Modeling Systems by Probabilistic Process Algebra: An Event Structures Approach

Joost-Pieter Katoen [a] and Rom Langerak [a] and Diego Latella [b*]

[a]Department of Computer Science, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands, e-mail: { katoen, langerak }@cs.utwente.nl

[b]CNR, Ist. CNUCE, Via S. Maria 36, 56126 Pisa, Italy, e-mail: latella@fdt.cnuce.cnr.it

Performance and reliability analysis of distributed systems based on formal specifications is an important and widely recognized issue. This paper treats a probabilistic version of (a subset of) the process algebra LOTOS. It incorporates a probabilistic choice assigning a probability of occurrence to each of its alternatives. Opposed to the traditional interleaving semantics used for existing probabilistic process algebras the presented language is based on a true concurrency semantics. This enables us to distinguish between non-determinism and parallelism, to reduce the state explosion problem and, moreover, to analyse part of the system without considering other (irrelevant) parts. In this paper the language is presented and the formal semantics is defined by using an extension of bundle event structures. A short example illustrates the novelties of the language and links the language to stochastic analysis based on semi-Markov chains.

Keyword Codes: C.2.4; C.4; D.2.1; F.3.2; F.4.3
Keywords: Distributed Systems; Event Structures; LOTOS; Performance of Systems; Requirements/Specification; Semantics

1. INTRODUCTION

The process algebra LOTOS is an ISO standard for the specification of distributed systems [7]. In this paper we extend (a subset of) this language with a *probabilistic choice* operator in order to model the stochastic behaviour of systems, which is important for performance and reliability analysis.

The semantics we propose in this paper is a probabilistic extension of an event structures variant, namely *bundle event structures*, introduced in [10, 11]; this extension is called *probabilistic BES* [2].

Being a "true concurrency" model, *BES distinguish* between parallelism and non-determinism. This provides a good basis for probabilistic reasoning since probabilistic information is usually associated to non-determinism and not to parallelism. Moreover, the direct representation of parallelism leads to state spaces of reasonable size. This reduces the *state explosion* problem which is typical for other concurrency models and which makes stochastic analysis prohibitive. Finally, *BES* allow for *local* analysis: i.e. if one

*This work has been partially funded by the CNR-NATO Advanced Fellowships Program.

[2]In the sequel, the following abbreviations will be used: *bes* for *bundle event structure*, *BES* for *bundle event structures*, π*bes* for *probabilistic bes*, and π*BES* for *probabilistic BES*.

is interested in analyzing only part of a given system, it is relatively easy to isolate the information related to that part from the whole bundle event structure.

To our knowledge, this is the first proposal of a *Probabilistic Process Algebra* (PPA) with a true concurrency semantics. Most PPAs proposed in literature are based either on algebras for *clock-synchronous* processes, (e.g. [4]), or on *interleaving semantics* (e.g. [20, 5]). Synchronous algebras are suitable for modeling hardware components or synchronous systems like systolic arrays etc. However, there are many systems, like communication protocols, that cannot be adequately described by these algebras. On the other hand, the interleaving approach does not distinguish parallelism from non-determinism, suffers from the state explosion problem, and lacks locality.

The language we propose, called $\mathcal{L}$ in the sequel, is a simple extension of a subset of Basic LOTOS [1]. It includes *action-prefix* $(\mu\,;\,B)$, *non-deterministic choice* $(B_1\,[]\,B_2)$, and *parallel composition* $(B_1\,|[\,G\,]|\,B_2)$ without value passing. $\mathcal{L}$ is enriched with a binary *probabilistic choice* operator $[]_p$. Under the assumption that the choice between B_1 and B_2 *cannot* be influenced by the external environment, $B_1\,[]_p\,B_2$ non-deterministically chooses to behave either like B_1 or B_2, the probability to behave like B_1 (respectively B_2) being p (respectively $1{-}p$). In our language, the above assumption is met by means of *syntactical constraints* on B_1 and B_2 and allows us to think of $B_1\,[]_p\,B_2$ as a stochastic experiment (see Section 2).

Other approaches [20, 2, 4, 24, 8, 15, 16, 19, 22, 23] allow the use of probabilities in contexts in which the probabilistic choice can be influenced by the external environment. Such a possibility leads to a more complicated semantics since it must deal with stochastic experiments that are *not* independent and, consequently, it must allow the dynamic redefinition of the probability space of behaviours depending on the context in which a given behaviour expression is placed. On the other hand, there are many applications for which the use of probabilities in the above contexts is not necessary since the phenomena one usually wants to describe by probabilistic behaviours are typically out of control from the environment (like faults, for instance) [5, 18].

In the present proposal probabilistic choice is restricted to be performed between processes the first action of which are required to be *silent* moves, denoted by i. So, for instance, probabilistic choices like $a; B_1\,[]_p\,a; B_2$ and $a; B_1\,[]_p\,i; B_2$ are not taken into consideration here although their non-probabilistic counterparts express instances of internal non-determinism. The reason for this choice is to keep our model as simple as possible. It is worth reminding that $a; B_1\,[]\,a; B_2$ is testing equivalent [17] to $i; a; B_1\,[]\,i; a; B_2$ and that $a; B_1\,[]\,i; B_2$ is testing equivalent to $i; ((a; B_1)\,[]\,B_2)\,[]\,i; B_2$, so that the model we propose is still expressive enough as long as reasoning modulo testing equivalence is acceptable.

This paper is organized as follows. In Section 2 the syntax of $\mathcal{L}$ is defined. πBES are defined in Section 3 and in Section 4 a πBES-based semantics for $\mathcal{L}$ is given. Moreover, in Section 4 it is shown that using *BES* semantics the correctness relation is *identity.* That is, the probabilistic bundle event structure for an arbitrary expression e ($e \in \mathcal{L}$) is identical to the (ordinary) bundle event structure associated with the LOTOS counterpart of e. Finally, in Section 5 a sample specification is given, on which some stochastic analysis is performed.

