PPNAL: Performance Evaluation in an Algebraic Model for Probabilistic and Non-deterministic Processes *

F. L. Pelayo † F. Cuartero V. Valero D. Cazorla

Abstract

In this paper we present an Algebraic Language which allows us to describe probabilistic and non-deterministic processes as well as to evaluate index performances: **PPNAL** (**P**erformance **P**robabilistic and **N**on-deterministic **A**lgebraic **L**anguage). We also analyse the Alternating Bit Protocol by using this language, showing the average time to send a message, taking into account that channels may fail with some known probability.

1 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 **PPNAL** are defined by:

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

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

The operators are interpreted as in the 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 belonging to A, and $recX : P$ is the recursion operator. The new operators that we consider 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 λ. Let us observe that now $a.P$ is a particular case of this operator taking $\lambda = \infty$, which means that $\mathbf{E}[Exp[\infty]] = 0$, and then this action has duration 0.

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

---

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

†Department of Computer Science. Escuela Politécnica Superior de Albacete. University of Castilla-La Mancha. Campus Univ. s/n. 02071, Albacete, Spain. E-mail:{fpelayo, fernando, valentin}@info-ab.uclm.es, dcazorla@pol-ab.uclm.es

2 Operational Semantics

The operational semantics is defined in the usual way, by using a labelled transition system. Rules have the following aspect:

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

where r is a probability, a is an action type and λ is a parameter of an Exponential Distribution that models the duration of the action.

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

Rules S4 capture *race policy*, i.e., if an action can be executed by both arguments then the quickest one is selected for its execution. Thus, immediate actions (those having zero duration) have preference over the other ones, i.e., they are urgent, but if we have some of them that can be executed at a certain instant, the choice will be made in a non-deterministic way. When all the possible action have non-zero duration, the action becoming available earlier will be selected.

Rules S5 describe parallel composition semantics which execute actions from either P or Q when they do not belong to the synchronization set A, in the usual way, but when they are in A, the time associated to the action is now obtained taking the minimum of the parameter of this action in both processes, i.e., the slowest one imposes its time over the other one.

Functions **Tips** and **D** are defined as follows:

Tips is a function defined on the set of multisets of actions which associates to every multiset, the set consisting of their action types:

$$\mathbf{Tips} : \mathcal{M}[\Delta \times \mathbb{R}^*_+] \longrightarrow \mathcal{P}[\Delta]$$

D is a function defined on the set of **PPNAL** processes which associates to every process its multiset of available actions:

$$\mathbf{D} : \{\mathbf{PPNAL}\ \textit{processes}\} \rightarrow \mathcal{M}[\Delta \times \mathbb{R}^*_+]$$

3 Performance Evaluation

Once presented the semantic rules, we can generate the *transition graph* for every process P.

So, we can use this graph, in order to evaluate some performance indexes. However, the evaluation method here present only covers system without non-deterministic behaviours. The extension to general systems is now in progress, and will be presented in a future work.

For that, we first relabel every transition: $\xrightarrow{q,a,\lambda}$ with $\xrightarrow{q,1/\lambda}$, thus obtaining the *temporal graph* which has the information of the average time associated to every transition (Fig.1).

Next, we eliminate those transitions having a temporal cost zero joining the nodes connected by that transition in a single one, labelled by S_i or by S_j (Fig.2). The probability of an eliminated arc is then multiplied by the corresponding one

Table 1: Operational Semantics

(S1a) $$\frac{}{a.P \xrightarrow{1,a,\infty} P}$$

(S1b) $$\frac{}{\langle a,\lambda\rangle.P \xrightarrow{1,a,\lambda} P}$$

(S2a) $$\frac{P \xrightarrow{q,a,\lambda} P'}{P \oplus_p Q \xrightarrow{p\cdot q,a,\lambda} P'}$$

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

(S3a) $$\frac{P \xrightarrow{q,a,\lambda} P'}{P \oplus Q \xrightarrow{q,a,\lambda} P'}$$

(S3b) $$\frac{Q \xrightarrow{q,a,\lambda} Q'}{P \oplus Q \xrightarrow{q,a,\lambda} Q'}$$

(S4a) $$\frac{P \xrightarrow{q,a,\lambda} P' \wedge a \notin \mathbf{Tips}[\mathbf{D}[Q]]}{P + Q \xrightarrow{q,a,\lambda} P'}$$

(S4b) $$\frac{Q \xrightarrow{q,a,\lambda} Q' \wedge a \notin \mathbf{Tips}[\mathbf{D}[P]]}{P + Q \xrightarrow{q,a,\lambda} Q'}$$

