CSP Probabilístico (PCSP): Un modelo operacional

F. Cuartero Gómez
Dpto.Informática
Esc. Politécnica
Univ. Castilla-La Mancha
Albacete, SPAIN 02071
Fax: 34 67 506650
utfcuartero@02ccv1.ucma.es

D. de Frutos Escrig
Dpto.Informática y Automática
Fac. Matemáticas
Univ. Complutense
Madrid, SPAIN 28040
Fax: 34 1 5439489
W575 @ EMDUCM11.bitnet

V. Valero Ruiz
Dpto.Informática
Esc. Politécnica
Univ. Castilla-La Mancha
Albacete, SPAIN 02071
Fax: 34 67 506650
utvalero@02ccv1.ucma.es

Resumen

Presentamos un algebra de procesos para la modelización de procesos comunicantes probabilísticos. El tipo de álgebra es asíncrono, al igual que los modelos clásicos, y contrariamente a la mayoría de álgebras probabilísticas estudiadas hasta el momento (PCCS), que son síncronas. La principal dificultad en nuestro modelo proviene de disponer en el mismo de dos tipos de elección (interna y externa), al contrario que en los anteriores, donde sólo existe uno. Otra dificultad, que nuestro modelo no alcanza a resolver es el funcionamiento del operador de ocultamiento. Si queremos mantener su significado clásico, necesitaríamos añadir más potencia a nuestra semántica. Definiremos una semántica operacional del lenguaje, y la usaremos para razonar sobre su potencia mediante un ejemplo no trivial, en el que mostraremos la posibilidad de modelizar propiedades como la seguridad (*reliability*), y el cálculo de tiempos medios (*performance*).

1 Introducción

Los algoritmos aleatorizados están siendo utilizados cada vez con mayor asiduidad. En particular ya han comenzado a ser utilizados en la ya de por sí compleja programación de sistemas distribuidos (veáse [11]). El uso de estos algoritmos redunda en una mayor velocidad y simplificación de las soluciones. Incluso, en ocasiones no existe alternativa a la aleatorización de las soluciones. Por tanto, los formalismos matemáticos utilizados en la especificación de sistemas en los que aparecen comportamientos aleatorios debe incluir componentes probabilísticas. Los mismos permitirán en particular estudiar la corrección de dichos sistemas, que a diferencia de lo que sucede en el caso determinista, no vendrá determinada por alguna propiedad que sólo pueda ser cierta o falsa, sino que puede venir determinada por la probabilidad de que una cierta propiedad se cumpla.

El lenguaje *Communicating Sequential Processes (CSP)* [5] ofrece un formalismo adecuado para la especificación de sistemas distribuidos no deterministas. Sus principales ventajas radican en servir como soporte de un razonamiento algebraico efectivo en el tratamiento de procesos concurrentes. Nuestra intención en el presente trabajo es construir una versión probabilística lo más sencilla posible de CSP que combine toda su potencia (al menos en lo que se refiere al tratamiento de la elección) con las nociones

probabilísticas de corrección (que también pueden extenderse a la terminación). Este lenguaje lo hemos denominado CSP probabilístico (PCSP).

2 Sintaxis de PCSP

La sintaxis de PCSP es una sencilla variante de la de CSP, con la única diferencia de añadir una distribución de probabilidades a todos aquellos operadores que conllevan una elección, interna o externa. La resumimos mediante la expresión BNF

$$P ::= STOP \mid X \mid a \rightarrow P \mid [p_1]P \sqcap [p_2]P \mid [p_1]P \,\Box\, [p_2]P$$

$$\mid [p_1]P \parallel_A [p_2]P \mid P\backslash(a,p) \mid \mu X.P$$

donde $p \in [0,1]$, p_1, $p_2 \in [0,1]$, con $p_1 + p_2 = 1$ y $X \in Idf$.

El significado intuitivo de cada uno de los operadores es el siguiente:

STOP y PREFIJO: STOP es el proceso que simboliza una máquina rota, incapaz de realizar ninguna acción. Dados un proceso P y una acción a, $a \rightarrow P$ es un proceso que ejecuta en primer lugar la acción a, y después pasa a comportarse como el proceso P.

ELECCIÓN INTERNA: Dados dos procesos P y Q, y $p_1, p_2 \in [0,1]$ tales que $p_1 + p_2 = 1$, entonces $[p_1]P \sqcap [p_2]Q$ es un proceso que se comporta como P con probabilidad p_1 o como Q con probabilidad p_2.

