela*
  El grafo para este operador será obtenido mediante el producto cartesiano de los grafos que se encuentran involucrados en el paralelo, atendiendo a las acciones de sincronización.

– *Ocultamiento*
  Se modificará la etiqueta de los arcos que están etiquetados con la acción a ocultar.
– *Proceso X*
  El grafo para el proceso X es similar al obtenido para el proceso Stop.
– *Recursión*
  Para obtener el grafo de la recursión sólamente habrá que cambiar el nodo destino de aquellos arcos que llevaban a un nodo X, y hacer que lleguen al nodo inicio de proceso al que representa X.

Ejemplo 2 Siguiendo la definición establecida, el grafo obtenido por la herramienta para el proceso de la exclusión mutua será el mostrado en la figura 1. □

3.3 Problema de explosión de estados

Esta definición de los grafos plantea un problema como es el de explosión de estados, el cual es producido principalmente por la construcción del operador paralelo. Este es un gran problema en cuya solución nos encontramos trabajando actualmente buscando algunos algoritmos de reducción de grafos. El primer algoritmo aplicado es el de eliminación de aquellos nodos que no son alcanzables desde el nodo inicial del grafo, así como todos los arcos que nacen de estos nodos.

Aparte de este algoritmo, se están estudiando otros que podemos clasificar en dos grandes grupos.

– Algoritmos encaminados a detectar no alcazabilidad de determinados estados por motivos temporales.
– Algoritmos encaminados a obtener grafos equivalentes al inicial, con un número menor de nodos o arcos.

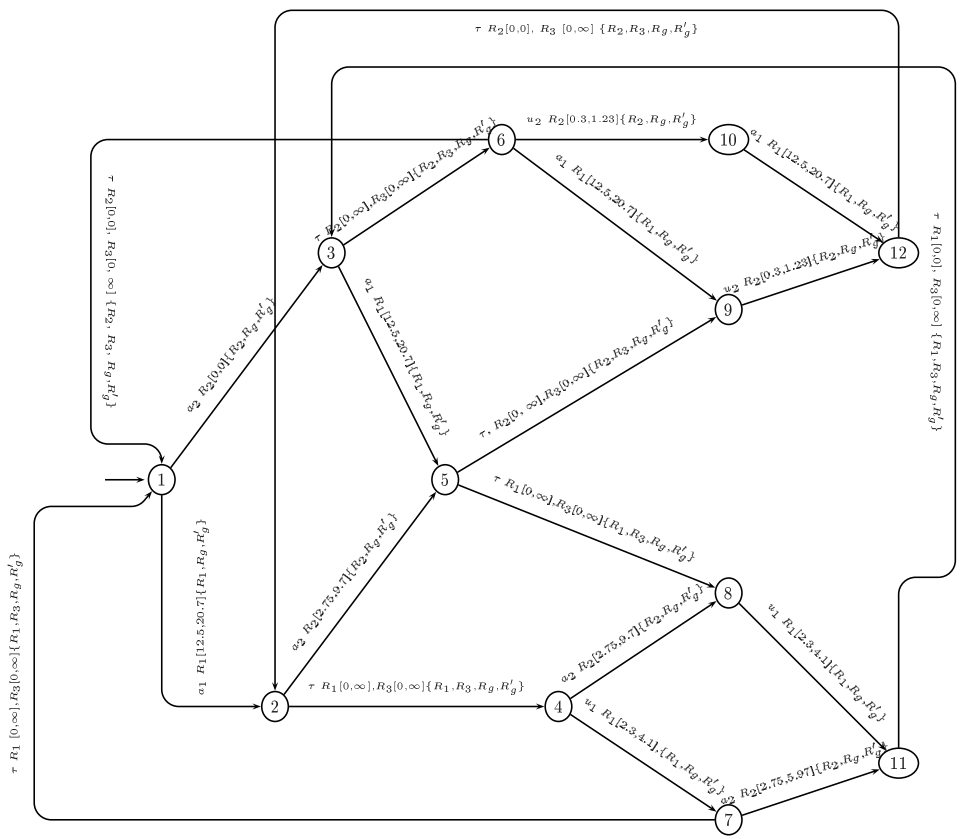
4 Evaluación de propiedades de los sistemas especificados

Una vez creado el grafo de estados dinámicos para un determinado proceso, éste puede ser utilizado para la evaluación de determinadas propiedades, tales como la ausencia de bloqueos, estudio de intervalos de tiempo para alcanzar un estado determinado, etc. A continuación describiremos el algoritmo utilizado para el cálculo de estas cotas.