(S4c) $$\frac{P \xrightarrow{q,a,\infty} P'}{P + Q \xrightarrow{q,a,\infty} P'}$$

(S4d) $$\frac{Q \xrightarrow{q,a,\infty} Q'}{P + Q \xrightarrow{q,a,\infty} Q'}$$

(S4e) $$\frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q'}{P + Q \xrightarrow{q\cdot r,a,\lambda_1+\lambda_2} P'}$$

(S4f) $$\frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q'}{P + Q \xrightarrow{q\cdot r,a,\lambda_1+\lambda_2} Q'}$$

(S4g) $$\frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q'}{P + Q \xrightarrow{q\cdot(1-r),a,\lambda_1} P'}$$

(S4h) $$\frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q'}{P + Q \xrightarrow{(1-q)\cdot r,a,\lambda_2} Q'}$$

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

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

(S5c) $$\frac{P \xrightarrow{q,a,\infty} P' \wedge a \notin A}{P||_A Q \xrightarrow{q,a,\infty} P'||_A Q}$$

(S5d) $$\frac{Q \xrightarrow{q,a,\infty} Q' \wedge a \notin A}{P||_A Q \xrightarrow{q,a,\infty} P||_A Q'}$$

(S5e) $$\frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q' \wedge a \notin A}{P||_A Q \xrightarrow{q\cdot r,a,\lambda_1+\lambda_2} P'||_A Q}$$

(S5f) $$\frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q' \wedge a \notin A}{P||_A Q \xrightarrow{q\cdot r,a,\lambda_1+\lambda_2} P||_A Q'}$$

(S5g) $$\frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q' \wedge a \notin A}{P||_A Q \xrightarrow{q\cdot(1-r),a,\lambda_1} P'||_A Q}$$

(S5h) $$\frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q' \wedge a \notin A}{P||_A Q \xrightarrow{(1-q)\cdot r,a,\lambda_2} P||_A Q'}$$

(S5i) $$\frac{P \xrightarrow{q,a,\lambda_1} P' \wedge Q \xrightarrow{r,a,\lambda_2} Q' \wedge a \in A}{P||_A Q \xrightarrow{q\cdot r,a,\mathbf{min}[\{\lambda_1,\lambda_2\}]} P'||_A Q'}$$

(S6a) $$\frac{P[recX : P/X] \xrightarrow{q,a,\lambda} P'}{recX : P \xrightarrow{q,a,\lambda} P'}$$

of the outgoing arcs from this new node $S_i \vee S_j$. Then, we thus obtain a *reduced temporal graph* and from this graph we may compute the required time in order

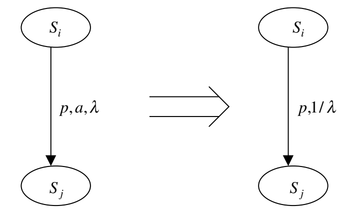

Figure 1: Evaluation Algorithm - Phase 1

to go from the initial state, S_0, to the final one, S_F. Let us observe that the *reduced temporal graph* only has one initial state, which will be drawn by a double oval, and one final state, pictured by a double rectangle.

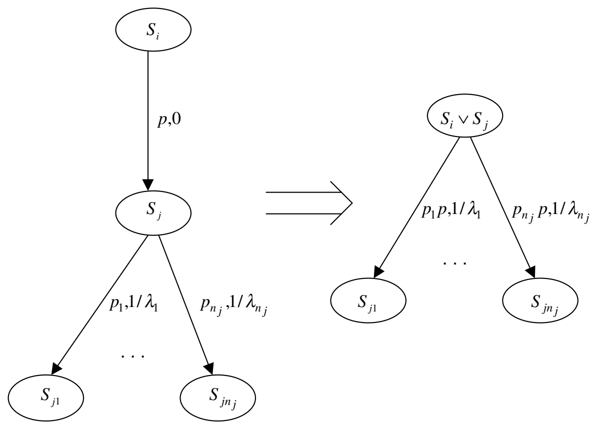

Figure 2: Evaluation Algorithm - Phase 2

Then, let us denote by T_i the expected time to evolve from the state i to the final state.

We have

$$T_i = \sum_{j=1}^{n_i} p_j (1/\lambda_j + T_{S_{ij}})$$

being n_i the number of outgoing arcs from the state i, and the label (Fig.3) of the arc connecting the state i with S_{ij} is $p_j, 1/\lambda_j$. As a consequence, the average time to reach the final state from the starting one, so, will be T_{S_0}, which should be computed by unfolding this recurrent equation.

