# An Example of Performance Evaluation by using the Stochastic Process Algebra: ROSA

Fernando L. Pelayo  Fernando Cuartero  Valentín Valero  Diego Cazorla
Escuela Politécnica Superior de Albacete *
University of Castilla-La Mancha
Campus Universitario s/n. 02071 Albacete, Spain
{fpelayo, fernando, valentin, dcazorla}@info-ab.uclm.es

## Abstract

*We present an algebraic language for the description of probabilistic and non-deterministic processes, which allows us to evaluate performance indexes as well as to check some temporal requirements:* ***ROSA*** *(**R**easoning **O**n **S**tochastic **A**lgebras). As application, we analyse the Alternating Bit Protocol obtaining the average time to send a message, considering that channels may fail with a known probability.*

**Keywords:** Formal Methods, Non-Deterministic and Probabilistic Aspects, Stochastic Process Algebra, Performance Analysis, Real-Time Requirements and Designs Specifications, Fault Tolerance, Case Studies.

## 1. Introduction

In the eighties, fundamentals in process algebras were firmly established. The proliferation of distributed systems, real-time systems and fault-tolerant systems has produced that some factors like time and probabilities, which were abstracted in earlier process algebra models, now, became as very relevant elements in these descriptions.

Due to this fact, we find that currently it is not enough to consider qualitative aspects of concurrent systems. Thus, several researchers have studied possible extensions of these models by including some quantitative factors, like time restrictions, priorities and probabilities.

A first approach to incorporate probabilities in process algebra can be found in [2], where the probabilistic experiment concept is introduced. PCCS is subsequently defined in [7], an algebraic language based in the synchronic version of CCS, in which the non-deterministic choice operator is replaced by the probabilistic choice operator, $\sum[p_i]E_i$, where $p_i$ is the probability with which the process becomes like $E_i$. In [15] the model is expanded, and now the reactive, the generative and the stratified interpretations are presented for probabilistic concurrent processes.

Later, in [8] *Timed Probabilistic Calculus of Communicating Systems* (TPCCS) is defined, which extends CCS both with probabilities and time. The language has a probabilistic choice operator and a non-deterministic choice operator, in an alternative way, i.e., at any state we may find either a probabilistic or a non-deterministic choice. Due to this fact the model is called "alternating model".

We may find also some extensions of CSP [4], [5], where a reactive interpretation of the probabilities is considered, whereas in [13] a generative interpretation is studied. Both models present a testing semantics which is fully abstract with respect to the denotational semantics based on acceptance trees.

However, the underlying idea in all these probabilistic models is to replace a non-deterministic behaviour by a probabilistic one, in which every possible behaviour of the system is quantified by means of a probability. But obviously, this requires a quantitative knowledge of the possible behaviours of the modelled system. Unfortunately, it is not always possible to have such a knowledge, although it can be sometimes estimated [11].

Then, we have that concurrent systems have both non-deterministic and probabilistic behaviours, that we should capture in our descriptions. For instance, a communication channel may fail with a known probability, so this should be captured in the model, but we can not predict the order in which actions of parallel processes are going to be executed, so non-determinism is still required.

Following these ideas, in [16] a probabilistic extension of CCS with a probabilistic choice operator is presented: $\oplus_p$, which has a generative interpretation, while the original choice operator of CCS: $\oplus$, has been maintained to represent the classical non-determinism. In [12] a probabilistic extension of CSP is presented (PCSP), which also maintains

* This work has been partially supported by the CICYT project "Formal development of distributed systems", TIC97-0669-c03-02.

a non-deterministic operator. For this language, an operational semantics is presented and some denotational semantics are given according to three kinds of nodes: probabilistic nodes, non-deterministic nodes and action nodes. In [3], TPAL is presented, an algebraic language with probability and time, it is based on LOTOS, where the internal choice is replaced by a probabilistic choice. This language imposes the alternation of both probabilistic and non-deterministic terms.

It is also desirable to take into account performance aspects in the design of concurrent systems. It is true that often efficiency considerations are forgotten until the system works properly, i.e., once it has been fully tested. However, it becomes obvious that there are some systems (real-time systems) where we need to know a priori if the system will fulfil with its restrictions. Then a model for describing concurrent systems should include also, some capabilities to determine efficiency considerations.

They are presented in [10], [6], [9] and [1] some stochastic process algebras which consider these capabilities. In the last one is described the *Extended Markovian Process Algebra EMPA*, which has some similar characteristics to the one that we present, like the conception of two kinds of actions according to their duration (modelled by Exponentials), and the *race policy*. But in **ROSA**, the time parameter of any action does not restrict its capabilities for synchronizing. Essentially, for the sake of clarity on designed systems (following some ideas presented in [14]), we follow the classic policy of defining an appropriate syntax for each operation. Moreover, the model we present has different semantics interpretations in the concept of some operators (like parallel one).

The paper is structured as follows: in Section 2, the syntax of **ROSA** is defined, the operational semantics for this language is defined in Section 3 and in Section 4 the performance evaluation algorithm is shown. We define an equivalent relation over **ROSA** processes in Section 5 and, finally, in Section 6 we present the case study: the Alternating Bit Protocol.