2. SYNTAX OF $\mathcal{L}$

We define the set of well-formed expressions by means of an abstract syntax and an extra constraint on this syntax.

Definition 2.1 *Subset of LOTOS*

$$B ::= \mathbf{stop} \;\Big|\; (\mu\,;\,B) \;\Big|\; (B\,[]\,B) \;\Big|\; (B\,[]_p\,B) \;\Big|\; (B\,|[\,G\,]|\,B) \quad .$$

□

With $G \subset \mathit{Gates}$, with *Gates* the set of LOTOS gate-identifiers, $\mu \in \mathit{Gates} \cup \{\, i \,\}$ and probability $p \in (0,1)$ (that is, $0 < p < 1$). Parentheses are omitted when not necessary. The set of terms generated by the above grammar is denoted by *Bex*. We remark that there would be no intrinsic difficulties in dealing with a richer subset of LOTOS including also **exit**, enabling, hiding, and so on. For the sake of clarity and simplicity these constructs are omitted here.

The idea is that probabilistic choices give rise to stochastic experiments, that is, sets of possible events where *each* event is assigned a certain probability. As the use of probabilistic choices together with non-deterministic choices leads to alternatives without a probability assigned to them, we want to avoid the intermixing of these two forms of choices. For similar reasons parallel compositions cannot be operands of probabilistic choices. (The effect of these restrictions is reflected in lemmas 4.5 and 4.6, see Section 4.) These ideas are formalized as follows.

Definition 2.2 *Predicate $\mathcal{PI}$*

$\mathcal{PI} : \mathit{Bex} \rightarrow \mathrm{Bool}$ is defined as follows:

$$\begin{array}{lcl}
\mathcal{PI}(\mathbf{stop}) & = & \text{false} \\
\mathcal{PI}(\mu\,;\,B) & = & \text{false} \\
\mathcal{PI}(B_1\,[]\,B_2) & = & \text{false} \\
\mathcal{PI}(B_1\,[]_p\,B_2) & = & \text{true} \\
\mathcal{PI}(B_1\,|[\,G\,]|\,B_2) & = & \mathcal{PI}(B_1) \;\vee\; \mathcal{PI}(B_2) \quad .
\end{array}$$

□

Informally, $\mathcal{PI}$ identifies (parallel compositions of) *probabilistic choices* (possibly with other constructs). As it will be clear from the semantics of $\mathcal{L}$ the initial events of such expressions will be assigned probabilities (see Section 4).

Definition 2.3 *Language $\mathcal{L}$*

Expression $B \in \mathcal{L}$ if and only if $B \in \mathit{Bex}$ and

1. $B = B_1\,[]\,B_2 \;\Rightarrow\; \neg\,(\mathcal{PI}(B_1) \;\vee\; \mathcal{PI}(B_2))$
2. $B = B_1\,[]_p\,B_2 \;\Rightarrow\; (\exists\, B_1', B_1'', q : B_1 = B_1'\,[]_q\,B_1'' \;\vee\; B_1 = i\,;\,B_1')$
   $\qquad\qquad\qquad \wedge\; (\exists\, B_2', B_2'', r : B_2 = B_2'\,[]_r\,B_2'' \;\vee\; B_2 = i\,;\,B_2') \quad .$

□

3. THE πBES MODEL

As already mentioned in Section 1 the semantics of $\mathcal{L}$ is based on a true concurrency model and is defined by using an extension to *bundle event structures*. Bundle event structures [10] consist of labelled events which model the occurrence of actions, together with relations of causality, conflict and independence between events. Causality is a relation between a set of events X (bundle set) and an event e. The interpretation is that if e happens, at least one of the events in X must have happened already. In order to make the causal dependencies between events in a system run unambiguous all events in X must be pairwise in conflict, so if e happens, exactly one event in X has happened already. Conflict is a symmetric binary relation between events and the intended meaning is that when events a and b are in conflict, they can never both happen in a single system run. When there is neither a conflict nor causal relation between events the events are said to be independent. This means that if independent events are enabled they can occur in any order or simultaneously.

Definition 3.1 *Bundle event structure*

A *bundle event structure* $\mathcal{E}$ is a 4-tuple $(E, \#, \mapsto, l)$ with:

- E, a set of *events*
- $\# \subseteq E \times E$, the *conflict relation*
- $\mapsto \subseteq 2^E \times E$, the *causality relation*
- $l : E \rightarrow Act$, the *action-labeling* function, where Act is a set of action labels

such that the following properties hold:

1. $\forall X \subseteq E, e \in E : X \mapsto e \Rightarrow (\forall e1, e2 \in X : e1 \neq e2 \;\Rightarrow\; e1\#e2)$
2. $\#$ is *irreflexive* and *symmetric* .

$\square$

BES are graphically represented in the following way. Events are denoted as dots; near the dot the action label is given. Conflicts are indicated by dotted lines. A bundle (X, e) is indicated by drawing an arrow from each element of X to e and connecting all lines by small lines.

Essentially, a πbes is a *bes* where some events are labeled also by probabilities. As mentioned in Section 1 probabilities are assigned to internal events only. See for example Figure 1. In particular, probabilities are associated to those internal events which can be grouped together so as to model a discrete probability space. That is, such events must be grouped into "cluster" of mutually conflicting events. Notice that in a given πbes there can be more than one cluster as in Figure 1(c,d). Different clusters just represent different (possibly independent, as in Figure 1(c)) stochastic experiments. Notice, finally, that requiring clusters to represent stochastic experiments implies that all the events in any cluster must be "pointed to" by the same bundle sets. Actually, the probability label associated to an event e, is the *conditional* probability of e to happen *given that* e is enabled. A cluster meaningfully represents a stochastic experiment *only* if all events in the cluster are enabled when e is enabled. The above concepts are formalized in the following:

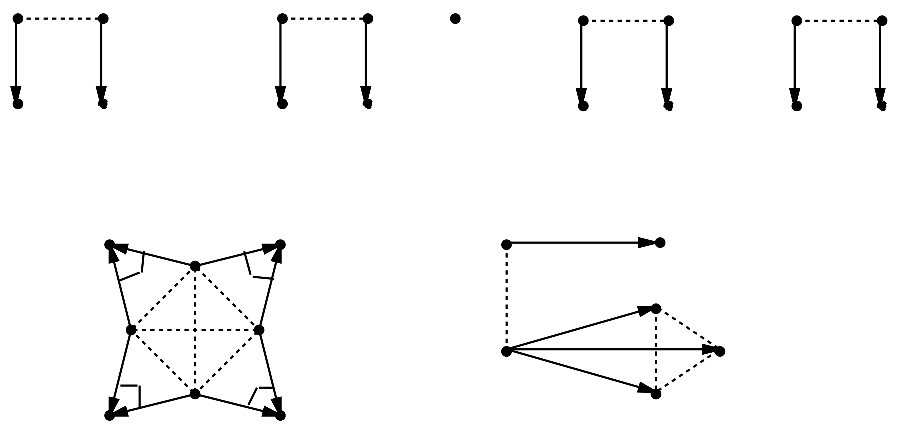

Figure 1. Sample πBES.

Definition 3.2 *Cluster*

For $(E, \#, \mapsto, l)$, a *cluster* is a set Q of events, $Q \subseteq E$, satisfying $\forall e \in Q$:

1. $l(e) = i$
2. $Q \setminus \{ e \} = \{ e' \mid e' \# e \} \ \wedge \ Q \setminus \{ e \} \neq \varnothing$
3. $\forall e' \in Q, X \subseteq E : X \mapsto e \Leftrightarrow X \mapsto e'$.

□

Notice that constraint 2. implies that all distinct events in Q are in conflict with one another and are not in conflict with events *not* in Q.

Definition 3.3 *Probabilistic bundle event structure*

A *probabilistic* bundle event structure (πbes) is a 5-tuple $(E, \#, \mapsto, l, \pi)$ with $(E, \#, \mapsto, l)$ a *bes* and $\pi : E \to (0, 1)$ a partial function, the *probability-labeling* function, such that for all $e \in dom(\pi)$ and $Q = \{ e' \mid e' \# e \} \cup \{ e \}$ we have [3]:

1. Q is a cluster
2. $Q \subseteq dom(\pi)$
3. $\sum_{e' \in Q} \pi(e') = 1$.

□

Figure 2 shows some *BES* which are not πBES; in particular the structure in Figure 2(a) violates condition 2 of definition 3.2 whereas Figure 2(b) does not fulfill condition 3 of definition 3.2.

[3]For any function $f : A \to B$, $dom(f)$ is defined as $\{ a \mid \exists b : b = f(a) \}$.

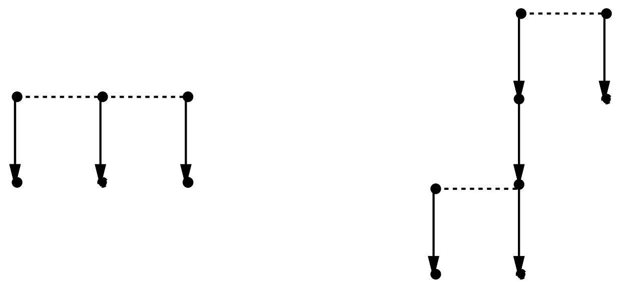

Figure 2. Sample *BES* that are not π*BES*.

Once *assigned* probability distributions to clusters, the next step is to "reason" about them. This means *computing* probabilities for the dynamic representations of an event structure, namely *configurations* [10]. We first of all recall some definitions. Let $\mathcal{E} = (E, \#, \mapsto, l, \pi)$ be a π*bes*.

Definition 3.4 *Proving sequence of* $\mathcal{E}$

A *proving sequence* of $\mathcal{E}$ is a sequence of distinct events $e_1 \ldots e_n \in E$, satisfying:

1. $\{e_1, \ldots, e_n\}$ is conflict-free, i.e. $\forall e_i, e_j : \neg (e_i \# e_j)$, and
2. $\forall X \subseteq E : X \mapsto e_i \;\Rightarrow\; \{ e_1, \ldots, e_{i-1} \} \cap X \neq \varnothing$ for $1 < i \leq n$.

□

Definition 3.5 *Configuration*

A set $C \subseteq E$ is called a *configuration* if there is a proving sequence $e_1 \ldots e_n$ such that $C = \{ e_1, \ldots, e_n \}$.

□

The concept of configuration can be thought of as to correspond intuitively to a system run.

It is worth noting that in the general case, π being a *partial* function, the set of *all* configurations of a π*bes* does not generate a random space. That is, there are configurations for which it *does not* make sense to speak about probabilities. For instance, it does not make sense to speak about the probability of configuration $\{ e_5 \}$ for the π*bes* of Figure 3. Also, there are *sets* of configurations the elements of which must be *indistinguishable* from the probabilistic point of view. For instance, again with reference to Figure 3, the following configurations are probabilistically indistinguishable:

$$c_1 = \{ e_1 \}, c_2 = \{ e_1, e_5 \}, c_3 = \{ e_1, e_3 \}, c_4 = \{ e_1, e_3, e_5 \} \quad .$$

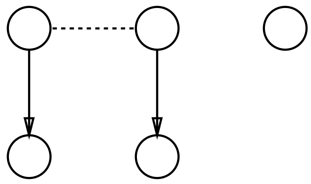

Figure 3. A π*bes*.

In other words, whenever it is known that a configuration belonging to the set $\{ c_1, c_2, c_3, c_4 \}$ has happened (i.e. its events have happened) it does not make sense to reason about the probability that *a particular* one has happened. On the other hand, all the above configurations share a common feature, viz. the fact that event e_1 has been chosen; moreover, the probability of choosing e_1 is $\frac{1}{3}$ (notice that e_1 is the only event appearing in the configurations above for which π is defined).

