Extended Markovian Process Algebra *

Marco Bernardo and Roberto Gorrieri

Università di Bologna, Dipartimento di Scienze dell'Informazione
Mura Anteo Zamboni 7, 40127 Bologna, Italy
E-mail: {bernardo, gorrieri}@cs.unibo.it

Abstract. EMPA enhances the expressiveness of classical process algebras by integrating functional and performance descriptions of concurrent systems. This is achieved by offering, besides passive actions (useful for pure nondeterminism), actions whose duration is exponentially distributed as well as immediate actions (useful for performance abstraction), parametrized by priority levels (hence prioritized choices) and weights (hence probabilistic choices). In order to analyze an EMPA term, from its integrated semantic model (a transition system labeled on both action types and action durations) we derive a functional semantic model (a transition system labeled on action types only) and a performance semantic model (a Markov chain). We show that an integrated analysis, i.e. a notion of equivalence on the integrated semantic model, is not only convenient but also necessary to achieve compositionality.

1 Introduction

The need of integrating the performance modeling and analysis of a concurrent system into the design process of the system itself has been widely recognized (see, e.g., [20, 6]). Unfortunately, it often happens that a system is first fully designed and tested for functionality, and afterwards tested for efficiency. The major drawback is that, whenever the performance is detected to be poor, the system has to be designed again, thereby negatively affecting both the design costs and the delivery at a fixed deadline. Another relevant drawback is that tests for functionality and performance are carried out on two different models of the system, so one has to make sure that these two models are consistent, i.e. they really describe (different aspects of) the same system.

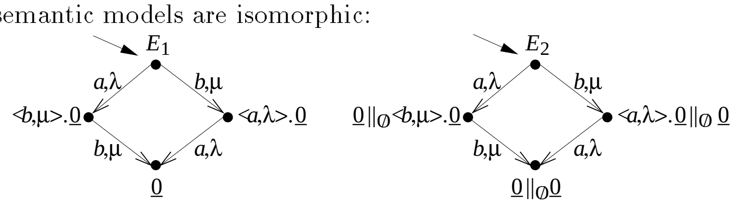
In the last two decades a remarkable effort has been made in order to make existing formal description techniques suitable to support performance modeling and analysis by introducing the concept of time. One of the most mature fields where functional and performance aspects of concurrent systems are both considered is that of generalized stochastic Petri nets (GSPNs) [1], where a zero or exponentially distributed firing delay is associated with each net transition. Functional and performance analyses are carried out separately on two different projected models (a classical Petri net and a Markov chain) obtained from the same integrated model (the GSPN), so we are guaranteed that the projected

* This research has been partially funded by MURST, CNR and LOMAPS n. 8130.

models are consistent. However two problems have to be addressed: (*i*) lack of compositionality, and (*ii*) inability to perform an analysis directly on the integrated model, which can be much more efficient as there is no need of building projected models.

In order to solve the two problems above, in this paper we propose the adoption of a stochastic process algebra that extends the expressiveness of classical process algebras by representing each action as a pair composed of a type and a duration. Such a stochastic process algebra (*i*) naturally supplies compositionality, and (*ii*) allows functional and performance analyses to be carried out both on two consistent projected semantic models (a transition system labeled only on the type of the actions, and a Markov chain) and directly on the integrated semantic model (a transition system labeled on both the type and the duration of the actions) thanks to a suitable integrated equivalence.

The stochastic process algebra is called *Extended Markovian Process Algebra (EMPA)* [3]. Its name stems from the fact that action durations are mainly expressed by means of exponentially distributed random variables (hence Markovian), but it is also possible to express actions having duration zero as well as actions whose duration is unspecified (hence Extended). The restriction to exponentially distributed durations simplifies the performance evaluation, as the performance models are Markov chains. Also, such a restriction affects the semantic treatment, because the memoryless property of the exponential distribution allows us to define an integrated semantics for EMPA through the interleaving approach. For instance, given an action a whose duration is exponentially distributed with rate λ, and an action b whose duration is exponentially distributed with rate μ, consider term $E_1 \equiv <a, \lambda>.<b, \mu>.\underline{0} + <b, \mu>.<a, \lambda>.\underline{0}$ that executes either a followed by b or b followed by a, and term $E_2 \equiv <a, \lambda>.\underline{0} \,\|_\emptyset <b, \mu>.\underline{0}$ that executes a in parallel with b. Their integrated semantic models are isomorphic:



This is correct from the performance point of view due to the memoryless property of the exponential distribution: if E_2 completes a before b, then the residual time to the completion of b is still exponentially distributed with rate μ, so the rate labeling the transition from state $\underline{0} \,\|_\emptyset <b, \mu>.\underline{0}$ to state $\underline{0} \,\|_\emptyset \underline{0}$ is μ itself instead of μ conditional on λ.