## 2. Syntax of the Language

Let $\Delta = \{a, b, c, ..\}$ be a set of action types. Let $p, q, r$ be probabilities and let greek letters be time parameters of actions.

Terms of **ROSA** are defined by:

$$P ::= \mathbf{0} | a.P | \langle a, \lambda \rangle.P | P \oplus P | P + P | P \oplus_r P | P ||_A P | recX : P$$

where $r \in (0, 1)$, $\lambda \in \mathbb{R}^+ - \{0\}$, $A \subset \Delta$ and $P$ is a process of **ROSA**.

The operators are interpreted as in classical process algebras, i.e., $action.P$ stands for the prefix operator, $\oplus$ and $+$ are respectively the internal and the external choice, $||_A$ is the parallel operator synchronizing in actions which types belong to $A$, and $recX : P$ is the recursion operator.

The new operators that we have included are:

- $\langle a, \lambda \rangle.P$, which stands for a timed prefix operator, where action labelled with $a$ has a duration following an Exponential random distribution, with parameter $\lambda$. Let us observe that now $a.P$ is a particular case of this operator taking $\lambda = \infty$, which means that $\mathbf{E}[Exp[\infty]] = 0$, i.e. this action has duration 0.

- $P \oplus_r Q$ denotes the probabilistic choice with the classic generative interpretation, thus with probability $r$ the process behaves like $P$ and with probability $1 - r$ it behaves like $Q$.

## 3. Operational Semantics

The operational semantics is defined in the Plotkin and Milner´s style (see in Tab.1), by using a labelled transition system with two kinds of transitions:

- Probabilistic transitions, which have the following form:

  $$P \xrightarrow{r,a,\lambda} Q$$

  where $r$ is a probability, $a$ is an action type and $\lambda$ is a parameter of an Exponential distribution that models the duration of the action. This rule says that the process $P$ can evolve by executing the action labelled by $a$, with probability $r$ and with duration described by an Exponential distribution with parameter $\lambda$.

- Non-deterministic transitions, which are as follows:

  $$P \longrightarrow Q$$

  By means of these rules we capture the internal decisions that the system makes, solving the non-deterministic choices.

Rules S1 to S3 and rule S6 are quite obvious, and they do not require any more comments.

Rules S4 capture the semantics of the external choice; concretely S4a and S4b represent the evolution by executing an action that only is at the disposal of one component. When an action can be executed by both arguments, rules S4c to S4h establish the evolution of $P + Q$, by distinguishing the possible cases according to the duration of the involved actions. If the action is immediate for both components (it has zero duration), then the choice is solved in a non-deterministic way, which is captured by rules S4c and S4d, where firstly we must evolve to an intermediate state in order to solve the non-deterministic choice, and then we

$$\textbf{(S1a)}\quad \frac{}{a.P \xrightarrow{1,a,\infty} P} \qquad\qquad \textbf{(S1b)}\quad \frac{}{\langle a,\lambda\rangle.P \xrightarrow{1,a,\lambda} P}$$

$$\textbf{(S2a)}\quad \frac{P \xrightarrow{q,a,\lambda} P'}{P \oplus_p Q \xrightarrow{p\cdot q,a,\lambda} P'} \qquad\qquad \textbf{(S2b)}\quad \frac{Q \xrightarrow{t,a,\lambda} Q'}{P \oplus_p Q \xrightarrow{(1-p)\cdot t,a,\lambda} Q'}$$

$$\textbf{(S3a)}\quad \frac{}{P \oplus Q \longrightarrow P} \qquad\qquad \textbf{(S3b)}\quad \frac{}{P \oplus Q \longrightarrow Q}$$

$$\textbf{(S4a)}\quad \frac{P \xrightarrow{q,a,\lambda} P' \wedge a \notin \mathbf{Type}[\mathbf{D}[Q]]}{P + Q \xrightarrow{q,a,\lambda} P'} \qquad\qquad \textbf{(S4b)}\quad \frac{Q \xrightarrow{q,a,\lambda} Q' \wedge a \notin \mathbf{Type}[\mathbf{D}[P]]}{P + Q \xrightarrow{q,a,\lambda} Q'}$$

$$\textbf{(S4c)}\quad \frac{P \xrightarrow{q,a,\infty} P' \wedge Q \xrightarrow{r,a,\infty} Q'}{P + Q \longrightarrow P \xrightarrow{q,a,\infty} P'} \qquad\qquad \textbf{(S4d)}\quad \frac{P \xrightarrow{q,a,\infty} P' \wedge Q \xrightarrow{r,a,\infty} Q'}{P + Q \longrightarrow Q \xrightarrow{r,a,\infty} Q'}$$

$$\textbf{(S4e)}\quad \frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q' \wedge \lambda_2 \neq \infty}{P + Q \longrightarrow P \xrightarrow{q\cdot r,a,\lambda_1+\lambda_2} P'} \qquad\qquad \textbf{(S4f)}\quad \frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q' \wedge \lambda_1 \neq \infty}{P + Q \longrightarrow Q \xrightarrow{q\cdot r,a,\lambda_1+\lambda_2} Q'}$$