So, the only question which makes sense in this example is "What is the probability of having *any* configuration which contains e_1?". The answer to this question can be put in the following way: $\mathbb{P}(\{ c_1, c_2, c_3, c_4 \}) = \frac{1}{3}$. Below we will formalize this idea. Let $\mathcal{E} = (E, \#, \mapsto, l, \pi)$ be a π*bes* and $\mathcal{C}_{\mathcal{E}}$ be the set of configurations of $\mathcal{E}$.

Definition 3.6 *Relation* $\rightleftharpoons$

For configurations $C_1, C_2 \in \mathcal{C}_{\mathcal{E}}$ binary relation $\rightleftharpoons$ is defined as

$$C_1 \rightleftharpoons C_2 \;\Leftrightarrow\; (C_1 \cap dom(\pi) = C_2 \cap dom(\pi)) \quad .$$

□

Lemma 3.7 $\rightleftharpoons$ is an equivalence relation on configurations.

Let $[C]_{\rightleftharpoons}$ denote the equivalence class of configuration C under $\rightleftharpoons$. That is, $[C]_{\rightleftharpoons}$ contains all configurations of $\mathcal{E}$ that are equivalent (under $\rightleftharpoons$) to C. An equivalence class $[C]_{\rightleftharpoons}$ is represented by $C \cap dom(\pi)$, which is called the *stochastic choice* of $[C]_{\rightleftharpoons}$.

Definition 3.8 *Probability of configurations*

For a set of configurations V and π*bes* $\mathcal{E}$ such that $V = [C]_{\rightleftharpoons}$ for some $C \in \mathcal{C}_{\mathcal{E}}$ and $C \cap dom(\pi) \neq \varnothing$, the probability of V, denoted $\mathbb{P}(V)$, is defined by

$$\mathbb{P}(V) = \prod_{e \in C \cap dom(\pi)} \pi(e) \quad .$$

□

Stated in words, the probability of a set of configurations is defined for (non-empty) equivalence classes of configurations (under $\rightleftharpoons$) and is equivalent to the probability of the stochastic choice of this equivalence class.

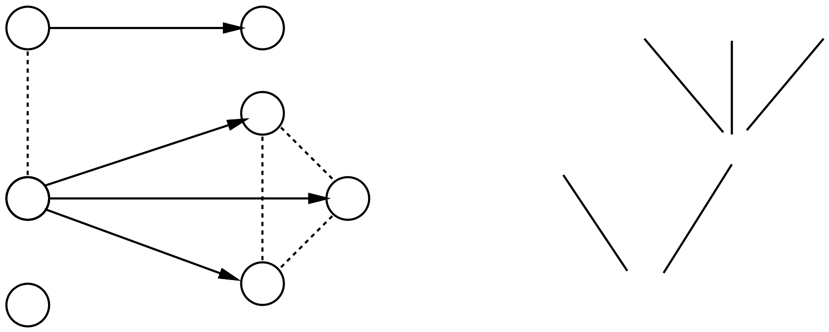

Figure 4. An example πbes (a) and its stochastic choices (b).

Table 1
Equivalence classes, stochastic choices and function $\mathbb{P}$ for πbes of Figure 4(a).

Equivalence Class	Stochastic Choice	$\mathbb{P}$
$\varnothing, \{x_7\}$	$H_0 = \varnothing$	undefined
$\{x_1\}, \{x_1, x_7\}, \{x_1, x_3\}, \{x_1, x_3, x_7\}$	$H_1 = \{x_1\}$	2/3
$\{x_2\}, \{x_2, x_7\}$	$H_2 = \{x_2\}$	1/3
$\{x_2, x_4\}, \{x_2, x_4, x_7\}$	$H_3 = \{x_2, x_4\}$	1/6
$\{x_2, x_5\}, \{x_2, x_5, x_7\}$	$H_4 = \{x_2, x_5\}$	1/12
$\{x_2, x_6\}, \{x_2, x_6, x_7\}$	$H_5 = \{x_2, x_6\}$	1/12

As an example consider the πbes of Figure 4(a). The equivalence classes, stochastic choices and probability function of this πbes are summarized in Table 1.

Notice that the set of *all* stochastic choices of $\mathcal{E}$ does *not* constitute a random space, since the sum of its probabilities may differ from 1. The set of *maximal* (under set inclusion) stochastic choices of $\mathcal{E}$ constitutes a random space [4]. Other (sub-)spaces can be derived from the set of maximal stochastic choices by replacing all stochastic choices containing a given stochastic choice H by H itself. For the πbes of Figure 4(a) we have the following two random spaces (see Figure 4(b)): $S_1 = \{H_1, H_2\}$ and $S_2 = \{H_1, H_3, H_4, H_5\}$. Notice that S_1 can be obtained from S_2, since $H_2 \subseteq H_3$, $H_2 \subseteq H_4$, and $H_2 \subseteq H_5$.

4. πBES SEMANTICS OF $\mathcal{L}$

In this section we give a πBES semantics to $\mathcal{L}$. We define a mapping $[\![\ ]\!]$ that maps each expression B of $\mathcal{L}$ to a πbes $[\![B]\!]$. This function is a straightforward extension of the

[4]A set X is called maximal w.r.t. set inclusion iff $\forall Y : \neg(X \subseteq Y)$.

BES-semantics for LOTOS given in [10], the only difference from which is the addition of the component dealing with probabilities. First, we extend the definition of *init* to π*BES*.

Definition 4.1 *Set of initial events*

For $\pi bes\ \mathcal{E} = (E, \#, \mapsto, l, \pi)$, $init(\mathcal{E})$ is the set $\{\, e \in E \mid \neg\,(\exists\, X \subseteq E : X \mapsto e)\,\}$.

□

The definition of the semantics of **stop**, action-prefix, and non-deterministic choice is quite straightforward and is given below. In the sequel let $[\![\, B_1 \,]\!] = \mathcal{E}_1 = (E_1, \#_1, \mapsto_1, l_1, \pi_1)$ and $[\![\, B_2 \,]\!] = \mathcal{E}_2 = (E_2, \#_2, \mapsto_2, l_2, \pi_2)$ with $E_1 \cap E_2 = \varnothing$. We suppose there is an infinite universe E_U of events.