The paper is organized as follows. In Sect. 2 we introduce EMPA by giving the syntax of terms and the meaning of operators. In Sect. 3 we define the integrated operational interleaving semantics, the functional semantics and the performance semantics. In Sect. 4 we introduce a notion of integrated equivalence in the bisimulation style and we show that it is a congruence: this will prove that the analysis carried out on the integrated semantic model is not only convenient but even necessary for compositionality purposes. In Sect. 5 we present an example showing the expressiveness of EMPA. Finally, in Sect. 6 we report some concluding remarks. [2]

2 Syntax and Informal Semantics

2.1 Actions: Types and Rates

Each action is a pair "$<a, \tilde{\lambda}>$" consisting of the *type* of the action and the *rate* of the action. The rate indicates the speed at which the action occurs from the point of view of an external observer: the rate is used as a concise way to denote the random variable specifying the duration of the action. Depending on the type, like in classical process algebras, actions are divided into *external* and *internal*: as usual, we denote by τ the only internal action type we use. Depending on the rate, actions are divided into:

- *Active actions*, i.e. actions whose rate is fixed, in turn divided into:
  - *Exponentially timed actions*, i.e. actions whose rate λ is a positive real number. Such a number is interpreted as the parameter of the exponential distribution specifying the duration of the action.
  - *Immediate actions*, i.e. actions whose rate $\infty_{l,w}$ is infinite. Such actions have duration zero, and each of them is given a priority level $l \in \mathbb{N}_+$ and a weight $w \in \mathbb{R}_+$.
- *Passive actions*, i.e. actions whose rate (denoted by $*$) is undefined. The duration of a passive action is fixed only by synchronizing it with an active action of the same type.

The classification of actions based on their rates implies that: (i) exponentially timed actions model activities that are relevant from the performance point of view, (ii) immediate actions model logical events as well as activities that are either irrelevant from the performance point of view or unboundedly faster than the others, (iii) passive actions model activities waiting for the synchronization with timed activities.

We denote the set of actions by $Act = AType \times ARate$ where $AType$ is the set of types and $ARate = \mathbb{R}_+ \cup Inf \cup \{*\}$, with $Inf = \{\infty_{l,w} \mid l \in \mathbb{N}_+ \wedge w \in \mathbb{R}_+\}$, is the set of rates. We use $a, b, c, \ldots$ as metavariables for $AType$, $\tilde{\lambda}, \tilde{\mu}, \tilde{\gamma}, \ldots$ for $ARate$, and $\lambda, \mu, \gamma, \ldots$ for $\mathbb{R}_+$.

2.2 Syntax of Terms and Informal Semantics of Operators

Let $Const$ be a set of *constants*, ranged over by $A, B, C, \ldots$, and let $Relab = \{\varphi : AType \longrightarrow AType \mid \varphi(\tau) = \tau \wedge \varphi(AType - \{\tau\}) \subseteq AType - \{\tau\}\}$ be a set of *relabeling functions*.

[2] Due to lack of space, the reader is referred to [3] for an extensive presentation of EMPA as well as proofs of results.

Definition 1. The set $\mathcal{L}$ of *process terms* is generated by the following syntax

$$E ::= \underline{0} \mid <a, \tilde{\lambda}>.E \mid E/L \mid E\backslash H \mid E[\varphi] \mid E + E \mid E \parallel_S E \mid A$$

where $L, S \subseteq AType - \{\tau\}$ and $H \subseteq AType$. The set $\mathcal{L}$ will be ranged over by $E, F, G, \ldots$. We denote by $\mathcal{G}$ the set of closed and guarded terms of $\mathcal{L}$. ■

In the rest of the section we informally explain the semantics of the operators: the formal semantics will be presented in Sect. 3. Since the *null term* "$\underline{0}$", the *prefix operator* "$<a, \tilde{\lambda}>.\_$", the *functional abstraction operator* "$\_/L$" (the hiding operator of CSP [11]), the *functional relabeling operator* "$\_[\varphi]$", and the *alternative composition operator* "$\_+\_$" are typical of classical process algebras, we concentrate on the two remaining operators.

The *temporal restriction operator* "$\_\backslash H$" prevents the execution of passive actions whose type is in H. Based on this operator, we define the notion of *temporal closure*: a term is temporally closed if it cannot execute passive actions. Thus, the temporal closure property singles out terms that are completely specified from the performance viewpoint.

The *parallel composition operator* "$\_\parallel_S \_$" is based on two synchronization disciplines. The synchronization discipline on action types is the same as that of CSP [11]. The synchronization discipline on action rates states that action $<a, \tilde{\lambda}>$ can be synchronized with action $<a, \tilde{\mu}>$ if and only if $\min(\tilde{\lambda}, \tilde{\mu}) = *$, [3] and the rate of the resulting action is given by $\max(\tilde{\lambda}, \tilde{\mu})$ up to normalization (see Sect. 3). In other words, in a synchronization at most one active action can be involved and its rate determines the rate of the resulting action, up to normalization. This choice leads to an intuitive treatment of the synchronization discipline on action rates, as it is based on the client-server paradigm.

2.3 Race Policy

Since several active actions may be simultaneously executable, we need a mechanism for choosing the action to be executed. Like in [1, 8, 10], we adopt the *race policy*: the action sampling the least duration succeeds.