ELECCIÓN EXTERNA: Dados dos procesos P y Q, y $p_1, p_2 \in [0,1]$ tales que $p_1 + p_2 = 1$, entonces $[p_1]P \,\Box\, [p_2]Q$ es un proceso cuyo comportamiento es principalmente determinista. Si el entorno del proceso decide ejecutar una acción que puede ser ejecutada por P o Q, pero no por ambos, será ejecutada por el proceso correspondiente con la probabilidad con la que este proceso puede ejecutar la acción, continuando después en consecuencia.

Sólo en el caso de que ambos puedan realizar dicha acción la elección será aleatoria, de acuerdo a las probabilidades p_1 y p_2. Sin embargo, aún en este caso, la elección externa no sería equivalente a la interna, ya que además de las dos probabilidades citadas hay que tener en cuenta las probabilidades que cada proceso tiene de aceptar la ejecución de dicha acción. Así, la probabilidad de ejecutar una acción es mayor en el proceso resultante de elegir entre dos, que en cualquiera de ellos.

COMPOSICIÓN PARALELA: Dados dos procesos P y Q, $p_1, p_2 \in [0,1]$ tales que $p_1 + p_2 = 1$ y un conjunto de acciones A, entonces $[p_1]P \parallel_A [p_2]Q$ es un proceso que corresponde a la ejecución concurrente de P y Q, sincronizando sobre las acciones de A. Dada una acción $a \in A$, para que ésta pueda realizarse, ambos procesos deben estar en condiciones de ejecutarla, obteniéndose como resultado de esa ejecución una única acción a. Para las acciones no contenidas en el conjunto A de sincronización, los procesos evolucionan independientemente, salvo que ambos puedan realizar la citada acción, en cuyo caso se decide a quién corresponde la ejecución atendiendo a las probabilidades p_1 y p_2, de la misma forma que en la elección externa.

OCULTACIÓN: Dado un proceso P y dado un par (a,q); con $a \in \Sigma$ y $q \in [0,1]$, entonces $P\backslash(a,q)$ es un proceso en el que las ejecuciones de la acción oculta a no pueden ser observadas. La probabilidad q tiene por objeto decidir sobre las elecciones externas en el caso de que aparezca la acción a como una de las posibles opciones. En este caso, el entorno no siempre tiene la posibilidad de elegir, porque puede ocurrir que el sistema

decida autónomamente ejecutar la acción oculta. Esta elección entre la acción oculta y las observables será entonces aleatoria, y la probabilidad con la que el sistema decidirá elegir la acción oculta será q.

RECURSIÓN: La recursión nos permite expresar patrones de comportamiento para procesos infinitos. Podemos ver la semántica de este operador como la solución de un sistema de ecuaciones mutuamente recursivas, donde los identificadores representan los procesos incógnitas.

3 Semántica operacional de PCSP

La semántica que presentaremos, y que posteriormente utilizaremos para razonar sobre los comportamientos de los procesos, será una semántica operacional. Así, vamos a definir la evolución o ejecución de un proceso mediante un sistema probabilístico de transiciones (veáse [7]), donde las transiciones serán aquellas generadas por un conjunto de reglas, cada una de las cuales contribuye a definir la semántica de alguno de los operadores del lenguaje. Seguiremos el estilo estructurado de Plotkin [10] y Milner [9].

Definición 1 Una *transición probabilística observable* es una tupla $\langle P, Q, a, p\rangle$ donde P y Q son procesos, $a \in \Sigma$, y $p \in (0,1]$. Usualmente representaremos una transición de la forma siguiente:

$$P \xrightarrow{a}_p Q$$

□

Seguidamente, presentamos y comentamos las reglas de transición correspondientes al conjunto de operadores del lenguaje.