$$\textbf{(S4g)}\quad \frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q' \wedge \exists\lambda_i \neq \infty}{P + Q \xrightarrow{q\cdot(1-r),a,\lambda_1} P'} \qquad\qquad \textbf{(S4h)}\quad \frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q' \wedge \exists\lambda_i \neq \infty}{P + Q \xrightarrow{(1-q)\cdot r,a,\lambda_2} Q'}$$

$$\textbf{(S5a)}\quad \frac{P \xrightarrow{q,a,\lambda} P' \wedge a \notin \mathbf{Type}[\mathbf{D}[Q]] \cup A}{P||_A Q \xrightarrow{q,a,\lambda} P'||_A Q} \qquad\qquad \textbf{(S5b)}\quad \frac{Q \xrightarrow{q,a,\lambda} Q' \wedge a \notin \mathbf{Type}[\mathbf{D}[P]] \cup A}{P||_A Q \xrightarrow{q,a,\lambda} P||_A Q'}$$

$$\textbf{(S5c)}\quad \frac{P \xrightarrow{q,a,\infty} P' \wedge Q \xrightarrow{r,a,\infty} Q' \wedge a \notin A}{P||_A Q \longrightarrow P''||_A Q \xrightarrow{q,a,\infty} P'||_A Q} \qquad\qquad \textbf{(S5d)}\quad \frac{P \xrightarrow{q,a,\infty} P' \wedge Q \xrightarrow{r,a,\infty} Q' \wedge a \notin A}{P||_A Q \longrightarrow P||_A Q'' \xrightarrow{r,a,\infty} P||_A Q'}$$

$$\textbf{(S5e)}\quad \frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q' \wedge a \notin A \wedge \lambda_2 \neq \infty}{P||_A Q \longrightarrow P''||_A Q \xrightarrow{q\cdot r,a,\lambda_1+\lambda_2} P'||_A Q} \qquad\qquad \textbf{(S5f)}\quad \frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q' \wedge a \notin A \wedge \lambda_1 \neq \infty}{P||_A Q \longrightarrow P||_A Q'' \xrightarrow{q\cdot r,a,\lambda_1+\lambda_2} P||_A Q'}$$

$$\textbf{(S5g)}\quad \frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q' \wedge a \notin A \wedge \exists\lambda_i{\neq}\infty}{P||_A Q \xrightarrow{q\cdot(1-r),a,\lambda_1} P'||_A Q} \qquad\qquad \textbf{(S5h)}\quad \frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q' \wedge a \notin A \wedge \exists\lambda_i{\neq}\infty}{P||_A Q \xrightarrow{(1-q)\cdot r,a,\lambda_2} P||_A Q'}$$

$$\textbf{(S5i)}\quad \frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q' \wedge a \in A}{P||_A Q \xrightarrow{q\cdot r,a,\mathbf{min}[\{\lambda_1,\lambda_2\}]} P'||_A Q'} \qquad\qquad \textbf{(S6a)}\quad \frac{P[recX : P/X] \xrightarrow{q,a,\lambda} P'}{recX : P \xrightarrow{q,a,\lambda} P'}$$

**Table 1. Operational Semantics for ROSA**

evolve by executing the involved action. Whereas if the action is not immediate for at least one of the components, we have that the evolution will be made either in a probabilistic way, or, firstly in a non-deterministic way and in a probabilistic way later, according to the following cases distinction (in both, immediate actions have some kind of urgent character):

- If both components cooperate for executing the action (rules S4e and S4f), then a non-deterministic evolution takes place and afterwards a probabilistic one, which takes as new parameter the sum of the parameters of that action in each component, which means that the action will have a shorter duration, and the probability for this evolution is the product of the probabilities in each component (*race policy*).
- Otherwise, for rules S4g and S4h we capture the evolution when only one of the components execute the action. In this case we preserve the duration of the action corresponding to the component that execute it, but the probability for this case is the product of $q$ and $1 - r$, where $r$ is the probability to execute the action by the component which does not cooperate in its execution.

Rules S5 describe the behaviour for the parallel composition operator, they are very similar to those described for the external choice operation. We just need to mention rule S5i, which captures the evolution of $P||_A Q$ by executing an action belonging to the synchronization set $A$. In this case we see that the new duration is obtained from an Exponential distribution with parameter $\mathbf{min}[\{\lambda_1, \lambda_2\}]$, which means that the slowest one imposes its duration.