If we consider a state enabling only exponentially timed actions, the race policy establishes that (i) the random variable describing the *sojorn time* in that state is the minimum of the exponentially distributed random variables describing the duration of the enabled actions, and (ii) the *execution probability* of each enabled action is determined as well by the exponentially distributed random variables describing the duration of the enabled actions. In order to compute the two quantities above, we exploit the property that the minimum of n independent exponentially distributed random variables is an exponentially distributed random variable whose rate is the sum of the n original rates. As a consequence, if n exponentially timed actions $<a_1, \lambda_1>, <a_2, \lambda_2>, \ldots, <a_n, \lambda_n>$ are enabled, then the sojorn time is exponentially distributed with rate $\lambda = \sum_{i=1}^{n} \lambda_i$ and the execution probability of $<a_k, \lambda_k>$, $1 \leq k \leq n$, is given by λ_k/λ.

If we consider instead a state enabling both exponentially timed and immediate actions, the race policy establishes that (i) the exponentially timed actions

[3] We assume that $* < \lambda < \infty_{l,w}$ for all $\lambda \in \mathbb{R}_+$ and $\infty_{l,w} \in Inf$.

cannot be executed at all because they cannot sample zero durations, and (*ii*) the sojorn time in that state is zero. Only the enabled immediate actions having the highest priority level are actually executable, and the choice among them is probabilistically made by giving each of them an execution probability proportional to its weight. As a consequence, if n immediate actions $<a_1, \infty_{l,w_1}>$, $<a_2, \infty_{l,w_2}>$, ..., $<a_n, \infty_{l,w_n}>$ are enabled and no immediate action with higher priority level is enabled, then the execution probability of $<a_k, \infty_{l,w_k}>$, $1 \leq k \leq n$, is given by w_k/w where $w = \sum_{i=1}^{n} w_i$. [4]

Concerning the alternative composition operator, we realize that in its simpler form it expresses a choice between two active actions whose nature is:

- *Prioritized* if the two active actions have different priority levels. The choice is solved *implicitly* if it concerns an exponentially timed action and an immediate action (because the choice is implicitly determined by the race policy), *explicitly* if it concerns two immediate actions having different priority levels (because the priority levels explicitly determine the choice).
- *Probabilistic* if the two active actions have the same priority level. The choice is solved *implicitly* if it concerns two exponentially timed actions (because their execution probabilities are implicitly determined by their durations due to the race policy), *explicitly* if it concerns two immediate actions having the same priority level (because their execution probabilities are explicitly determined by their weights).

Now we turn our attention to passive actions, which cannot be undergone to the race policy since their duration is unspecified. Passive actions that are not synchronized with active actions of the same type result in models that are not temporally closed. Nevertheless, these models could be useful: since passive actions can be safely viewed as actions of a classical process algebra, a choice between two passive actions has a *nondeterministic* nature.
If we consider instead passive actions that do engage in synchronizations, then their execution probability is assigned a value. Whenever n passive actions can be separately synchronized with the same active action, we assume that each of the synchronizations is assigned the same execution probability, given by the execution probability of the active action divided by n. For example, terms $<a, \lambda>.\underline{0} \,\|_{\{a\}} (<a, *>.\underline{0} + <a, *>.\underline{0})$ and $<a, \lambda>.\underline{0} \,\|_{\{a\}} (<a, *>.\underline{0} \,\|_{\emptyset} <a, *>.\underline{0})$ comprise two passive actions that can be separately synchronized with the same active action, and each of the two synchronizations is given probability 1/2.

3 Integrated, Functional and Performance Semantics

The main problem to tackle when defining the semantics for EMPA is that the actions executable by a given term may have different priority levels, and only those having the highest priority level are actually executable. Let us call *potential move* of a given term a pair composed of (*i*) an action executable by the

[4] w and λ above are called the *exit rate* of the corresponding state.

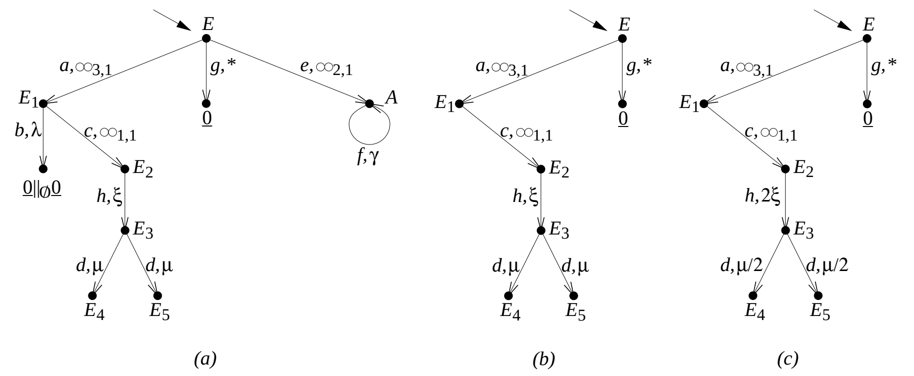


Fig. 1. Integrated interleaving model of term E

term, and (*ii*) a derivative term obtained by executing that action. To solve the problem above, we compute inductively all the potential moves of a given term regardless of priority levels, and then we select those having the highest priority level. This is motivated, in a stochastic framework, by the fact that the actual executability as well as the execution probability of an action depend upon all the actions that are executable at the same time when it is executable: only if we know all the potential moves of a given term, we can correctly determine its transitions and their rates. This is clarified by the following example.