1) $\dfrac{}{a \to P \xrightarrow{a}_1 P}$

2) $\dfrac{P \xrightarrow{a}_p Q}{[q]P \sqcap [1-q]R \xrightarrow{a}_{pq} Q}$

3) $\dfrac{P \xrightarrow{a}_p Q}{[q]R \sqcap [1-q]P \xrightarrow{a}_{p(1-q)} Q}$

4) $\dfrac{P \xrightarrow{a}_p P',\ Q \xrightarrow{a}_q}{[r]P \,\square\, [1-r]Q \xrightarrow{a}_s P'}$
donde $s = pqr + p(1-q)$

5) $\dfrac{P \xrightarrow{a}_p ,\ Q \xrightarrow{a}_q Q'}{[r]P \,\square\, [1-r]Q \xrightarrow{a}_s Q'}$
donde $s = pq(1-r) + (1-p)q$

6) $\dfrac{P \xrightarrow{a}_p P' ,\ Q \xrightarrow{a}_{p'} ,\ a \notin A}{[q]P \parallel_A [1-q]Q \xrightarrow{a}_s [q]P' \parallel_A [1-q]Q}$
donde $s = pp'q + p(1-p')$

7) $\dfrac{P \xrightarrow{a}_{p'} ,\ Q \xrightarrow{a}_p Q' ,\ a \notin A}{[1-q]P \parallel_A [q]Q \xrightarrow{a}_s [1-q]P \parallel_A [q]Q'}$
donde $s = pp'q + p(1-p')$

8) $\dfrac{P \xrightarrow{a}_p P' ,\ Q \xrightarrow{a}_{p'} Q' ,\ a \in A}{[q]P \parallel_A [1-q]Q \xrightarrow{a}_{pp'} [q]P' \parallel_A [1-q]Q'}$

9) $\dfrac{P[P/X] \xrightarrow{a}_p P'}{\mu X.P \xrightarrow{a}_p P'}$

donde $P[P/X]$ representa la sustitución sintáctica en P del identificador X por el proceso P.

Observese que en algunas de estas reglas aparece la noción derivada $P \xrightarrow{a}_p$ cuya definición se incluye a continuación.

Definición 2 i) Llamamos semántica operacional del lenguaje PCSP (sin operador de ocultamiento) al multiconjunto de transiciones probabilísticas que pueden ser inferidas con el sistema de reglas anterior, y donde cada transición aparece tantas veces como formas diferentes haya de derivarla.

ii) Sea a una acción y P un proceso. Diremos que p es la probabilidad con la que P ejecuta la acción a, lo que representaremos de la forma

$$P \xrightarrow{a}_{p}$$

si $p = \sum_{i \in I} p_i$ donde I es el multiconjunto de transiciones

$$P \xrightarrow{a}_{p_i} P_i$$

que ejecutan la acción a, que pueden ser derivadas usando las reglas. □

Las tres primeras reglas son suficientemente claras y no merecen más comentario. En las reglas cuarta y quinta, la probabilidad de que el proceso $[r]P \,\square\, [1-r]Q$ ejecute la acción a y pase a comportarse como P', o Q', según cada transición generada, se calcula considerando la posibilidad de que o bien P o bien Q ejecuten dicha acción. Sin embargo, si ambos procesos pueden ejecutar dicha acción, sólo uno de ellos podrá ser elegido para hacerlo, y esta elección se hará teniendo en cuenta la probabilidad r indicada en la elección externa, que define la probabilidad con la que seleccionaremos el primer proceso cuando esta situación ocurre.

Las reglas sexta y séptima cubren el caso de *interleaving*. Con ellas se describe que si la acción a ejecutar no pertenece al conjunto de sincronización, será realizada por el proceso componente que esté capacitado para ello. Si ambos procesos lo están, se elegirá aleatoriamente, de manera análoga a como se ha descrito para el operador de elección externa, atendiendo a la probabilidad asociada al operador composición paralela, que también indica la probabilidad con la que será seleccionada en tales situaciones la primera componente.

La octava regla describe el caso en que la acción pertenece al conjunto de sincronización. Deseamos entonces que la acción sea ejecutada por ambos procesos simultáneamente, y ésto, obviamente ocurrirá con una probabilidad igual al producto de las correspondientes a las transiciones combinadas de los procesos componentes.

Finalmente, la última regla trata el operador de la recursión en la forma tradicional, no necesitando por ello de ningún comentario adicional.

Veamos ahora porqué es necesario trabajar con multiconjuntos al definir la semántica operacional del lenguaje. Consideremos al respecto el proceso $P = [\frac{1}{2}]a \rightarrow STOP \sqcap [\frac{1}{2}]a \rightarrow STOP$. Tenemos que

$\mathcal{T}_1 = P \xrightarrow{a}_{\frac{1}{2}} STOP$ es derivable aplicando las reglas 1 y 2, mientras
$\mathcal{T}_2 = P \xrightarrow{a}_{\frac{1}{2}} STOP$ lo es aplicando las reglas 1 y 3.