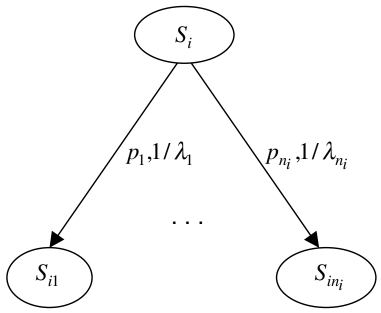

Figure 3: Evaluation Algorithm - Phase 3

4 Specification 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, by adding a bit to the messages for controlling the possible loss of them.

We study four cases for this protocol:

1. The first one 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 2 non-secure channels that may both fail with the same probability, p.

4.1 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):

$$\begin{array}{ll} \text{Sender:} & S = \langle tm, \gamma\rangle.S \\ \text{Channel:} & Ch = tm.\langle dm, \mu\rangle.Ch \\ \text{Receiver:} & R = dm.out.R \end{array}$$

Alternating Bit Protocol 1:

$$\begin{array}{l} ABP \equiv S \ ||_{A_1} \ Ch \ ||_{A_1} \ R \\ A_1 = \{tm, dm\} \end{array}$$

In Fig.4 we can see the *transition graph, temporal graph* and *reduced temporal graph* for this case. From the latest one we can evaluate the average time to complete the transmission of a message, T:

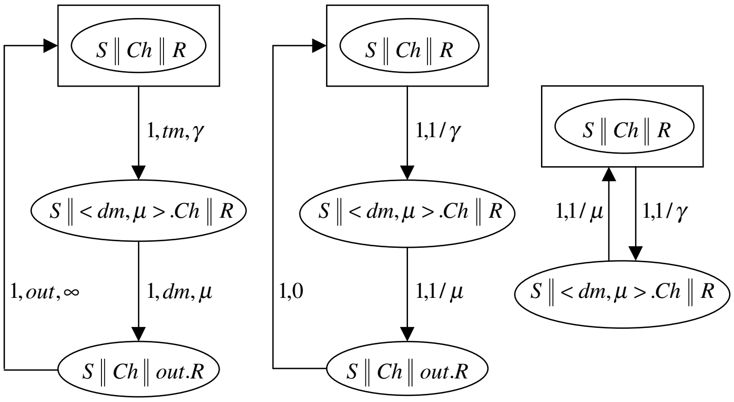

Figure 4: Case 1 - *transition, temporal,* and *reduced temporal graphs*

$T_{S_0} = 1(1/\gamma + T_1) = 1/\gamma + T_1$
$T_1 = 1(1/\mu + T_{S_F}) = 1/\mu$

Hence:

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

4.2 Case 2

Here we must consider a new process, Activator, which controls the alternation of the added bit to the messages. This 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 λ'', and the channel and the receiver must synchronize at dm_j, at λ, at ta_j and at λ'. For the sake of clarity we have preferred to separate the channel into two processes, (Sender$\longrightarrow$Receptor) Go and (in the other direction) $Ba(ck)$, to capture the different error messages which appear after some delay if the channel has fail in the corresponding direction:

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 with this protocol we can guarantee that a message always arrives to the receiver:

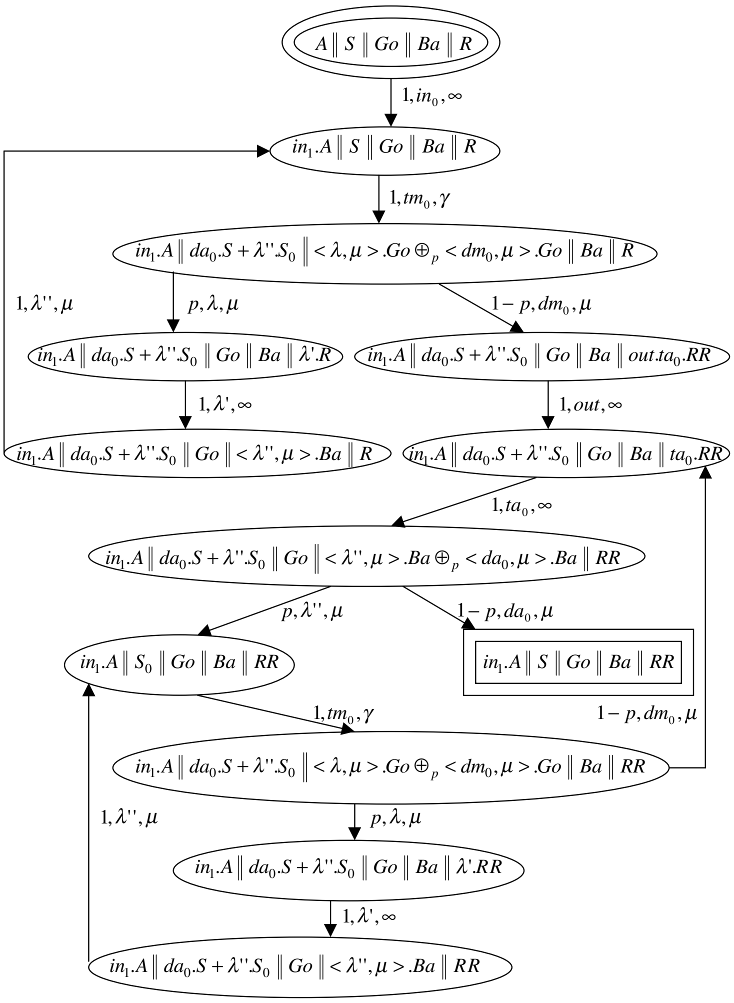

Figure 5: Case 2 - *transition graph*

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

Now we can calculate the average number of attempts to complete the transmission of a message as follows:
First observe that the random variable which simulates the number of attempts until n successes is a Negative Binomial, so, as we need to simulate the number of attempts until the first success, where $\mathbf{P}[success] = (1-p)^2$ (success means the channel works fine in both directions), then 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

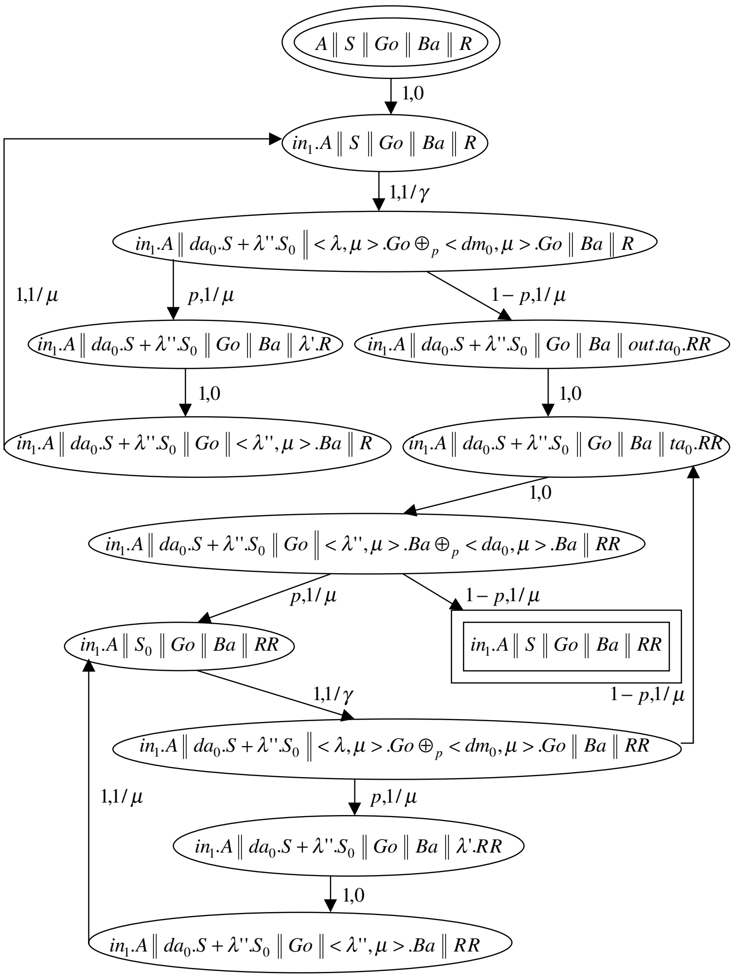

Figure 6: Case 2 - *temporal graph*

which channel works properly.