We have used two auxiliary functions for describing the operational semantics, $\mathbf{D}$ and $\mathbf{Type}$, and they are defined as follows:

**Definition 1** $\mathbf{D}$ *is a function defined on the set of **ROSA** processes which associates to every process its multiset of available actions, it is defined by means of structural induction over the set of **ROSA** processes:*

$$
\begin{array}{lllll}
\mathbf{D}: & \{\textbf{\textit{ROSA}}\ processes\} & \rightarrow & \mathcal{M}[\Delta \times \mathbb{R}^{+} - \{0\}] \\
 & \mathbf{0} & \rightsquigarrow & \varnothing \\
 & a.P & \rightsquigarrow & \{a\} \\
 & \langle a, \lambda\rangle.P & \rightsquigarrow & \{\langle a, \lambda\rangle\} \\
 & P \oplus Q & \rightsquigarrow & \mathbf{D}[P] \cup \mathbf{D}[Q] \\
 & P + Q & \rightsquigarrow & \mathbf{D}[P] \cup \mathbf{D}[Q] \\
 & P \oplus_r Q & \rightsquigarrow & \mathbf{D}[P] \cup \mathbf{D}[Q] \\
 & P||_A Q & \rightsquigarrow & \mathbf{D}[P] \cup \mathbf{D}[Q] \\
 & recX : P & \rightsquigarrow & \mathbf{D}[P]
\end{array}
$$

**Definition 2** $\mathbf{Type}$ *is a function defined on the set of multisets of actions which associates to every multiset, the set consisting of their action types:*

$$\mathbf{Type} : \mathcal{M}[\Delta \times \mathbb{R}^{+} - \{0\}] \longrightarrow \mathcal{P}[\Delta]$$

## 4. Performance Evaluation

Starting from the operational semantics, we may construct for each process a *transition graph*, which can be used for performance evaluation studies, according to the following steps:

1. To begin with, non-labelled transitions are eliminated from this *transition graph*, which correspond to non-deterministic choices, and we split the graph into several new graphs, according to Fig.1. Thus, we obtain a set of *deterministic transition graphs* which have only transitions labelled with actions.

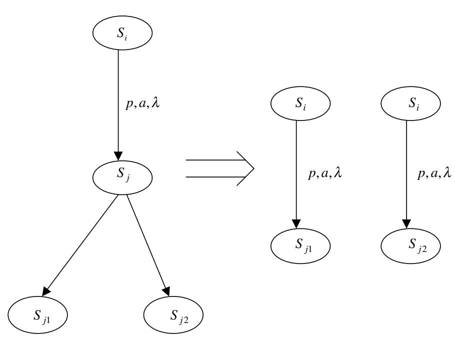

**Figure 1. Evaluation Algorithm - Phase 1**

2. Now, from each one of these *deterministic transition graphs*, we relabel every transition: $\xrightarrow{q,a,\lambda}$ with $\xrightarrow{q,1/\lambda}$, thus obtaining the so called *deterministic temporal graph*, which has the information about the average time associated to every transition (Fig.2).

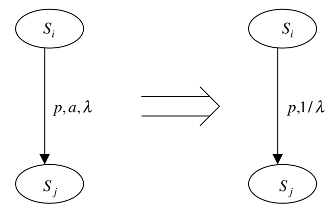

**Figure 2. Evaluation Algorithm - Phase 2**

3. Next, for each one of these graphs, we eliminate those transitions having a zero temporal cost by joining the nodes connected by that transition in a single one, labelled either by $S_i$ or by $S_j$ (Fig.3). The probability of an eliminated arc is going to multiply those probabilities corresponding to the outgoing arcs from this new node $S_i \vee S_j$.

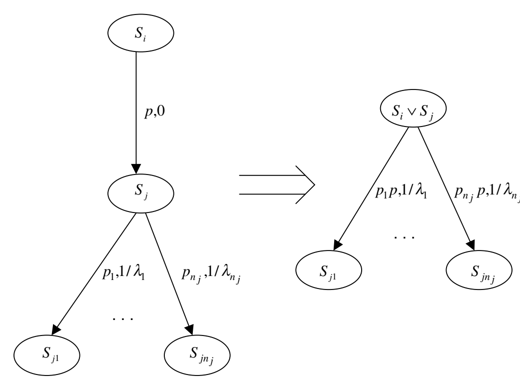

**Figure 3. Evaluation Algorithm - Phase 3**

4. Once we have carried out all these steps, we get a set of *deterministic reduced temporal graphs*. For each one we may compute the required time to reach a concrete state $S_F$ from the initial one $S_0$ (let us observe that each *deterministic reduced temporal graph* only has one initial state, which will be drawn by a double oval, and one final state, represented by a double rectangle).

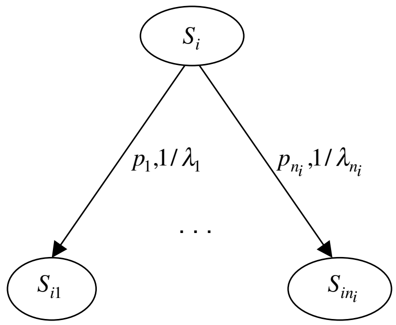


**Figure 4. Evaluation Algorithm - Phase 4**

Then, for each *deterministic reduced temporal graph* we denote by $T_{S_i}$ the expected time to evolve from the state $S_i$ to the final state ($S_F$).