Con ambas derivaciones se infiere que P puede ejecutar la acción a con probabilidad $\frac{1}{2}$. Sin embargo, parece necesario que quede constancia de que se trata de dos computaciones diferentes, a fin de que pueda inferirse a la postre que efectivamente la acción a será ejecutada con probabilidad 1.

Sin embargo, hemos preferido evitar una formalización explícita del multiconjunto de transiciones derivadas como el utilizado en [3], donde las transiciones están indexadas con índices que indican la vía en que han sido generadas, para no perder al lector con un tecnicismo adicional, que no añade ninguna idea nueva.

Comentamos por último el tema del operador de ocultación. Para el mismo no hemos presentado regla alguna, lo que naturalmente implica que no ha quedado definida su semántica. Ello se debe a que lamentablemente no es posible capturar su comportamiento deseado, en el marco de una semántica que razone en base al estudio de una acción por vez.

El problema radica en el hecho de que al aplicar el operador de ocultamiento sobre una elección externa, ésta se convierte en parcialmente interna. En cambio el significado de las elecciones internas no varía al aplicarles un ocultamiento. Consideremos al respecto el siguiente

Ejemplo 1 Sean P y Q los siguientes procesos:

$$\begin{aligned} P = &\ [\tfrac{1}{2}]([\tfrac{1}{2}]a \to STOP \Box [\tfrac{1}{2}]b \to STOP) \sqcap [\tfrac{1}{2}]STOP \\ Q = &\ [\tfrac{1}{2}]a \to STOP \sqcap [\tfrac{1}{2}]b \to STOP \end{aligned}$$

Ambos procesos son equivalentes respecto de cualquier noción basada en la presente semántica operacional, pues la misma es idéntica para ambos. Sin embargo, si ocultamos la acción a el comportamiento deseado pasa a ser diferente, pues en $P\backslash(a,q)$ la probabilidad de ejecutar la acción b disminuye, mientras que en $Q\backslash(a,q)$ sigue siendo $\frac{1}{2}$, lo que es imposible de lograr si efectivamente las semánticas de P y Q son idénticas. □

4 Computaciones de procesos probabilísticos

En la sección anterior, hemos introducido el conjunto de reglas que definen las transiciones de los procesos probabilísticos. No obstante, las transiciones sólo definen el primer paso de la evolución de un proceso. Por ello, necesitamos un razonamiento más potente, encadenando transiciones para adentrarnos en la ejecución de los procesos.

Definición 3 Llamamos computaciones de un proceso probabilístico P a las secuencias de transiciones:

$$P = P_0 \xrightarrow{a_1}_{p_1} P_1, \ \ldots\ , P_{n-1} \xrightarrow{a_n}_{p_n} P_n$$

incluídas en la semántica operacional de PCSP. Usualmente las representaremos en la forma

$$\mathcal{C} = P \xrightarrow{a_1}_{p_1} P_1 \xrightarrow{a_2}_{p_2} \ldots \xrightarrow{a_n}_{p_n} P_n$$

Llamaremos una *derivada* de P generada por $\mathcal{C}$ al triple $\langle s, P_n, p\rangle$, donde $s = a_1 \ldots a_n$ y $p = p_1 \cdots p_n$. Usualmente representaremos estas derivadas con la notación:

$$P \overset{s}{\Longrightarrow}_p P_n$$

De nuevo cada proceso tiene un multiconjunto de computaciones, en el que cada computación aparece tantas veces como indica el producto de las veces que aparecen las componentes en la semántica del lenguaje. □

La noción de derivada puede ser fácilmente formalizada en la forma siguiente:

Definición 4 Sea s una secuencia (posiblemente vacía) de acciones, $p \in (0,1]$ y P, Q dos procesos. Entonces, $P \overset{s}{\Longrightarrow}_p Q$ es una computación si y sólo si es posible inferirla por medio del uso de las reglas:

1. $P \overset{\langle\rangle}{\Longrightarrow}_1 P$
2. Si $P \xrightarrow{a}_p P'$ y $P' \overset{s}{\Longrightarrow}_q Q$, entonces $P \overset{as}{\Longrightarrow}_{pq} Q$