Definition 4.2 **stop**, *action-prefix and non-deterministic choice*

$[\![\, \mathbf{stop} \,]\!] = (\varnothing, \varnothing, \varnothing, \varnothing, \varnothing)$

$[\![\, \mu\, ;\ B_1 \,]\!] = (E, \#_1, \mapsto, l, \pi_1)$ where

- $E = E_1 \cup \{\, e \,\}$ for some $e \in E_U \setminus E_1$
- $\mapsto\ = \ \mapsto_1 \cup (\{\, \{\, e \,\}\,\} \times init(\mathcal{E}_1))$
- $l = l_1 \cup \{\, (e, \mu) \,\}$

$[\![\, B_1 \,[]\, B_2 \,]\!] = (E_1 \cup E_2, \#, \mapsto_1 \cup \mapsto_2, l_1 \cup l_2, \pi_1 \cup \pi_2)$ where

- $\# = \#_1 \cup \#_2 \cup (init(\mathcal{E}_1) \times init(\mathcal{E}_2))$.

□

Apart from the probability part π the semantics of the probabilistic expression $B{=}B_1 \,[]_p\, B_2$ is equivalent to the non-deterministic choice. For non-initial events of B, π is defined as the union of π_1 and π_2. For initial events the situation is slightly more complicated. All probabilities of initial events of B_1 must be multiplied with p and those of B_2 with $1{-}p$. In order to do so we have to distinguish between events that are assigned a probability within B_1 and B_2 and those that are not. We thus obtain:

Definition 4.3 *Probabilistic choice*

$[\![\, B_1 \,[]_p\, B_2 \,]\!] = (E_1 \cup E_2, \#, \mapsto_1 \cup \mapsto_2, l_1 \cup l_2, \pi)$ where

- $\# = \#_1 \cup \#_2 \cup (init(\mathcal{E}_1) \times init(\mathcal{E}_2))$
- $$\begin{aligned} \pi = {} & (\pi_1 \cup \pi_2 \setminus \{\, (e, r) \mid e \in init(\mathcal{E}_1) \cup init(\mathcal{E}_2) \ \wedge\ r \in (0,1) \,\}) \\ & \cup \{\, (e, p) \mid e \in init(\mathcal{E}_1) \ \wedge\ e \notin dom(\pi_1) \,\} \\ & \cup \{\, (e, pq) \mid e \in init(\mathcal{E}_1) \ \wedge\ (e, q) \in \pi_1 \,\} \\ & \cup \{\, (e, 1{-}p) \mid e \in init(\mathcal{E}_2) \ \wedge\ e \notin dom(\pi_2) \,\} \\ & \cup \{\, (e, (1{-}p)q) \mid e \in init(\mathcal{E}_2) \ \wedge\ (e, q) \in \pi_2 \,\} \ . \end{aligned}$$

□

Finally we define the semantics of the parallel composition operator. We want to define $[\![B_1 \,|[G]|\, B_2]\!]=\mathcal{E}$. The events of $\mathcal{E}$ are constructed in the following way: event e that does not need to synchronize (i.e. $l(e) \notin G$) is paired with the auxiliary symbol $*$, and an event that occurs in both processes is paired with all events in the other process that are equally labeled. Thus events are pairs of events of B_1 and B_2, or with one component equal to $*$. Two events are now put in conflict if any of their components are in conflict, or if different events have a common component different from $*$. A bundle is introduced such that if we take the projection on the i-th component (i=1, 2) of all events in the bundle we obtain a bundle in $[\![B_i]\!]$. Finally, events are assigned a probability when one of their components is equal to $*$ and the other component is assigned a probability in $[\![B_i]\!]$. Note that no redefinition of the probabilities takes place, as usually in interleaving approaches.

Definition 4.4 *Parallel composition*

$[\![B_1 \,|[G]|\, B_2]\!] = (E, \#, \mapsto, l, \pi)$ where

- $E = (E_1^f \times \{*\}) \cup (\{*\} \times E_2^f) \cup \{\, (e_1, e_2) \in E_1^s \times E_2^s \mid l(e_1) = l(e_2) \,\}$, where
  - $E_j^s = \{\, e \in E_j \mid l(e) \in G \,\}, j = 1, 2$ (synchronization events)
  - $E_j^f = E_j \setminus E_j^s$ (non-synchronizing events)
- $(e_1, e_2)\#(e_1', e_2') \Leftrightarrow (e_1 \#_1 e_1') \;\vee\; (e_2 \#_2 e_2') \;\vee$
  $(e_1 = e_1' \neq * \;\wedge\; e_2 \neq e_2') \;\vee\; (e_2 = e_2' \neq * \;\wedge\; e_1 \neq e_1')$
- $X \mapsto (e_1, e_2) \Leftrightarrow$
  $\exists\, X_1 \subseteq E_1 : (X_1 \mapsto_1 e_1 \;\wedge\; X = \{\, (e_j, e_k) \in E \mid e_j \in X_1 \,\}) \vee$
  $\exists\, X_2 \subseteq E_2 : (X_2 \mapsto_2 e_2 \;\wedge\; X = \{\, (e_j, e_k) \in E \mid e_k \in X_2 \,\})$
- $l((e_1, e_2)) =$ **if** $e_1 = *$ **then** $l_2(e_2)$ **else** $l_1(e_1)$
- $\pi = \{\, ((e, *), p) \mid (e, p) \in \pi_1 \,\} \cup \{\, ((*, e), p) \mid (e, p) \in \pi_2 \,\}$.