We have ($T_{S_i} = \sum_{j=1}^{n_i} p_j(1/\lambda_j + T_{S_{ij}})$) being $n_i$ the number of outgoing arcs from the state $S_i$, and the label of the arc connecting the state $S_i$ with $S_{ij}$ is $(p_j, 1/\lambda_j)$ as we can see in Fig.4. As a consequence, the average time to reach the final state from the starting one will be $T_{S_0}$, which could be computed by unfolding this recurrent equation.

At this point, we have calculated as many average times as different non-deterministic ways exist (according to the obtained *deterministic reduced temporal graphs*) to evolve from the initial state to the final one (note that the number of different ways is exponential on the number of non-deterministic choices). The highest and the lowest average time thus obtained provide us with time information concerning the reachability of the state $S_F$.

## 5. Strong Stochastic Bisimulation

In this section we define an equivalence relation between ROSA processes, which is based on performance considerations, i.e., it is rather a strong equivalence relation, which discriminates processes not only for their functional behaviour, but even for the times that actions have associated to be executed.

Then, we identify processes that may execute at every state the same actions, but even, they must have associated the same average times.

Therefore, in order to define an equivalence relation on **ROSA** processes, we will distinguish processes like the Classical Process Algebras do (taking into account functional differences), moreover, we consider that those **ROSA** processes that may evolve with the same actions, but having associated different average times are different.

**Definition 3** *Computation*

*$P$ can evolve to $Q$ by executing an action typed with $a$ with probability $r$, with duration modelled by an $Exp[\lambda]$, denoted by $P \overset{r,a,\lambda}{\Longrightarrow} Q$, if and only if:*

- *All the possible different evolutions from $P$ to $Q$ by means of the execution of an action labelled with $a$, make up the set: $\{P \overset{r_i,a,\lambda_i}{\longrightarrow} Q / i \in I \subset \mathbb{N}\}$*
- $r = \sum_{i \in I} r_i$
- $\frac{r}{\lambda} = \sum_{i \in I} \frac{r_i}{\lambda_i}$

**Definition 4** *Strong Bisimulation*

*An equivalence relation $\mathcal{R}$ between **ROSA** processes is a Strong Stochastic Bisimulation if and only if:*
*$\forall P, Q \in \{$**ROSA** $processes\}$ . $(P, Q) \in \mathcal{R}$ iff:*

- *Whenever $P \longrightarrow P'$ there is a **ROSA** process $Q'$ such that $Q \longrightarrow Q'$ with $(P', Q') \in \mathcal{R}$*
- *Whenever $Q \longrightarrow Q'$ there is a **ROSA** process $P'$ such that $P \longrightarrow P'$ with $(Q', P') \in \mathcal{R}$*
- *Whenever $P \overset{r,a,\lambda}{\Longrightarrow} P'$ there is a **ROSA** process $Q'$ such that $Q \overset{r,a,\lambda}{\Longrightarrow} Q'$ with $(P', Q') \in \mathcal{R}$*
- *Whenever $Q \overset{r,a,\lambda}{\Longrightarrow} Q'$ there is a **ROSA** process $P'$ such that $P \overset{r,a,\lambda}{\Longrightarrow} P'$ with $(Q', P') \in \mathcal{R}$*

**Definition 5** *Partition on the set of **ROSA** processes*

*Two **ROSA** processes, $P$ and $Q$ are Strongly Stochastic Bisimilar, denoted by $P \sim_{SSB} Q$, if there exists a strong stochastic bisimulation which contains the pair $(P, Q)$.*

**Definition 6** *Highest and Lowest*

*Let $P$ be a process, and let $\{T_P\}$ be the set of the temporal costs of the different non deterministic computations for that process. We define:*

- *Highest $\{T_P\}$ is the greatest time associated to any computation of $P$.*
- *Lowest $\{T_P\}$ is the lowest time associated to any computation of $P$.*

**Proposition 1** *Temporal consequences*

*$\forall P, Q \in \{$**ROSA** $processes\}$ . $P \sim_{SSB} Q$ implies:*

- *Highest $\{ T_P \}$ = Highest $\{ T_Q \}$*
- *Lowest $\{ T_P \}$ = Lowest $\{ T_Q \}$*

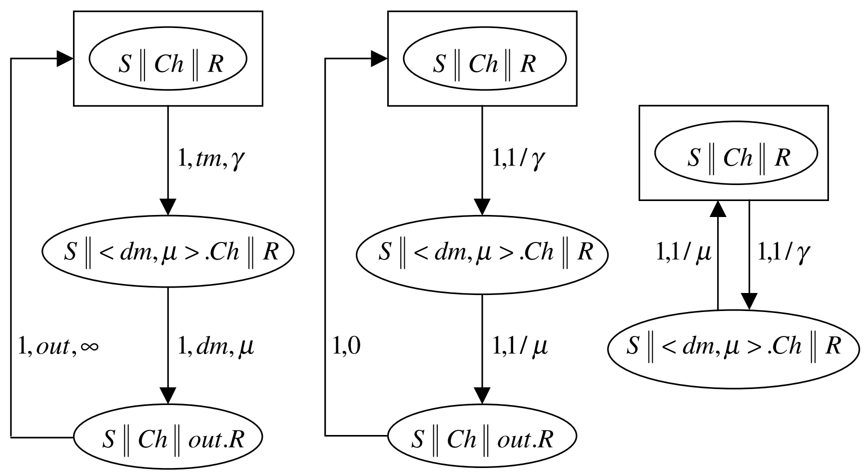


**Figure 5. Case 1 -** *deterministic transition, deterministic temporal,* **and** *deterministic reduced temporal graphs*