□

5 Ejemplo. Protocolo AUY

Presentaremos un ejemplo en el cual modelamos el protocolo AUY (ver [1]) utilizado en la transmisión de datos, que asegura una transmisión segura sobre un sistema en el que los canales de comunicación pueden fallar. Este ejemplo también se ha estudiado en [3], sobre el marco del álgebra de procesos (PCCS) allí presentado. Sin embargo, en nuestra opinión, este tipo de álgebra síncrona plantea algunos problemas que dificultan el modelado de sistemas. Unos derivados de la sincronía, pues es necesario sincronizar *a mano* todas las componentes del sistema que no están sincronizadas de una manera natural, y otros, derivados de la inexistencia de un operador explícito de elección externa, que sólo puede ser parcialmente simulado por medio de una adecuada combinación de las probabilidades en el único operador de elección que incluye, junto con la restricción de los procesos a un adecuado conjunto de acciones observables. Este mecanismo no es muy intuitivo, puesto que un cambio en las probabilidades, o una restricción a un conjunto inadecuado de acciones, implicaría la desaparición de la posibilidad de esta interpretación de la elección como externa. Por otra parte, no resulta en absoluto evidente generalizar esta simulación a una ocurrencia cualquiera de la elección externa.

En nuestro lenguaje estos problemas desaparecen, puesto que disponemos de un operador paralelo asíncrono, y un operador explícito de elección externa, ambos con una semántica adecuada.

Por simplicidad, omitiremos a lo largo del ejemplo las probabilidades en las apariciones de los operadores de elección externa y composición paralela, puesto que de hecho no son necesarias, ya que nunca aparecerán elecciones entre procesos que puedan ejecutar como primera acción una misma.

El sistema consiste de dos agentes, un transmisor y un receptor, que se comunican mediante mensajes. El transmisor envía un mensaje al receptor usando un canal no seguro, que puede perderlo con una probabilidad conocida p. Suponemos que si el mensaje original se pierde durante la transmisión, el canal lo reemplazará con un mensaje nulo λ, que enviará al receptor. Este tomará el mensaje de su canal de entrada, y si no es λ, lo enviará al exterior, devolviendo un mensaje de reconocimiento al transmisor que enviará por un segundo canal. De nuevo, el canal podrá perder el mensaje (suponemos que con la misma probabilidad p, aunque esta suposición no es necesaria), en cuyo caso el transmisor recibirá un mensaje nulo λ'. Si el transmisor recibe esta señal λ', repetirá la emisión, enviando de nuevo el mensaje original. Finalmente, cuando el transmisor reciba la señal de reconocimiento podrá comenzar un nuevo ciclo.

El proceso PCSP que modela este protocolo es el proceso P definido como sigue:

Transmisor:

$$T = in \rightarrow T_1$$
$$T_1 = msg \rightarrow (ack \rightarrow T \,\Box\, \lambda' \rightarrow T_1)$$

Canal Primero:

$$C_1 = msg \rightarrow ([p]\lambda \rightarrow C_1 \sqcap [1-p]msg' \rightarrow C_1)$$

Canal Segundo:

$$C_2 = (ack' \rightarrow ([p]\lambda' \rightarrow C_2 \sqcap [1-p]ack \rightarrow C_2)) \Box (\lambda'' \rightarrow \lambda' \rightarrow C_2)$$

Receptor:

$$R = (msg' \rightarrow out \rightarrow ack' \rightarrow R) \Box (\lambda \rightarrow \lambda'' \rightarrow R)$$

Protocolo AUY:

$$P = T \|_{\{msg,ack,\lambda'\}} ((C_1 \|_{\emptyset} C_2) \|_{\{msg',ack',\lambda,\lambda''\}} R)$$

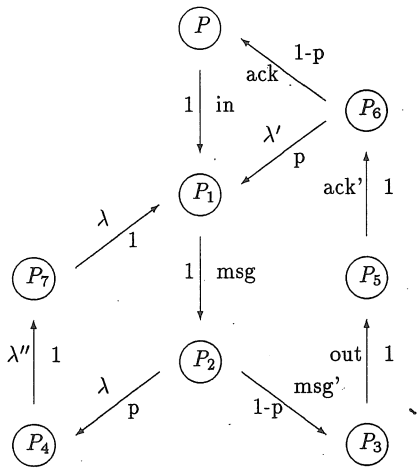

Figura 1: Sistema Probabilístico de Transiciones para el Protocolo AUY

El comportamiento de este proceso se muestra en el sistema probabilístico de transiciones representado en la figura 1. En donde:

$$
\begin{aligned}
P_1 &= T_1 \|_{\{msg,ack,\lambda'\}}((C_1 \|_\emptyset C_2)\|_{\{msg',ack',\lambda,\lambda''\}} R) \\
P_2 &= (ack \rightarrow T \Box \lambda' \rightarrow T_1)\|_{\{msg,ack,\lambda'\}}((([p]\lambda \rightarrow C_1 \sqcap [1-p]msg' \rightarrow C_1)\|_\emptyset C_2) \\
&\quad \|_{\{msg',ack',\lambda,\lambda''\}} R) \\
P_3 &= (ack \rightarrow T \Box \lambda' \rightarrow T_1)\|_{\{msg,ack,\lambda'\}}((C_1 \|_\emptyset C_2)\|_{\{msg',ack',\lambda,\lambda''\}}(out \rightarrow ack' \rightarrow R)) \\
P_4 &= (ack \rightarrow T \Box \lambda' \rightarrow T_1)\|_{\{msg,ack,\lambda'\}}((C_1 \|_\emptyset C_2)\|_{\{msg',ack',\lambda,\lambda'\}}(\lambda'' \rightarrow R)) \\
P_5 &= (ack \rightarrow T \Box \lambda' \rightarrow T_1)\|_{\{msg,ack,\lambda'\}}((C_1 \|_\emptyset C_2)\|_{\{msg',ack',\lambda,\lambda''\}}(ack' \rightarrow R)) \\
P_6 &= (ack \rightarrow T \Box \lambda' \rightarrow T_1)\|_{\{msg,ack,\lambda'\}}((C_1 \|_\emptyset([p]\lambda' \rightarrow C_2 \sqcap [1-p]ack \rightarrow C_2)) \\
&\quad \|_{\{msg',ack',\lambda,\lambda''\}} R) \\
P_7 &= (ack \rightarrow T \Box \lambda' \rightarrow T_1)\|_{\{msg,ack,\lambda'\}}((C_1 \|_\emptyset(\lambda'' \rightarrow C_2))\|_{\{msg',ack',\lambda,\lambda''\}} R)
\end{aligned}
$$

Nuestro lenguaje nos permite razonar sobre diferentes propiedades del protocolo, tales como la fiabilidad (*reliability*) y el tiempo esperado de transmisión. Por ejemplo, es sencillo comprobar que la probabilidad con la que un ciclo termina tras un bucle simple es $(1-p)^2$. Sin embargo, el hecho más importante es que todos los elementos de la modelización del sistema son muy naturales, y no hemos necesitado ningún truco como el que aparece (y necesariamente!) en el modelo presentado en [3].

Para finalizar, formalizaremos seguidamente la prueba de la seguridad del protocolo (cualquier mensaje será transmitido, y el reconocimiento se recibirá con probabilidad 1), y calcularemos la longitud media de un ciclo de transmisiones.

Cualquier computación de nuestro sistema comenzará con la transición

$$P \xrightarrow{in}_1 P_1$$

y un ciclo terminará cuando el estado P se recupere.

Desde P_1 tenemos las siguientes computaciones correspondientes a un bucle básico del protocolo:

$$P_1 \xrightarrow{s_1}_{p_1} P_1 \quad s_1 = \langle msg\ \lambda\ \lambda''\ \lambda' \rangle \ \text{ y } \ p_1 = p$$
$$P_1 \xrightarrow{s_2}_{p_2} P_1 \quad s_1 = \langle msg\ msg'\ out\ ack'\ \lambda' \rangle \ \text{ y } \ p_2 = p(1-p)$$
$$P_1 \xrightarrow{s_3}_{p_3} P \quad s_1 = \langle msg\ msg'\ out\ ack'\ ack \rangle \ \text{ y } \ p_3 = (1-p)^2$$

Entonces, denotaremos con $f(n)$ la probabilidad de completar el ciclo tras n bucles fallidos, o lo que es lo mismo tras $n+1$ bucles básicos. De las probabilidades de las computaciones previas, este valor puede ser calculado como sigue:

$$f(n) = (p + p(1-p))^n (1-p)^2 = (1-p)^2 p^n (2-p)^n$$

Entonces la probabilidad de finalizar el ciclo en tiempo finito viene dada por la expresión

$$\sum_{i=0}^{\infty} f(i) = (1-p)^2 \sum_{i=0}^{\infty} (2p-p^2)^i = \frac{(1-p)^2}{1-(2p-p^2)} = \frac{(1-p)^2}{(1-p)^2} = 1$$

Esto prueba que el protocolo es seguro, puesto que la probabilidad de tener una transmisión correcta es 1.