Example 1. Consider term $E \equiv <a, \infty_{3,1}>.E_1 + <g, *>.\underline{0} + <e, \infty_{2,1}>.A$ where $E_1 \equiv <b, \lambda>.(\underline{0} \,||_{\emptyset}\, \underline{0}) + <c, \infty_{1,1}>.E_2$, $E_2 \equiv <h, \xi>.E_3 + <h, \xi>.E_3$, $E_3 \equiv <d, \mu>.\underline{0} \,||_{\{d\}} (<d, *>.\underline{0} \,||_{\emptyset} <d, *>.\underline{0})$, $A \stackrel{\Delta}{=} <f, \gamma>.A$. Suppose we apply to E standard rules for classical process algebras, thereby disregarding priority levels. Then we obtain the labeled transition system (LTS) in Fig. 1(*a*) where $E_4 \equiv \underline{0} \,||_{\{d\}} (\underline{0} \,||_{\emptyset} <d, *>.\underline{0})$ and $E_5 \equiv \underline{0} \,||_{\{d\}} (<d, *>.\underline{0} \,||_{\emptyset}\, \underline{0})$.

Now assume that priority levels are taken into account. Then lower priority transitions must be pruned, thus resulting in the LTS in Fig. 1(*b*): note that the passive transition has not been discarded. This is obtained by means of an auxiliary function called *Select*.

Finally, consider the rate of the transition from E_2 to E_3 and the rate of the two transitions from E_3 to E_4 and E_5. In the correct semantic model for E, such rates have to be like in Fig. 1(*c*). The rate of the transition from E_2 to E_3 is $2 \cdot \xi$ instead of ξ because in E_2 two exponentially timed actions with rate ξ occur and the race policy has been adopted. To achieve this, we keep track of the multiplicity [5] of potential moves, and then we construct transitions by using an auxiliary function called *Melt* that merges together those potential moves having

[5] We use "$\{\!|$" and "$|\!\}$" as brackets for multisets, "$\_ \oplus \_$" to denote multiset union, $\mathcal{M}u_{fin}(S)$ ($\mathcal{P}_{fin}(S)$) to denote the collection of finite multisets (sets) over set S, and $\pi_i(M)$ to denote the multiset obtained by projecting the tuples in multiset M on their i-th component.

the same action type, the same priority level and the same derivative term. The rate of transitions derived from the merging of potential moves is computed by means of another auxiliary function called *Min*. The rate of the two transitions from E_3 to E_4 and E_5 is $\mu/2$ instead of μ because in E_3 only one exponentially timed action with rate μ occurs: the value $\mu/2$ stems from the assumption about the execution probability of passive actions made in Sect. 2.3. A normalization of rates is required, and this is carried out by means of an auxiliary function called *Norm*. ■

Now let us turn our attention to the definition of the integrated operational interleaving semantics for EMPA. The transition relation $\longrightarrow$ is the least subset of $\mathcal{G} \times Act \times \mathcal{G}$ satisfying the inference rule reported in the first part of Table 1. This rule selects the potential moves having the highest priority level, and then merges together the remaining potential moves having the same action type, the same priority level and the same derivative term. The first operation is carried out through functions $Select : \mathcal{M}u_{fin}(Act \times \mathcal{G}) \longrightarrow \mathcal{M}u_{fin}(Act \times \mathcal{G})$ and $PL : Act \longrightarrow PLSet$, with $PLSet = \{-1\} \cup \mathbb{N}$, which are defined in the third part of Table 1. The second operation is carried out through function $Melt : \mathcal{M}u_{fin}(Act \times \mathcal{G}) \longrightarrow \mathcal{P}_{fin}(Act \times \mathcal{G})$ and partial function $Min : (ARate \times ARate) \multimap ARate$, which are defined in the fourth part of Table 1.

The multiset $PM(E)$ of potential moves of $E \in \mathcal{G}$ is defined by structural induction as the least element of $\mathcal{M}u_{fin}(Act \times \mathcal{G})$ satisfying the rules reported in the second part of Table 1. The normalization of the rates of potential moves resulting from the synchronization of the same active action with several independent or alternative passive actions is carried out through partial function $Norm : (AType \times ARate \times ARate \times \mathcal{M}u_{fin}(Act \times \mathcal{G}) \times \mathcal{M}u_{fin}(Act \times \mathcal{G})) \multimap ARate$ and function $Split : (ARate \times \mathbb{R}_{]0,1]}) \longrightarrow ARate$, which are defined in the fifth part of Table 1. Note that $Norm(a, \tilde{\lambda}, \tilde{\mu}, PM_1, PM_2)$ is defined if and only if $\min(\tilde{\lambda}, \tilde{\mu}) = *$, which is the condition on action rates we have required in Sect. 2.2 in order for a synchronization to be permitted.