## 6. Specification and Analysis of the Alternating Bit Protocol

The alternating bit protocol is a communication protocol that guarantees a secure communication (without replaying messages) between two stations (Sender and Receiver) through a channel which can fail with a known probability, by adding a bit to the messages for controlling the possible loss of them. We study four cases for this protocol:

1. In the first one, we consider the case which consists of 1 sender, 1 receiver and 1 secure channel.
2. Now, we consider again 1 sender, 1 receiver and 1 channel, which can fail with a known probability $p$.
3. In the third case we extend the model to consider $n$ senders, $m$ receivers and 1 channel that may fail with probability $p$.
4. In the final case we will consider $n$ senders, $m$ receivers and $k$ non-secure channels that may all fail with the same probability $p$.

Due to the fact that the Alternating Bit Protocol has not got non-deterministic behaviours, we do not need to deal with the first phase of the evaluation algorithm.

### 6.1. Case 1

Here we have that the sender and the channel must synchronize at $tm$ (transmission of message). The channel and the receiver must synchronize at $dm$ (deliver of message):

| | |
|---|---|
| Sender: | $S = \langle tm, \gamma\rangle.S$ |
| Channel: | $Ch = tm.\langle dm, \mu\rangle.Ch$ |
| Receiver: | $R = dm.out.R$ |

Alternating Bit Protocol 1:

$$ABP \equiv S \;||_{A_1}\; Ch \;||_{A_1}\; R$$
$$A_1 = \{tm, dm\}$$

In Fig.5 we can see from left to right, the *deterministic transition graph, the deterministic temporal graph* and the *deterministic reduced temporal graph* for this case (let us observe that the initial and the final states are the same). From the latest graph we can compute the average time to complete the transmission of a message, as follows:

$T_{S_0} = 1(1/\gamma + T_{S_1}) = 1/\gamma + T_{S_1}$
$T_{S_1} = 1(1/\mu + T_{S_F}) = 1/\mu$

Hence:

$$\mathrm{T} = \frac{1}{\gamma} + \frac{1}{\mu}$$

### 6.2. Case 2

We must consider now, a new process, Activator, which controls the alternation of the added bit to the messages. Both the activator and the sender must synchronize at $in_j$ (which corresponds to add the bit $j$ to the message). The sender and the channel must synchronize at $tm_j$, at $da_j$ and at $\lambda''$, and the channel and the receiver must synchronize at $dm_j$, at $\lambda$, at $ta_j$ and at $\lambda'$. For the sake of clarity we have preferred to separate the channel into two processes, $Go$ and $Ba$ (one for each direction), to capture the different error messages which appear after some delay if the channel has failed in the corresponding direction, these are, $\lambda$ (the transmission fail from sender to receiver), $\lambda'$ (this is the error code transmitted by the receiver when it has received $\lambda$) and $\lambda''$ (which indicates to the sender that the transmission has failed):