□

The following behaviour expressions correspond to the πBES of Figure 1 (trailing **stop**s are omitted, and $|||$ denotes $|[\varnothing]|$):

(a) $i\,;a\,[\,]_{1/3}\,i\,;b$

(b) $(i\,;a\,[\,]_{1/3}\,i\,;b)\,|||\,c$

(c) $(i\,;a\,[\,]_{1/3}\,i\,;b)\,|||\,(i\,;c\,[\,]_{3/4}\,i\,;d)$

(d) $(((i;(a\,|||\,c))\,[\,]_{3/4}\,(i;(a\,|||\,d)))\,[\,]_{1/3}\,((i;(c\,|||\,b))\,[\,]_{3/4}\,(i;(d\,|||\,b))))\,|[a,b,c,d]|\,(a\,|||\,b\,|||\,c\,|||\,d)$

(e) $i\,;(i\,[\,]_{1/2}\,(i\,[\,]_{1/2}\,i))\,[\,]_{1/3}\,i\,;a$.

The following lemmas follow from the definition of $[\![\;]\!]$ and the syntactical constraints of Section 2. The proofs of the lemmas are omitted in this paper and can be found in [9].

Lemma 4.5 For all $B \in \mathcal{L} : \neg\mathcal{PI}(B) \Rightarrow init([\![B]\!]) \cap dom(\pi) = \varnothing$.

Lemma 4.6 For all $B_1, B_2 \in \mathcal{L} : B_1\,[\,]_p\,B_2 \in \mathcal{L} \Rightarrow init([\![B_1\,[\,]_p\,B_2]\!])$ is a cluster .

The following lemma states the relationship between expressions of $\mathcal{L}$ and the πBES model.

Lemma 4.7 For all $B \in \mathcal{L}\ : [\![B]\!]$ is a πbes .

For all $B \in \mathcal{L}$ let B_L denote the LOTOS expression obtained from B by systematically replacing all probabilistic choices by non-deterministic choices. Furthermore, let $\mathcal{E}(B_L)$ be the (ordinary) bundle event structure corresponding to B_L as defined in [10]. We now have the following theorem.

Theorem 4.8 *Correctness Theorem*

$$\text{For all } B \in \mathcal{L}\ : [\![B]\!] = (E, \#, \mapsto, l, \pi) \ \Rightarrow\ \mathcal{E}(B_L) = (E, \#, \mapsto, l) \quad .$$

Proof: Trivial, since removal of the parts concerning π in the definition of $[\![\ ]\!]$ leads to the bundle event semantics of LOTOS in [10], and the semantics of $[]_p$ is equal to the semantics of $[]$ when the part concerning π is removed. □

5. AN EXAMPLE

In this section we provide a simple example to illustrate how systems can be described in $\mathcal{L}$ and, more importantly, how the πBES semantics can be used as a starting-point for performing a stochastic analysis. The example is rather intuitive in the sense that no formal mapping is given here between the πBES and the analysis framework, i.e. Markov chains. Nevertheless, we think it is useful to show, even in a rather informal way, how the model can be used for stochastic analysis.

A few remarks considering the example are in order. First, we do not consider value passing as this is not included in $\mathcal{L}$. As we will see, this simplification will not hamper us. Moreover, we remark that for the sake of brevity trailing **stop**s are omitted from expressions. Furthermore, recursion is used so as to describe the iterative behaviour of processes. We stress that recursion is not (yet) included in our language [5]. However, the extension of $\mathcal{L}$ with tail recursion —the kind of recursion used in the sequel— is rather straightforward. Finally, we want to emphasize that although the example below suggests that our semantical model restricts us to the use of Markovian models for performance analysis this is definitely not the case. We could equally well provide examples for which another type of stochastic analysis is appropriate.

5.1. Semi-Markov chains

In this section we briefly consider the basics of discrete *semi-Markov processes*. A more elaborated presentation can be found in [6, 21].

In contrast to the traditional discrete Markov processes with geometrically distributed residence times, a semi-Markov chain (SMC) allows an *arbitrary* distribution of residence times. Needless to say, a Markov chain is thus also a semi-Markov chain. An SMC changes states in accordance with a Markov chain but takes a random amount of time between changes. Thus an SMC does *not* possess the Markovian property that given the present state the future is independent of the past: when predicting the future not only the current state is of importance, but also the amount of time already spent in that state

[5]A proposal for dealing with recursion can be found in [14].

plays a role. At the moments of transition the SMC behaves identical to an ordinary Markov chain, in fact, when only considering transition instants (i.e., abstracting from the residence times), we have an "embedded" Markov chain.

Let P_{ij} be the transition probability going from state S_i to S_j. Given that the next state is S_j, the number of time-units until the transition from S_i to S_j has distribution F_{ij}. Then, the probability $R_i(k)$ of being in state S_i for k time-units

$$R_i(k) = \sum_j P_{ij} * F_{ij}(k) \quad . \tag{1}$$

Let r_i denotes the mean of R_i. r_i is usually called the (average) residence time in state S_i. Now define T_i to be the average number of time-units between successive transitions into state S_i, and let ϕ_i denote the fraction of time the system is in S_i (on the long run), or, equivalently, the stationary probability of the SMC being in state S_i. Then we have

$$\phi_i = \frac{r_i}{T_i} \quad . \tag{2}$$

First observe the SMC abstracting from the residence times, and define ψ_i as the stationary probability of this system being in state S_i. Thus ψ_i denotes the (stationary) probability of the system being in state S_i at a transition instant. Stated otherwise, ψ_i is the *fraction of instants* at which the system is in state S_i, considering an infinite amount of transition instants. In order to obtain the *fraction of time* ϕ_i the system is in state S_i, the residence times must be taken into account. We have

$$\phi_i = \frac{\psi_i * r_i}{\sum_j \psi_j * r_j} \quad . \tag{3}$$

ψ_i corresponds to the stationary probabilities of the embedded Markov chain and can be calculated in the following way, provided the embedded Markov chain is ergodic:

$$\begin{aligned} \psi_i &= \textstyle\sum_j P_{ji} * \psi_j \\ \textstyle\sum_i \psi_i &= 1 \quad . \end{aligned} \tag{4}$$

The SMCs of our examples all have ergodic embedded Markov chains. We use the above equations to calculate ψ_i. Using (1) the average recurrence times are obtained and, subsequently they are used in order to obtain expressions for ϕ_i (using (3)).