Definition 2. The *integrated operational interleaving semantics* of $E \in \mathcal{G}$ is the LTS $\mathcal{I}[\![E]\!] = (\uparrow E, Act, \longrightarrow_E, E)$ where $\uparrow E$ is the set of states reachable from E, and $\longrightarrow_E$ is $\longrightarrow$ restricted to $\uparrow E \times Act \times \uparrow E$. ■

Definition 3. $E \in \mathcal{G}$ is *temporally closed* if and only if $\mathcal{I}[\![E]\!]$ is isomorphic to $\mathcal{I}[\![E \backslash AType]\!]$. We denote by $\mathcal{E}$ the set of temporally closed terms of $\mathcal{G}$. ■

Given a term $E \in \mathcal{G}$, its integrated operational interleaving semantics $\mathcal{I}[\![E]\!]$ fully represents the behavior of E because transitions are decorated by both the action type and the action rate. One can think of obtaining the *functional semantics* $\mathcal{F}[\![E]\!]$ and the *performance semantics* $\mathcal{P}[\![E]\!]$ of E from $\mathcal{I}[\![E]\!]$ by simply dropping action rates and action types, respectively. As a matter of fact, this is the case for the functional semantics.

Definition 4. The *functional semantics* of $E \in \mathcal{G}$ is the LTS $\mathcal{F}[\![E]\!] = (\uparrow E, AType, \longrightarrow_{E,\mathcal{F}}, E)$ where $\longrightarrow_{E,\mathcal{F}}$ is $\longrightarrow_E$ restricted to $\uparrow E \times AType \times \uparrow E$. ■

$$\frac{(<a,\tilde{\lambda}>, E') \in \mathit{Melt}(\mathit{Select}(PM(E)))}{E \xrightarrow{a,\tilde{\lambda}} E'}$$

$$PM(\underline{0}) = \emptyset$$

$$PM(<a,\tilde{\lambda}>.E) = \{\!|\, (<a,\tilde{\lambda}>, E) \,|\!\}$$

$$\begin{aligned} PM(E/L) = {} & \{\!|\, (<a,\tilde{\lambda}>, E'/L) \mid (<a,\tilde{\lambda}>, E') \in PM(E) \wedge a \notin L \,|\!\} \oplus \\ & \{\!|\, (<\tau,\tilde{\lambda}>, E'/L) \mid (<a,\tilde{\lambda}>, E') \in PM(E) \wedge a \in L \,|\!\} \end{aligned}$$

$$PM(E \backslash H) = \{\!|\, (<a,\tilde{\lambda}>, E' \backslash H) \mid (<a,\tilde{\lambda}>, E') \in PM(E) \wedge \neg(a \in H \wedge \tilde{\lambda} = *) \,|\!\}$$

$$PM(E[\varphi]) = \{\!|\, (<\varphi(a),\tilde{\lambda}>, E'[\varphi]) \mid (<a,\tilde{\lambda}>, E') \in PM(E) \,|\!\}$$

$$PM(E_1 + E_2) = PM(E_1) \oplus PM(E_2)$$

$$\begin{aligned} PM(E_1 \,\|_S\, E_2) = {} & \{\!|\, (<a,\tilde{\lambda}>, E_1' \,\|_S\, E_2) \mid a \notin S \wedge (<a,\tilde{\lambda}>, E_1') \in PM(E_1) \,|\!\} \oplus \\ & \{\!|\, (<a,\tilde{\lambda}>, E_1 \,\|_S\, E_2') \mid a \notin S \wedge (<a,\tilde{\lambda}>, E_2') \in PM(E_2) \,|\!\} \oplus \\ & \{\!|\, (<a,\tilde{\gamma}>, E_1' \,\|_S\, E_2') \mid a \in S \wedge \\ & \qquad (<a,\tilde{\lambda}>, E_1') \in PM(E_1) \wedge \\ & \qquad (<a,\tilde{\mu}>, E_2') \in PM(E_2) \wedge \\ & \qquad \tilde{\gamma} = \mathit{Norm}(a,\tilde{\lambda},\tilde{\mu},PM(E_1),PM(E_2)) \,|\!\} \end{aligned}$$

$$PM(A) = PM(E) \qquad \text{if } A \stackrel{\Delta}{=} E$$

$$\begin{aligned} \mathit{Select}(PM) = \{\!|\, (<a,\tilde{\lambda}>, E) \in PM \mid {} & PL(<a,\tilde{\lambda}>) = -1 \vee \\ & \forall (<b,\tilde{\mu}>, E') \in PM.\, PL(<a,\tilde{\lambda}>) \geq PL(<b,\tilde{\mu}>) \,|\!\} \end{aligned}$$

$$PL(<a,*>) = -1 \qquad PL(<a,\lambda>) = 0 \qquad PL(<a,\infty_{l,w}>) = l$$

$$\begin{aligned} \mathit{Melt}(PM) = \{ (<a,\tilde{\lambda}>, E) \mid {} & (<a,\tilde{\mu}>, E) \in PM \wedge \\ & \tilde{\lambda} = \mathit{Min}\{\!|\, \tilde{\gamma} \mid (<a,\tilde{\gamma}>, E) \in PM \wedge PL(<a,\tilde{\gamma}>) = PL(<a,\tilde{\mu}>) \,|\!\}\} \end{aligned}$$

$$* \,\mathit{Min}\, * = * \qquad \lambda \,\mathit{Min}\, \lambda' = \lambda + \lambda' \qquad \infty_{l,w} \,\mathit{Min}\, \infty_{l,w'} = \infty_{l,w+w'}$$

$$\mathit{Norm}(a,\tilde{\lambda},\tilde{\mu},PM_1,PM_2) = \begin{cases} \mathit{Split}(\tilde{\lambda}, 1/(\pi_1(PM_2))(<a,*>)) & \text{if } \tilde{\mu} = * \\ \mathit{Split}(\tilde{\mu}, 1/(\pi_1(PM_1))(<a,*>)) & \text{if } \tilde{\lambda} = * \end{cases}$$

$$\mathit{Split}(*,\alpha) = * \qquad \mathit{Split}(\lambda,\alpha) = \lambda \cdot \alpha \qquad \mathit{Split}(\infty_{l,w},\alpha) = \infty_{l,w\cdot\alpha}$$