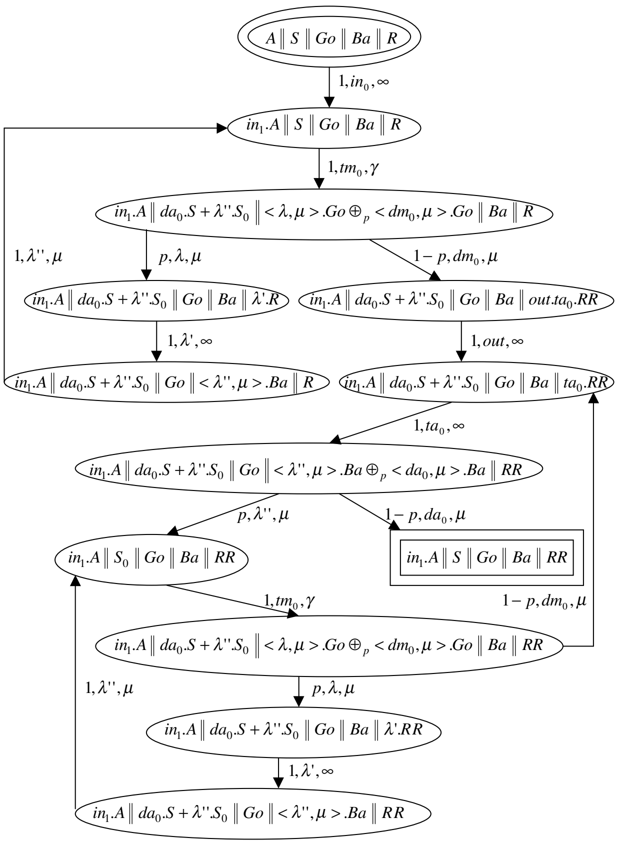


**Figure 6. Case 2 -** *deterministic transition graph*

Activator:
$A = in_0.in_1.A$
Sender:
$S = in_j.S_j$
$S_j = \langle tm_j, \gamma\rangle.(da_j.S + \lambda''.S_j)$
Channels:
$Go = tm_j.(\langle\lambda, \mu\rangle.Go \oplus_p \langle dm_j, \mu\rangle.Go)$
$Ba = ta_j.(\langle\lambda'', \mu\rangle.Ba \oplus_p \langle da_j, \mu\rangle.Ba) + \lambda'.\langle\lambda'', \mu\rangle.Ba$
Receiver:
$R = \lambda.\lambda'.R + dm_0.out.ta_0.RR + dm_1.ta_1.R$
$RR = \lambda.\lambda'.RR + dm_0.ta_0.RR + dm_1.out.ta_1.R$
Alternating Bit Protocol 2:
$ABP \equiv A \ ||_{A_2} \ S \ ||_{A_2} \ Go \ ||_{A_2} \ Ba \ ||_{A_2} \ R$
$A_2 = \{in_j, tm_j, dm_j, ta_j, da_j, \lambda, \lambda', \lambda''\}$

Firstly, let us check that this protocol guarantees that every message always arrives to the receiver:

$P[arrives] =$
$= \sum_{n=0}^{\infty} P[arrives\ at\ first\ time\ in\ attempt(n+1)] =$
$= \sum_{n=0}^{\infty} P[fail\ in\ earlier(n) attempts] \cdot$
$\cdot P[arrives\ at\ attempt(n+1)] =$
$= \sum_{n=0}^{\infty} (1-(1-p)^2)^n \cdot (1-p)^2 =$
$= (1-p)^2/(1-(2p-p^2)) = 1.$

Now, we can calculate the average number of attempts to complete the transmission of a message, as follows:
The random variable which simulates the number of attempts until n successes is a Negative Binomial, then, as we need to study the number of attempts until the first success, where $\mathbf{P}[success] = (1-p)^2$ (success means the channel works fine in both directions), the average number of attempts will be the expected value of $\mathbf{X} \backsim \mathbf{NB}[1, (1-p)^2]$, i.e., $\mathbf{E}[\mathbf{X}] = (1-(1-p)^2)/(1-p)^2 = (2p-p^2)/(1-p)^2$, which is the quotient between the probability with which the channel fails and the probability with which the channel works properly.

Finally, the average time T to complete the communication of a message can be obtained by using the *deterministic reduced temporal graph* (see Fig.7, where now we enumerate the states upside down, and left to right), this graph is reachable from the *deterministic transition graph* (see Fig.6), by means of the second and the third phases of the described method.

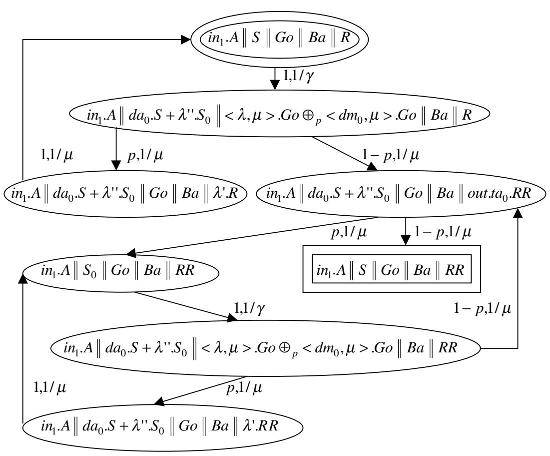


**Figure 7. Case 2 -** *deterministic reduced temporal graph*

Then, according to the Fig.7 we may easily obtain:
$T_{initial\ state} = T_0 = 1/\gamma + T_1$
$T_1 = p(1/\mu + T_2) + (1-p)(1/\mu + T_3)$
$T_2 = 1/\mu + T_0 = 1/\mu + 1/\gamma + T_1$
$T_3 = (1-p)(1/\mu + T_{S_F}) + p(1/\mu + T_4) = \frac{1-p}{\mu} + p(1/\mu + T_4)$
$T_4 = 1/\gamma + T_5$
$T_5 = p(1/\mu + T_6) + (1-p)(1/\mu + T_3)$
$T_6 = 1/\mu + T_4$
$\vdots$

Hence, it can be computed the value for T:

$$\mathrm{T} = \frac{1/\gamma + 2/\mu}{(1-p)^2}$$

### 6.3. Case 3

In this case we have $n$ senders competing for the use of the channel resource, which will be granted in a non-deterministic way.

Graphics for this model and for the next one are the same as above, except for the competition for the channel, which would introduce just a single state after the initial one, where conflicts for assigning the channels are solved, assuming for that a temporal cost equal to 0.