5.2. A sample system

One of the main novelties of our model is the locality aspect—if one is interested in analysing only a part of the system it is relatively easy to do so without considering other (irrelevant) parts. To illustrate this we consider the following example.

$$Q = (b\,;d\,|[\,d\,]|\,c\,;d) \quad ,$$

$$R = (i\,;d\,[]_p\,i\,;a\,;d) \quad \text{, and}$$

$$P = s\,;(R\,|[\,d\,]|\,Q) \quad .$$

It describes the parallel composition of two processes, Q and R. R can autonomously choose *not* to perform action a and this choice has probability p; in any case it will then

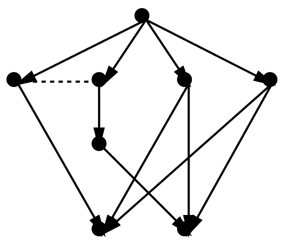

Figure 5. πbes of process P.

synchronize on d with Q, which first executes b and c in parallel, and then d. Figure 5 shows the corresponding πbes of their composite behaviour defined by P.

Now consider $X = P \,|[\, d \,]|\, d\, ; X$. The πbes corresponding to X is given in Figure 6(a). In fact, Figure 6(a) explicitly shows only the finite part of the πbes corresponding to "the body" of process X. Recursive calls of X must be considered as instantiations of the πbes shown in the figure replacing the dotted arrows in the obvious way. (In the sequel we shall often speak of "events" whereas —strictly speaking— we should rather speak of "instances" of those events since they belong to different instantiations of the finite part of the πbes shown in Figure 6(a).) Suppose we are interested in the generation of a

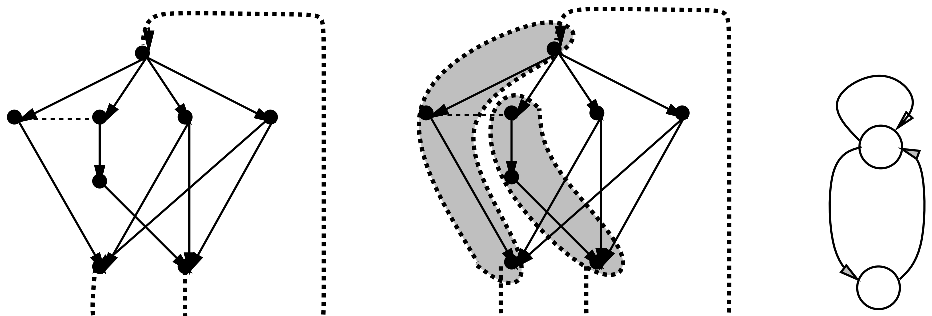

Figure 6. (a) πbes of X, (b) relevant "view" of the system, (c) Markov chain.

events, e.g. the *average delay* [6] between two a's. We can group relevant events into two

[6]Note that our model does not incorporate any timing information (yet). Therefore, we assume that each bundle lasts for one time-unit and that events are executed instantaneously. It is a subject of further

states (indicated by the two grey "islands" S_1 and S_2 in Figure 6(b)). Notice that we do not consider events labeled b and c. The way in which events are grouped imposes a particular "view" on the system which is characterized by abstracting from details that are irrelevant for the performance analysis one performs. The grouping we have chosen gives rise to the simple Markov chain depicted in Figure 6(c).

Let's further focus our attention on analysing the Markov chain. For simplicity reasons in this example the residence time in a certain state is taken to be the number of bundles in a state plus one. That is, the number of bundles spent in a state plus the one so as to leave that state. (This is equivalent to the number of events contained in that state.) Note that events themselves are considered to occur instantaneously, and moreover, that state transitions are also considered to be instantaneous.

The system stays in state S_1 for 3 time-units when by the next transition, it remains in that state, whereas it stays for a single time-unit in S_1 when moving to state S_2. Using (1) we obtain

$$r_1 = 1 + 2p \quad \text{, and } r_2 = 3 \quad .$$

For the (ergodic) Markov chain we have the following set of equations:

$$\begin{aligned} \psi_1 &= p\psi_1 + \psi_2 \\ \psi_2 &= (1-p)\psi_1 \quad . \end{aligned}$$

Using $\sum_i \psi_i = 1$ we obtain

$$\psi_1 = \frac{1}{2-p} \quad , \psi_2 = \frac{1-p}{2-p} \quad ,$$

as stationary probabilities of the embedded Markov chain, and for the semi-Markov chain we have, using (3),

$$\phi_1 = \frac{1+2p}{4-p} \quad , \phi_2 = \frac{3(1-p)}{4-p} \quad .$$

The average delay between two subsequent a-events is equal to the average number of time-units between successive transitions to state S_2. According to (2) this is equal to $\frac{r_2}{\phi_2}$, or $\frac{4-p}{1-p}$. For $p \to 0$ the average delay reaches 4 time-units, which is the optimal case (it lasts at least 4 bundles each time in between two subsequent a's). For $p \to 1$ a's are never generated—and the average delay reaches ∞.

In conclusion, we like to emphasize that the average delay between two subsequent a's is analysed without considering the —for our purposes— irrelevant part of process Q (more precisely, events b and c). This seems reasonable as only R is involved in generating a events. Here we claim that this 'locality' novelty is a direct consequence of the distinction between parallel composition and non-determinism in the πBES-model.

research to extend our model with timing information, leading to a timed πBES model.

6. CONCLUSIONS AND FUTURE WORK

This paper proposed a model for probabilistic bundle event structures, πBES. A language for the specification of probabilistic processes has been introduced and its πBES semantics has been given. It has been shown that the addition of probabilities to BES is "orthogonal" to the *BES* model itself in the sense that the original semantics of the non-probabilistic version of the language is easily recovered from the πBES semantics by simply removing the probabilities associated to some events. It is worth pointing out here that this is *not* the case when the interleaving semantics approach is followed [13, 12]. In that case, in fact, for any expression e of $\mathcal{L}$, the labelled transition system obtained by removing the probability values from the probabilistic transition system [4] of e and the labelled transition system of the LOTOS counterpart of e can only proved to be *testing equivalent*.