Table 1. Inductive rules for EMPA integrated interleaving semantics

The definition of the performance semantics requires instead a more careful treatment. Since in EMPA the durations of the actions are mainly expressed by exponentially distributed random variables, it is natural to associate with each term a homogeneous continuous-time Markov chain (HCTMC) [13] acting as a performance model. Since in a HCTMC neither passive transitions nor immediate transitions are allowed, we restrict ourselves to temporally closed terms and, given $E \in \mathcal{E}$, its performance semantics $\mathcal{P}[\![E]\!]$, hereafter called *Markovian semantics* and denoted by $\mathcal{M}[\![E]\!]$, is obtained from $\mathcal{I}[\![E]\!]$ by discarding action types and immediate transitions. Formally, $\mathcal{M}[\![E]\!]$ is represented as a variant of a LTS, called probabilistically rooted labeled transition system (PLTS), in which there is no initial state but a probability mass function that determines for each state the probability that it is the initial state. We report below the definition of PLTS together with a notion of equivalence in the style of [14] that will be used in the following.

Definition 5. A *probabilistically rooted labeled transition system (PLTS)* is a quadruple $(S, U, \longrightarrow, P)$ such that $S, U, \longrightarrow$ are defined as for a LTS, and the *initial state probability function* $P : S \longrightarrow \mathbb{R}_{[0,1]}$ satisfies $\sum_{s \in S} P(s) = 1$. ■

Definition 6. Let $Z_i = (S_i, \mathbb{R}_+ \cup Inf, \longrightarrow_i, P_i)$, $i \in \{1, 2\}$, be two PLTSs. We say that Z_1 is *p-bisimilar* to Z_2 if and only if there exists an equivalence relation $\mathcal{B} \subseteq (S_1 \cup S_2) \times (S_1 \cup S_2)$ such that:

- for each $C \in (S_1 \cup S_2)/\mathcal{B}$ it turns out $\sum_{s \in C \cap S_1} P_1(s) = \sum_{s \in C \cap S_2} P_2(s)$;
- whenever $(s_1, s_2) \in \mathcal{B} \cap (S_1 \times S_2)$, then for each $C \in (S_1 \cup S_2)/\mathcal{B}$ it turns out $Min\{\!|\, \tilde{\lambda} \mid s_1 \xrightarrow{\tilde{\lambda}}_1 s'_1 \wedge s'_1 \in C \cap S_1 \,|\!\} = Min\{\!|\, \tilde{\lambda} \mid s_2 \xrightarrow{\tilde{\lambda}}_2 s'_2 \wedge s'_2 \in C \cap S_2 \,|\!\}$. ■

The algorithm to transform $\mathcal{I}[\![E]\!]$ into $\mathcal{M}[\![E]\!]$ is organized in two phases.

1. The first phase drops action types and eliminates all the immediate transitions occurring in $\mathcal{I}[\![E]\!]$. The removal of the immediate transitions and the related states is justified from a stochastic point of view by the fact that the sojorn time in these states is zero.
2. The second phase aggregates states in order to reduce the HCTMC obtained at the end of the previous phase. The aggregation is based on the notion of ordinary lumping [12]: the state space is partitioned in such a way that the rates labeling the transitions from any two states lying in the same class to any class of the partition sum up to the same value.

Due to lack of space, we only show an application of such an algorithm.

Example 2. Consider $E \equiv {<}a, \lambda{>}.E_1$ where $E_1 \equiv {<}b, \infty_{1,2}{>}.A + {<}c, \infty_{1,1}{>}.B$, $A \stackrel{\Delta}{=} {<}e, \mu{>}.B$, $B \stackrel{\Delta}{=} {<}f, \mu{>}.A$. The LTS $\mathcal{I}[\![E]\!]$ is reported in Fig. 2(a). To obtain the Markovian semantics of E, we have to eliminate state E_1 together with its immediate transitions. This is carried out by splitting the exponentially timed transition from E to E_1 into two exponentially timed transitions from E to A

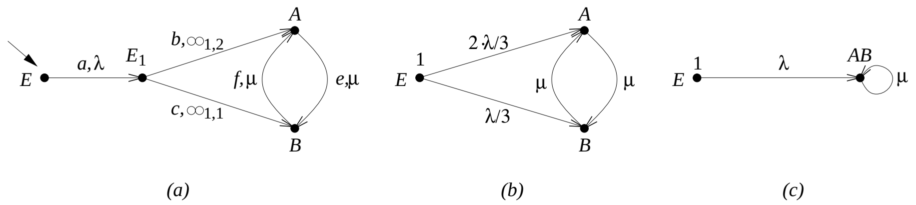


Fig. 2. Markovian model of term E

and B whose rates are $2 \cdot \lambda/3$ and $\lambda/3$, respectively: factors $2/3$ and $1/3$ are the execution probabilities of the two immediate transitions. The resulting PLTS is reported in Fig. 2(b).