4.1 Cálculo de cotas temporales

Consideramos una secuencia de transiciones s dentro del grafo $G[\![P]\!]$, de modo que al alcanzar el nodo final de esa secuencia tendremos para cada reloj R_j un límite inferior $Min(s, R_j)$ y uno superior$Max(s, R_j)$, que calculamos siguiendo el algoritmo siguiente .

En cada momento, consideraremos el subconjunto $\rho \subseteq Clocks$, como el conjunto de relojes que están sincronizados con el reloj global R_g, así también consideramos $Bounds(s, R_j) = [b_j, B_j]$ las cotas del reloj R_j tras haber seguido



Clocks=$\{R_1, R_2, R_3, R_g, R_g'\}$

Figura1. Grafo para el problema de la exclusión mutua.

la ruta s, y $Dif(s,R_j) = [d_j, D_j]$ la discrepancia entre el reloj R_j y el reloj global R_g al seguir la ruta s. Podemos destacar que debido a que $R_j \leq R_g$ se cumplirá que $d_j \leq 0$ y $D_j \leq 0$.

Inicialmente tendremos que $Bounds(s_0, R_j) = [0,0]$ y $Dif(s_0, R_j) = [0,0]$ para todos los relojes del grafo donde s_0 es la secuencia vacía.

En un determinado momento y tras haberse ejecutado la secuencia s tendremos que $Bounds(s,R_j) = [b_j, B_j]$ y $Dif(s,R_j) = [d_j, D_j]$. A partir de estos valores para una secuencia $s' = s \cdot e$ donde $\lambda(e) = (a, r, S)$, podemos calcular $Bounds(s', R_j)$ y $Dif(s', R_j)$ de la forma siguiente.

En primer lugar calculamos una cota superior para $Min_\tau(n,C)$, que representa el tiempo que podemos permanecer en el nodo n antes de que se deba ejecutar una transición etiquetada con τ, de la forma siguiente:

$$\Delta_\tau = \begin{cases} \infty\ , & \text{if } \nexists e'\ ,\ n \xrightarrow{e'} , \lambda_1(e') = \tau \\ \min\limits_{\substack{e',n \xrightarrow{e'} \\ \lambda_1(e')=\tau}} & \{ \max\limits_k \{\Psi_\tau(n,e',R_k)/r'(R_k) \text{ está definida, y } r' = \lambda_2(e')\}\} \end{cases}$$

donde

$$\Psi_\tau(n,e',R_k) = \max\{0\ ,\ \min\{\alpha_k + D_k\ ,\ B_m + \alpha_k - b_g\}\ \}$$

se calcula para cada uno de los relojes R_k para los de las transiciones que parten del nodo n y que están etiquetadas con τ para las cuales $\lambda(e') = (\tau, r', S')$ y $r'(R_k) = [\alpha_k, \beta_k]$;

De la misma forma podemos calcular una cota inferior γ para $\min(n,C,e)$ y una cota superior δ para $\max(n,C,e)$. Así tendremos que

$$\gamma = \max_K \{\Psi(n,e,R_k)/r'(R_k) \text{ está definida}\}$$

$$\delta = \min_K \{\Phi(n,e,R_k)/r'(R_k) \text{ está definida}\}$$

donde para cada reloj que tiene definida $r'(R_k)$ tendremos que

- $\Psi(n,e,R_k) = \max\{0\ ,\max\{\alpha_k + d_k\ ,\ b_k + \alpha_k - B_g\}\}$
- $\phi(n,e,R_k) = \min\{\beta_k + D_k\ ,\ B_k + \beta_k - b_g\}$

Finalmente podemos calcular $Bounds(s', R_j)$ y $Dif(s', R_j)$ de la forma siguiente:

- $b'_j = \begin{cases} b_j & \text{si } R_j \notin S \\ b_g + \gamma & \text{si } R_j \in S \end{cases}$
- $B'_j = \begin{cases} B_j & \text{si } R_j \notin S \\ B_g + \min\{\delta, \Delta_\tau\} & \text{si } R_j \in S \end{cases}$
- $d'_j = \begin{cases} 0 & \text{si } R_j \in S \\ \max\{b_j - B_g, d_j\} - \min\{\delta, \Delta_\tau\} & \text{si } R_j \notin S \end{cases}$
- $D'_j = \begin{cases} 0 & \text{si } R_j \in S \\ \min\{B_j - b_g - \delta, D_j - \alpha\} & \text{si } R_j \notin S \end{cases}$

Ejemplo 3 En el ejemplos de la exclusión mutua, podemos seguir el camino 1.2.3.4.1.5.9.11.1, que corresponde al uso del recurso por parte de los dos procesos involucrados, y obtener que las cotas de tiempo para el reloj R_g son $b_j = 6$ y $B_j = 32$.

Puede observarse que los valores obtenidos mediante esta algoritmo no son óptimos, pero la ventaja de dicho algoritmo es que resulta muy eficiente, frente a técnicas alternativas que tendrían que basarse en la resolución de inecuaciones. $\square$

5 Interfaz de la Herramienta

Pasemos a continuación a realizar una breve descripción del funcionamiento de esta herramienta.

Para iniciar la aplicación ejecutaremos el comando **tpal**, tras lo cual aparecerá la ventana principal de la misma, la cual podemos ver en la figura 2.

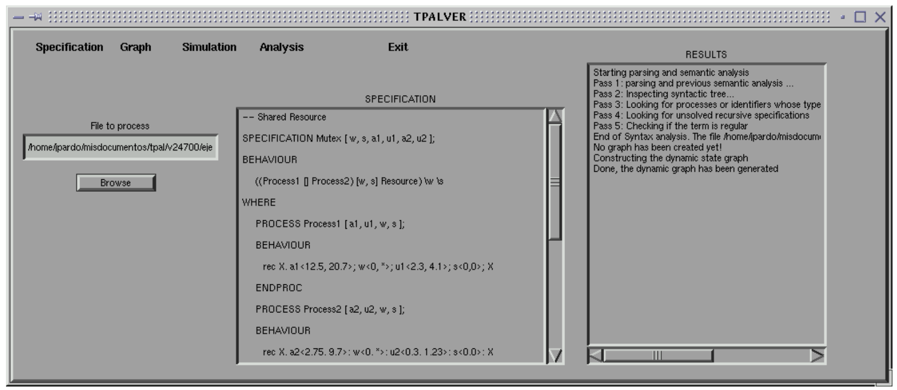


Figura2. Ventana principal de la Herramienta

5.1 Especificaciones

Apertura En primer lugar lo que debemos hacer es indicar el fichero que contenga la especificación que deseamos procesar, y para ellos deisponemos de tres alternativas.

1. Podemos escribir directamente el nombre del mismo en el recuadro **File to process**.
2. Si pulsamos sobre el botón browse que aparece en la ventana nos aparecerá un selector de ficheros mediante el cual podremos desplazarnos por el árbol de directorios hasta seleccionar el fichero deseado.
3. Podemos seleccionar la opción Open File del menú Specification con lo cual se mostrará también el selector de ficheros.

Tras seleccionar el fichero deseado, se mostrará su contenido en el recuadro "Specification".

Edición Para la edición de la especificación sobre la que se está trabajando debemos seleccionar la opción Edit del menú Specification. Esto lanzará un nuevo proceso con el editor joe, que nos permitirá modificar la especificación. Tras salvar los cambios y salir del editor se actualizará el recuadro de la especificación. (ver figura 2.)

Análisis sintáctico Es la primera fase del proceso de una especificación, y comprobará que la especificación se adecúa a la sintaxis. Para ello debemos seleccionar la opción Syntax Analysis del menú Specification.

5.2 Grafos

Una vez realizado el análisis sintáctico de una especificación y si ésta es correcta, podemos generar el grafo correspondiente atendiendo a la definición formal de los mismos. Para ello seleccionaremos la opción Generation del menú Graph.