Finally, the average time T to complete the communication of a message can be obtained as follows, by using the *reduced temporal graph* (see Fig.7 where now we enumerate the states upside down and left to right):

$$
\begin{aligned}
&T_{S_0} = 1/\gamma + T_1 \\
&T_1 = p(1/\mu + T_2) + (1-p)(1/\mu + T_3) \\
&T_2 = 1/\mu + T_{initial\ state} = 1/\mu + 1/\gamma + T_1 \\
&T_3 = (1-p)(1/\mu + T_{S_F}) + p(1/\mu + T_4) = \tfrac{1-p}{\mu} + p(1/\mu + T_4) \\
&T_4 = 1/\gamma + T_5 \\
&T_5 = p(1/\mu + T_6) + (1-p)(1/\mu + T_3) \\
&T_6 = 1/\mu + T_4 \\
&\quad \vdots
\end{aligned}
$$

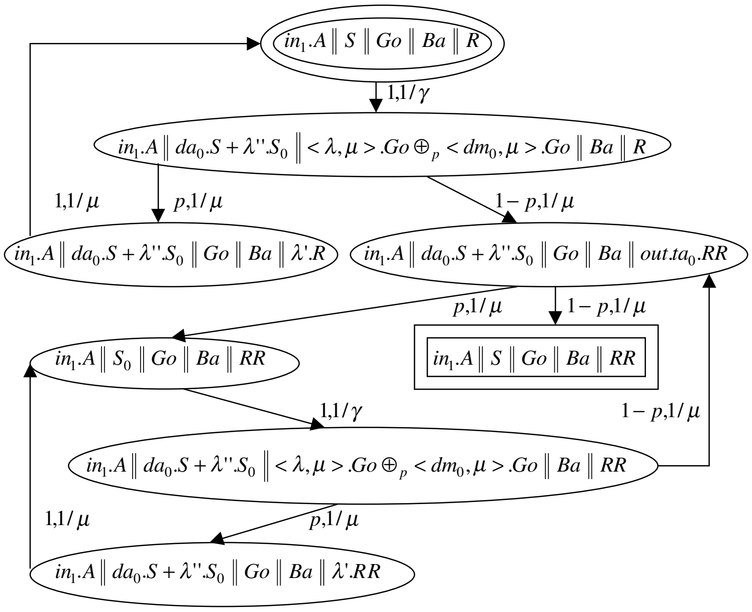

Figure 7: Case 2 - *reduced temporal graph*

Hence:

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

4.3 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:

$$\begin{array}{ll}
\text{Activator:} & A = in_0.in_1.A \\
\text{Senders:} & S = \sum_{i=1}^{n} S_{ik} \\
 & S_{ik} = in_j.S_{ijk} \\
 & S_{ijk} = \langle tm_{ijk}, \gamma_i \rangle.(da_j.S + \lambda''.S_{ijk}) \\
\text{Channels:} & Go = tm_{ijk}.(\langle \lambda, \mu \rangle.Go \oplus_p \langle dm_{jk}, \mu \rangle.Go) \\
 & Ba = ta_j.(\langle \lambda'', \mu \rangle.Ba \oplus_p \langle da_j, \mu \rangle.Ba) + \lambda'.\langle \lambda'', \mu \rangle.Ba \\
\text{Receivers:} & R = \lambda.\lambda'.R \sum_{k=1}^{m} (dm_{0k}.out_k.ta_0.RR + dm_{1k}.ta_1.R) \\
 & RR = \lambda.\lambda'.RR \sum_{k=1}^{m} (dm_{0k}.ta_0.RR + dm_{1k}.out_k.ta_1.R)
\end{array}$$

Alternating Bit Protocol 3:

$$\begin{array}{l}
ABP \equiv A \ ||_{A_3} \ S \ ||_{A_3} \ Go \ ||_{A_3} \ Ba \ ||_{A_3} \ R \\
A_3 = \{in_j, tm_{ijk}, dm_{jk}, ta_j, da_j, \lambda, \lambda', \lambda''\}
\end{array}$$

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 time $T = (1/\gamma_i + 2/\mu)/(1-p)^2$ 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:

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

4.4 Case 4

We assume that both channels have the same probability to fail, p. This model is in essence equal to the preceding one, with the exception that now we need for every receiver 2^2 states. We model the possibility of a sender to maintain as many communications in parallel as channels it has got assigned:

Activators:
$A_0 = in_{00}.in_0.in_{10}.in_0.A_0$
$A_1 = in_{01}.in_1.in_{11}.in_1.A_1$

Senders:
$$S = \sum_{i=1}^{n} S_{ik}$$
$S_{ik} = in_{jg}.S_{ijgk}$
$S_{ijgk} = \langle tm_{ijgk}, \gamma_i\rangle.(da_{jg}.in_g.S + \lambda''_g.S_{ijgk})$