Once the sender $i$ has obtained the channel, it requires $T = (1/\gamma_i + 2/\mu)/(1-p)^2$ to complete the communication. Now our problem is more related to a Classical Queue Theory problem, where we assume that the expected position in the queue is $j+1$, so the sender $i$ should wait the required time for the $j$ preceding senders to complete their transmissions, which is: $\sum_{n=1}^{j}(1/\gamma_n+2/\mu)/(1-p)^2$, hence adding the involved time for its own transmission: $(1/\gamma_i + 2/\mu)/(1-p)^2$, we obtain that:

$$\mathrm{T} = \frac{1/\gamma_i + 2/\mu + \sum_{n=1}^{j}(1/\gamma_n + 2/\mu)}{(1-p)^2}$$

### 6.4. Case 4

We assume that all the channels have the same probability to fail $p$, and that every sender can maintain as many communications in parallel as channels it has got assigned. This model is in essence equal to the preceding one, with the exceptions that now we need for every receiver $2^k$ states and one activator per channel. It has been obtained:

$$\mathrm{T} = \frac{1/\gamma_i + 2/\mu + \sum_{n=1}^{j}(1/k\gamma_n + 2/k\mu)}{(1-p)^2}$$

## 7. Conclusions and Future Work

We have presented an algebraic language, **ROSA**, for the modelling and analysis of probabilistic and non-deterministic systems.

We have defined an operational semantics and an evaluation algorithm for it, which allows to obtain for each process a set of *deterministic reduced temporal graphs* and the average time to reach the final state in each one. Finally, we can obtain the minimum and the maximum average time of the whole system.

Therefore, a Strong Stochastic Bisimulation relation on the set of **ROSA** processes has been defined.

As case studied, we have applied this method for the analysis of the Alternating Bit Protocol in some interesting cases.

Our current work is mainly focused on defining a Weak Stochastic Bisimulation over the set of **ROSA** processes, and on implementing a tool for its practical application.

Finally, we are also working on some other applications of this method, like the study of algorithms for compression and transmission of audio and video sequences.

## References


[1] M. Bernardo and R. Gorrieri. A tutorial on EMPA: A theory of concurrent processes with nondeterminism, priorities, probabilities and time. *Theoretical Computer Science*, 202:1–54, 1998.

[2] B. Bloom and A. Meyer. A remark on bisimulation between probabilistic processes. In Meyers and Tsailin, editors, *Logik at Botik*. Springer-Verlag, 1989.

[3] A. Bueno, V. Valero, and F. Cuartero. TPAL: A timed-probabilistic model for concurrent processes. In *Proceedings of APSEC'97/ICSC'97*, Hong-Kong, 1997.

[4] F. Cuartero, D. de Frutos, and V. Valero. PCSP: A denotational model for probabilistic processes. In *Proceedings of Third AMAST Workshop on Real-Time Systems*, 1996.

[5] F. Cuartero, D. de Frutos, and V. Valero. A sound and complete proof system for probabilistic processes. In *Proceedings of Fourth AMAST Workshop on Real-Time Systems*, 1997.

[6] P. R. D'Argenio, J. P. Katoen, and E. Brinksma. An algebraic approach to the specification of stochastic systems. *Programming Concepts and Methods*, pages 126–147, 1998.

[7] A. Giacalone, C.-C. Jou, and S. Smolka. Algebraic reasoning for probabilistic concurrent systems. In *Proceedings of Working Conference on Programming Concepts and Methods, IFIP TC 2*, Sea of Galilee, Israel, 1990.

[8] H. Hansson and B. Jonsson. A calculus for communicating systems with time and probabilities. In *Proceedings Real-Time Systems Symposium*, Orlando, Florida, 1990.

[9] H. Hermanns, U. Herzog, and J. P. Katoen. Process algebra for performance evaluation. *To appear in Theoretical Computer Science*, 1999.

[10] J. Hillston. A compositional approach to performance modelling. In *Cambridge University Press*. 1996.

[11] B. Jonsson and K. Larsen. Specification and refinement of probabilistic processes. In *LICS'91*, Amsterdam, 1991.

[12] G. Lowe. Representing nondeterministic and probabilistic behaviour in reactive processes. Technical Report PRG-TR-11-93, Oxford university computing laboratory, 1993.

[13] M. Núñez, D. de Frutos, and L. Llana. Acceptance trees for probabilistic processes. In *CONCUR'95, LNCS 962*, pages 249–263, 1995.

[14] F. L. Pelayo, F. Cuartero, V. Valero, and D.Cazorla. PPNAL: Performance evaluation in an algebraic model for probabilistic an non-deterministic processes. In *Proceedings of 15th Annual UK Performance Engineering Workshop*, pages 1–12, Bristol, U.K., 1999.

[15] R. van Glabbeek, S. Smolka, B. Steffen, and C. Tofts. Reactive, generative ans stratified models of probabilistic processes. In *Proceedings of 5th Annual IEEE Symposium on Logic in Computer Science*, pages 130–141, Philadelphia, PA, 1990.

[16] W. Yi and K. Larsen. Testing probabilistic and nondeterministic processes. In *Protocol Specification, Testing and Verification XII*, pages 47–61, 1992.