Una vez generado el grafo, podemos visualizarlo, ya sea en forma de texto o de forma gráfica. Para ello seleccionaremos la opción correspondiente del menú graph. Esta visualización, en caso de grafos muy grandes será parcial, mostrándose en color amarillo aquellos nodos cuyos nodos adyacentes no pueden ser mostrados. En estos casos nosotros podemos pulsar sobre uno de estos nodos, de modo que pase a ser el nodo actual y entonces se redibujará de nuevo el grafo. El nodo actual será mostrado en color rojo.

5.3 Simulación

También tenemos la posibilidad de realizar una simulación, bien de forma manual o de forma automática. En este último caso podemos habilitar la opción de "Avoid timelocks", lo que intentará evitar un deadlock por motivos de la expiración de tiempos.

Durante una simulación tendremos transiciones que se muestran en color rojo, que indican que están habilitadas para ser ejecutadas y en negro aquellas que no lo están. Para ejecutar una de las transiciones que estén habilitadas deberemos pinchar sobre ellas con el botón izquierdo del ratón. Si pulsamos sobre cualquier transición o nodo con el botón derecho, se abrirá una nueva ventana mostrando información sobre el objeto seleccionado.

Durante la simulación, además de la ejecución de transiciones podemos indicar el paso del tiempo, para ello pulsamos sobre el botón Delay de la ventana de simulación, tras lo cual se nos solicitará el tiempo que dejamos pasar. Este paso del tiempo podrá habilitar algunas transiciones así como deshabilitar otras que estuviesen ya habilitadas.

Configuración Antes de iniciar una simulación podemos definir varios parámetros del funcionamiento de la misma. Para ello seleccionamos la opción configuration del menú Simulation, tras lo cual se mostrará la ventana de esta configuración, la cual podemos verla en la figura 3:

En ella podemos indicar si la simulación se hará de forma manual o automática, si se realizará completa o sólamente N pasos.

Reset Seleccionando la opción Re-Inicialization del menú Simulation volveremos el estado inicial de la simulación para volver a comenzar ésta.

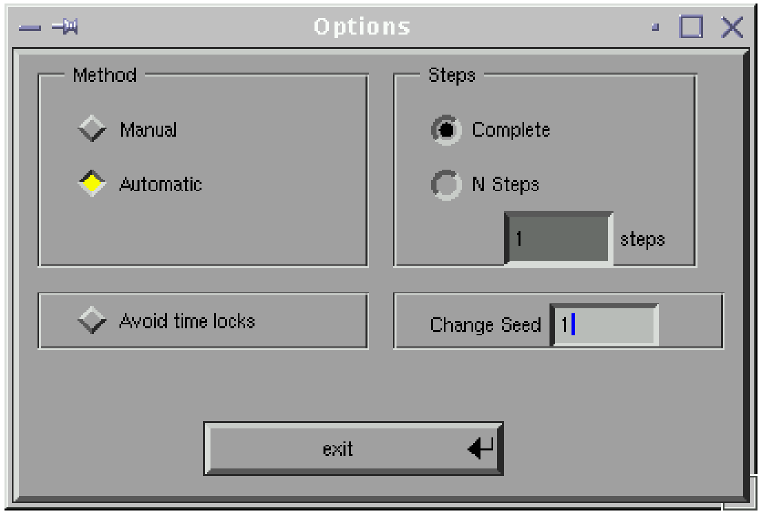


Figura3. Ventana de configuración de la Simulación

5.4 Análisis

Una vez creado el grafo, podemos realizar sobre el mismo el análisis de algunas propiedades, como es el cálculo de una cota superior e inferior para el tiempo de llegada a un nodo dado. Para ello debemos indicar el camino seguido hasta llegar a ese nodo.

6 Conclusiones

En este artículo se ha presentado TPAL, una herramienta para la simulación y la verificación de sistemas concurrentes, así como la definicón formal de algunos conceptos utilizados por esta herramienta.

Los trabajos que se plantean para incrementar la utilidad de esta herramienta se centran principalmente en dos direcciones: Definición de diversos algoritmos de reducción de grafos, y en el estudio de algoritmos para decidir diversas propiedades sobre las sistemas descritos.