It is also argued that, πBES being a *true concurrency* model, it both reduces the *state explosion* problem typical for other models and allows for *local* analysis, thus facilitating stochastic analysis. This conjecture has been shown by means of a simple example. (A stop-and-wait protocol is described in [9].) Several extensions can be foreseen for πBES. First of all it is necessary to incorporate recursion. This amounts to a simple extension to the definition for the *BES* semantics of recursive processes, which associates the latter to infinite *BES*. In [14] a model for finitely representing a rather interesting class of such infinite *BES* is proposed. Finite representation of (infinite) πBES is essential for applying Markov theory to probabilistic processes. The formal mapping from such representations to Markov processes is to be defined. Other extensions to the language as well as to the model will allow probabilities to be associated to visible events too. A possibility would be to extend the πBES model towards a *reactive* model of probabilistic processes [24].

Finally, but equally or even more important, comes the extension of πBES with time parameters, both deterministic and probabilistic. In this context it may be interesting to compare (suitable extensions of) πBES with stochastic Petri nets [3]. Moreover, algebraic relations between processes could be defined based on probabilities (and time).

Acknowledgements: The third author wants to thank the University of Twente and in particular the TIOS Group for the collaboration and the nice environment they provided to him.

REFERENCES

1. T. Bolognesi and E. Brinksma. Introduction to the ISO specification language LOTOS. *Computer Networks and ISDN Systems*, 14:25–59, 1987.
2. I. Christoff. Testing equivalences and fully abstract models for probabilistic processes. In J.C.M. Baeten and J.W. Klop, editors, *Proceedings of CONCUR'90: Theories of Concurrency*, LNCS 458, pages 126–140. Springer-Verlag, 1990.
3. M. Marsan Ajmone et. al. An introduction to generalized stochastic Petri nets. *Microelectronics and Reliability*, 31(4):699–725, 1991.
4. A. Giacalone, C.-C. Jou, and S.A. Smolka. Algebraic reasoning for probabilistic concurrent systems. In *Proceedings of the Working Conference on Programming Concepts and Methods IFIP TC 2*. North-Holland, 1990.
5. H. Hansson and B. Jonsson. A calculus for communicating systems with time and probabilities. In *Proceedings of 11th IEEE Real-Time Systems Symposium*, pages 278–287. IEEE Computer Society Press, 1990.

6. D.P. Heyman and M.J. Sobel. *Stochastic Models in Operations Research*, volume 1 - Stochastic Processes and Operating Characteristics. McGraw-Hill, New York, 1982.
7. International Standards Organization ISO. Information processing systems - Open Systems Interconnection - LOTOS - A formal description technique based on the temporal ordering of observational behaviour, 1989. ISO 8807.
8. C.-C. Jou and S.A. Smolka. Equivalences, congruences and complete axiomatizations for probabilistic processes. In J.C.M. Baeten and J.W. Klop, editors, *Proceedings of CONCUR'90: Theories of Concurrency*, LNCS 458, pages 367–383. Springer-Verlag, 1990.
9. J.-P. Katoen, R. Langerak, and D. Latella. Modeling systems by probabilistic process algebra: An event structures approach. Technical Report 93-29, University of Twente, 1993.
10. R. Langerak. *Transformations and Semantics for LOTOS*. PhD thesis, University of Twente, 1992.
11. R. Langerak. Bundle event structures: a non-interleaving semantics for LOTOS. In M. Diaz and R. Groz, editors, *Formal Description Techniques V*, pages 331–346. North-Holland, 1993.
12. R. Langerak and D. Latella. A language of finite probabilistic processes and its interleaving semantics. Technical Report 93-24, University of Twente, 1992.
13. D. Latella. A simple calculus of finite probabilistic processes. Technical Report 93-08, University of Twente, 1992.
14. D. Latella. Recursive bundle event structures. Technical Report 93-27, University of Twente, 1993.
15. D. Latella and P. Quaglia. A proposal for a calculus of probabilistic processes. Technical Report C91-27, C.N.R. - Ist. CNUCE, 1991.
16. D. Latella and P. Quaglia. A calculus of probabilistic synchronizing processes and some applications. Technical Report C92-17, C.N.R. - Ist. CNUCE, 1992.
17. R. De Nicola. Extensional equivalences for transition systems. *Acta Informatica*, 24:211–237, 1987.
18. S. Purushothaman and P.A. Subrahmanyam. Reasoning about probabilistic behaviour in concurrent systems. *IEEE Transactions on Software Engineering*, SE-13(6), 1987.
19. P. Quaglia. Proposta per una variante probabilistica di LOTOS. Tesi di laurea in scienze dell'informazione - Università degli studi di Pisa, Università di Pisa, 1991.
20. N. Rico and G. v. Bochmann. Performance description and analysis for distributed systems using a variant of LOTOS. In B. Jonsson et. al., editor, *Protocol Specification, Testing, and Verification IX*, pages 199–213. North-Holland, 1991.
21. S.M. Ross. *Stochastic Processes*. John Wiley & Sons, New York, 1983.
22. S.A. Smolka and B. Steffen. Priority as extremal probability. In J.C.M. Baeten and J.W. Klop, editors, *Proceedings of CONCUR'90: Theories of Concurrency*, LNCS 458, pages 456–466. Springer-Verlag, 1990.
23. C.M.N. Tofts. A synchronous calculus of relative frequency. In J.C.M. Baeten and J.W. Klop, editors, *Proceedings of CONCUR'90: Theories of Concurrency*, LNCS 458, pages 467–480. Springer-Verlag, 1990.
24. R. van Glabbeek, S.A. Smolka, B. Steffen, and C.M.N. Tofts. Reactive, generative, and stratified models of probabilistic processes. In *Proceedings of 5th IEEE Symposium on Logic in Computer Science*, pages 130–141, 1990.