Channels:
$Go_g = tm_{ijgk}.(\langle\lambda_g, \mu\rangle.Go_g \oplus_p \langle dm_{jgk}, \mu\rangle.Go_g)$
$Ba_g = ta_{jg}.(\langle\lambda''_g, \mu\rangle.Ba_g \oplus_p \langle da_{jg}, \mu\rangle.Ba_g) + \lambda'_g.\langle\lambda''_g, \mu\rangle.Ba_g$

Receivers:
$$R = \lambda_0.\lambda'_0.R + \lambda_1.\lambda'_1.R\sum_{k=1}^{m}\sum_{j=0}^{1}\sum_{g=0}^{1} dm_{jgk}.out_k.ta_{jg}.R^{jg}$$
$$R^{jg} = \lambda_0.\lambda'_0.R^{jg} + \lambda_1.\lambda'_1.R^{jg}\sum_{k=1}^{m}\sum_{r=0}^{1}\sum_{s=0}^{1}\begin{cases} dm_{rsk}.ta_{rs}.R^{rs} & r = j \wedge s = g \\ dm_{rsk}.out_k.ta_{rs}.R^{rs} & otherwise \end{cases}$$

Alternating Bit Protocol 4:

$ABP \equiv A_0 \,||_{A_4}\, A_1 \,||_{A_4}\, S \,||_{A_4}\, S \,||_{A_4}\, Go_0 \,||_{A_4}\, Ba_0 \,||_{A_4}\, Go_1 \,||_{A_4}\, Ba_1 \,||_{A_4}\, R \,||_{A_4}\, R$
$A_4 = \{in_{jg}, in_g, tm_{ijgk}, dm_{jgk}, ta_{jg}, da_{jg}, \lambda_g, \lambda'_g, \lambda''_g\}$

The analysis of the necessary time to complete the communication the sender i in this fourth case is similar as the case 3 except for the necessary time to complete their communication the j preceding senders, which in this case is a half of the previous one: $(\sum_{n=1}^{j}(1/\gamma_n + 2/\mu)/(1-p)^2)/2$, so adding its own time: $(1/\gamma_i + 2/\mu)/(1-p)^2$ we finally obtain:

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

This result can be easily generalized to k channels obtaining:

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

So let us observe that in the limit $k \to \infty$ every sender would need the same time as if it had an exclusive channel.

5 Conclusions and Future Work

We have presented an algebraic language, **PPNAL**, for the modelling and analysis of probabilistic and non-deterministic systems. We have defined an operational semantics for this language, which allows us to obtain for each process a *reduced temporal graph*, by applying a reduction method for the labelled transition system associated to it. From this *reduced temporal graph* we have seen how to obtain the average time to reach the final state of it. Concretely, we have applied this method for the analysis of the alternating bit protocol, in four cases we have distinguished.

Our current work is mainly focused on defining an equivalence relation for this language, and the generalization of the presented method of evaluation for processes having non-deterministic behaviours. Finally we are also working on some other non-trivial applications of this method, as well as on a tool for a practical application of it.

References

[BG98] Marco Bernardo and R. Gorrieri. A tutorial on EMPA: A theory of concurrent processes with nondeterminism, priorities, probabilities and time. *Theoretical Computer Science*, 202(1):1–54, 1998.

[CCV98] D. Cazorla, F. Cuartero, and V. Valero. PNAL: Un modelo algebraico para procesos probabilísticos y no deterministas. In *Jornadas de Concurrencia*, Pamplona, 1998.

[GJS90] A. Giacalone, C.-C. Jou, and S.A. 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.

[Hen88] M. Hennessy. *Algebraic Theory of Processes.* MIT Press, 1988.

[HJ90] H. Hansson and B. Jonsson. A calculus for communicating systems with time and probabilities. In *Proceedings Real-Time Systems Symposium*, Orlando, Florida, 1990.

[JL91] B. Jonsson and K.G. Larsen. Specification and refinement of probabilistic processes. In *LICS'91*, Amsterdam, 1991.

[Low93] G. Lowe. Representing nondeterministic and probabilistic behaviour in reactive processes. Technical Report PRG-TR-11-93, Oxford university computing laboratory, 1993.

[Mil89] R. Milner. *Communication and Concurrency.* Prentice Hall, 1989.

[vGSST90] R. van Glabbeek, S.A. Smolka, B.U. Steffen, and C.M.N. 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.

[YL92] W. Yi and K.G. Larsen. Testing probabilistic and nondeterministic processes. In *Protocol Specification, Testing and Verification XII*, pages 47–61, 1992.