El tiempo medio de emisión puede ser calculado por medio del número de emisiones realizadas hasta que un reconocimiento se reciba por el transmisor. Entonces, el tiempo medio de emisión (μ) viene dado por

$$\mu = \sum_{i=0}^{\infty} i\, f(i) = (1-p)^2 \sum_{i=0}^{\infty} i(2p-p^2)^i = (1-p)^2 \frac{2p-p^2}{(1-(2p-p^2))^2} =$$
$$(1-p)^2 \frac{p(2-p)}{(1-p)^4} = \frac{p(2-p)}{(1-p)^2}$$

6 Conclusiones y Trabajos Futuros

Hemos presentado una semántica operacional simple para un lenguaje que permite el modelado de sistemas concurrentes con comportamiento probabilístico. La base del sistema es el clásico CSP, en el que intentamos mantener el significado usual de sus operadores.

Este trabajo constituye sólo un primer paso de nuestra labor en el campo, y en el mismo no se ha conseguido cubrir el operador de ocultamiento, tema en el que nos encontramos trabajando en la actualidad. Para ello, además de controlar en cada paso la acción que se ejecuta, será necesario conocer además el contexto en que dicha acción es ejecutada, esto es, el conjuntos de acciones posibles de ejecutar en ese momento, lo que denominaremos el *estado* del proceso en ese instante. Por lo que una transición será

$$P \xrightarrow{a,A}_{p} P'$$

que interpretamos como: El proceso P alcanza el estado A, y tras ejecutar la acción a pasa a comportarse como Q con probabilidad p.

Estamos también trabajando en una semántica denotacional del lenguaje, que puede caracterizarse por medio de una semántica de pruebas, así como en una axiomatización completa en el caso finito, de la misma. Todo lo cual formará parte de la Tesis Doctoral del primero de los autores, de próxima finalización.

Si bien los desarrollos anunciados en el parrafo anterior son de una complejidad técnica bastante mayor a lo que se presenta en el presente trabajo, el mismo tiene para nosotros el interés de haber sido el punto de partida sin el cual nada habría sido posible, reuniendo las ideas intuitivas básicas para nuestro trabajo, y que resultan más fáciles de comprender en un marco técnicamente m'as simple, como el aquí presentado.

Referencias

[1] A.V.Aho, J.D.Ullman, M.Yannakakis. *Modeling communication protocols by automata.* Proc. 20 th. IEEE Symp. on Foundations of Computer Science, 267-273,1979.

[2] Ivan Christoff. *Testing Equivalences and Fully Abstract Models for Probabilistic Processes.*, CONCUR-90, LNCS 458, 1990.

[3] A. Giacalone, C.-C. Jou, S.A. Smolka. *Algebraic reasoning for probabilistic concurrent systems.* Pr. Working Conference on Programming Concepts and Methods. IFIP TC 2, Israel, 1990.

[4] R. van Glabbek, S.A. Smolka, B.U.Steffen, C.M.N.Tofts, *Reactive, generative and stratified models of probabilistic processes.* Proceedings of 5th Annual IEEE Symposium on Logic in Computer Science, 130-141, Philadelphia, PA, 1990.

[5] C.A.R. Hoare. *Communicating Sequential Processes.* Prentice-Hall International 1985.

[6] C.-C. Jou, S.A. Smolka. *Equivalences, Congruences and Complete Axiomatizations for Probabilistic Processes.* CONCUR-90, LNCS 458, 1990.

[7] K.G.Larsen, A.Skou *Bisimulation through probabilistic testing.* Proceedings of 16th Annual ACM Symposium on Principles of Programming Languages, 1989.

[8] C. Miguel. *State of the Art on Timed and Probabilistic Models.* Internal Report DIT - Univ. Politécnica de Madrid, 1991.

[9] R. Milner. *Communication and Concurrency.* Prentice-Hall International, 1985.

[10] G.D. Plotkin. *A structural approach to operational semantics.* Technical Report DAIMI FN-19, Aarhus University. 1981.

[11] W. Pugh. *Skip Lists: A Probabilistic Alternative to Balanced Trees* Communications of the ACM. June, 1990.