Afterwards, we discover that states A and B have the same rate to each of the other states, so they can be ordinarily lumped together. The resulting PLTS is reported in Fig. 2(c). Note that the two exponentially timed transitions from E to A and B have been merged into a single transition from E to AB whose rate is the sum of the rates of the two transitions, while the transition from A to B and the transition from B to A have just been merged into a single one from AB to itself. ■

4 Extended Markovian Bisimulation Equivalence

Following the bisimulation style, it is straightforward to define two projected notions of equivalence based on the two projected semantic models.

Definition 7. Let $E_1, E_2 \in \mathcal{G}$. We say that E_1 is *functionally equivalent* to E_2, written $E_1 \sim_F E_2$, if and only if $\mathcal{F}[\![E_1]\!]$ is bisimilar to $\mathcal{F}[\![E_2]\!]$. ■

Definition 8. Let $E_1, E_2 \in \mathcal{E}$. We say that E_1 is *performance equivalent* to E_2, written $E_1 \sim_P E_2$, if and only if $\mathcal{M}[\![E_1]\!]$ is p-bisimilar to $\mathcal{M}[\![E_2]\!]$. ■

A natural candidate notion of equivalence for EMPA may be $\sim_{FP} = \sim_F \cap \sim_P$. However, the examples below show that $\sim_{FP}$ is not a congruence.

Example 3. Consider terms $E_1 \equiv \,<\!a, \lambda\!>.\underline{0} + <\!b, \mu\!>.\underline{0}$ and $E_2 \equiv \,<\!a, \mu\!>.\underline{0} + <\!b, \lambda\!>.\underline{0}$. Then $E_1 \sim_{FP} E_2$ but $E_1 \,\|_{\{b\}}\, \underline{0} \not\sim_P E_2 \,\|_{\{b\}}\, \underline{0}$ because the left-hand side term has exit rate λ while the right-hand side term has exit rate μ. Note that the action with type a of E_1 has execution probability $\lambda/(\lambda + \mu)$, while the action with type a of E_2 has execution probability $\mu/(\lambda + \mu)$, and this is not detected by $\sim_{FP}$. ■

Example 4. Consider terms $E_1 \equiv \,<\!a, \infty_{1,1}\!>.\underline{0}$ and $E_2 \equiv \,<\!a, \infty_{2,1}\!>.\underline{0}$. Then $E_1 \sim_{FP} E_2$ but $E_1 + <\!b, \infty_{1,1}\!>.\underline{0} \not\sim_F E_2 + <\!b, \infty_{1,1}\!>.\underline{0}$ because the left-hand side term can execute an action with type b while the right-hand side term cannot. ■

Example 5. Consider terms $E_1 \equiv \boldsymbol{<a, \infty_{1,1}>}.\underline{0}$ and $E_2 \equiv \boldsymbol{<a, \infty_{1,2}>}.\underline{0}$. Then $E_1 \sim_{FP} E_2$ but $E_1 + \boldsymbol{<b, \infty_{1,1}>}.\boldsymbol{<b, \lambda>}.\underline{0} \not\sim_P E_2 + \boldsymbol{<b, \infty_{1,1}>}.\boldsymbol{<b, \lambda>}.\underline{0}$ because state $\boldsymbol{<b, \lambda>}.\underline{0}$ has initial state probability 1/2 in the Markovian semantics of the left-hand side term, 1/3 in the Markovian semantics of the right-hand side term. ■

The examples above show that $\sim_{FP}$ is unable to keep track of the link between the functional part and the performance part of the actions. This means that to achieve compositionality, it is *necessary* to define an equivalence based on the integrated semantic model. Incidentally, this is even *convenient* with respect to $\sim_{FP}$, since it avoids the need of building the two projected semantic models and checking them for bisimilarity and p-bisimilarity, respectively.

In order to define an integrated equivalence $\sim_I$ for EMPA in the bisimulation style, we can follow the guideline below:

- Active actions should be treated by following the notion of *probabilistic bisimulation* [14], which consists of requiring a bisimulation to be an equivalence relation such that two bisimilar terms have the same aggregated probability to reach the same equivalence class by executing actions of the same type and priority level.
  - For exponentially timed actions, the notion of probabilistic bisimulation must be refined by requiring additionally that two bisimilar terms have identically distributed sojorn times. For example, if we consider terms $E_1 \equiv \boldsymbol{<a, \lambda>}.F + \boldsymbol{<a, \mu>}.G$ and $E_2 \equiv \boldsymbol{<a, 2 \cdot \lambda>}.F + \boldsymbol{<a, 2 \cdot \mu>}.G$, then both transitions labeled with a, λ and $a, 2 \cdot \lambda$ have execution probability $\lambda/(\lambda + \mu)$, and both transitions labeled with a, μ and $a, 2 \cdot \mu$ have execution probability $\mu/(\lambda + \mu)$, but the average sojorn time of E_1 is twice the average sojorn time of E_2. Due to the race policy, requiring that two bisimilar terms have identically distributed sojorn times and the same aggregated probability to reach the same equivalence class by executing exponentially timed actions of the same type amounts to requiring that two bisimilar terms have the same aggregated rate to reach the same equivalence class by executing exponentially timed actions of the same type. For example, it must hold that
  $$\boldsymbol{<a, \lambda>}.F + \boldsymbol{<a, \mu>}.F \sim_I \boldsymbol{<a, \lambda + \mu>}.F$$
  This coincides with the notion of *Markovian bisimulation* [9, 10, 4].
  - For immediate actions, the notion of probabilistic bisimulation must be restated in terms of weights. As a consequence, two bisimilar terms are required to have the same aggregated weight to reach the same equivalence class by executing immediate actions of the same type and priority level. For example, it must hold that
  $$\boldsymbol{<a, \infty_{l,w_1}>}.F + \boldsymbol{<a, \infty_{l,w_2}>}.F \sim_I \boldsymbol{<a, \infty_{l,w_1+w_2}>}.F$$
  This coincides with the notion of *direct bisimulation* [19].
- Passive actions should be treated by following the classical notion of bisimulation [16]. Thus, bisimilar terms are required to have the same passive actions reaching the same equivalence class, regardless of the actual number

of these passive actions. For example, it must hold that

$$<a,*>.F + <a,*>.F \sim_I <a,*>.F$$

Concerning priority levels, it might seem useful to be able to write equations like

$$<a,\lambda>.E + <b,\infty_{l,w}>.F \sim_I <b,\infty_{l,w}>.F$$

$$<c,\infty_{l,w}>.E + <d,\infty_{l',w'}>.F \sim_I <d,\infty_{l',w'}>.F \quad \text{if } l' > l$$

The problem is that the applicability of such equations depends on the context: e.g., terms $(<a,\lambda>.E + <b,\infty_{l,w}>.F) \|_{\{b\}} \underline{0}$ and $(<b,\infty_{l,w}>.F) \|_{\{b\}} \underline{0}$ are not equivalent at all. To solve the problem, we follow the proposal of [2] by introducing a *priority interpretation operator* Θ, and considering the language $\mathcal{L}_\Theta$ generated by the following syntax

$$E ::= \underline{0} \mid <a,\tilde{\lambda}>.E \mid E/L \mid E\backslash H \mid E[\varphi] \mid \Theta(E) \mid E + E \mid E \|_S E \mid A$$

whose semantic rules are those reported in Table 1 except that the rule in the first part is replaced by

$$\frac{(<a,\tilde{\lambda}>, E') \in Melt(PM(E))}{E \xrightarrow{a,\tilde{\lambda}} E'}$$

and the following rule for Θ is introduced in the second part

$$PM(\Theta(E)) = Select(PM(E))$$

It is easily seen that EMPA coincides with the set of terms $\{\Theta(E) \mid E \in \mathcal{L}\}$.

To keep the definition of integrated equivalence as simple as possible, we formalize the key concept of *conditional exit rate* in a uniform way for all kinds of action through partial function $ERate : (\mathcal{G}_\Theta \times AType \times PLSet \times \mathcal{P}(\mathcal{G}_\Theta)) \multimap ARate$ defined by $ERate(E,a,l,C) = Min\{\!| \tilde{\lambda} \mid E \xrightarrow{a,\tilde{\lambda}} E' \wedge PL(<a,\tilde{\lambda}>) = l \wedge E' \in C \,|\!\}$.

Definition 9. An equivalence relation $\mathcal{B} \subseteq \mathcal{G}_\Theta \times \mathcal{G}_\Theta$ is a *strong extended Markovian bisimulation (strong EMB)* if and only if, whenever $(E_1, E_2) \in \mathcal{B}$, then for all $a \in AType$, $l \in PLSet$ and $C \in \mathcal{G}_\Theta/\mathcal{B}$ it turns out

$$ERate(E_1,a,l,C) = ERate(E_2,a,l,C) \qquad \blacksquare$$

Since the union of all the strong EMBs is a strong EMB as well, we call such a relation the *strong extended Markovian bisimulation equivalence (strong EMBE)*, denoted by $\sim_{EMB}$.

Theorem 10. *$\sim_{EMB}$ is preserved by all the operators as well as by recursive definitions.* ■

Theorem 11. *In $\mathcal{E} \times \mathcal{E}$ it turns out that $\sim_{EMB} \subseteq \sim_{FP}$.* ■

The inclusion above is strict, as one can see by considering the examples below. Additionally, such examples show that $\sim_{EMB}$ cannot abstract from either priority levels or weights of immediate actions; otherwise, the congruence property would not hold.

Example 6. Consider terms E_1 and E_2 of Example 3. Then $E_1 \sim_{FP} E_2$ but $E_1 \not\sim_{EMB} E_2$ because $ERate(E_1,a,0,[\underline{0}]_{\sim_{EMB}}) \neq ERate(E_2,a,0,[\underline{0}]_{\sim_{EMB}})$. ■

Example 7. Consider terms E_1 and E_2 of Example 4. Then $E_1 \sim_{FP} E_2$ but $E_1 \not\sim_{EMB} E_2$ because $ERate(E_1, a, 1, [\underline{0}]_{\sim_{EMB}}) \neq ERate(E_2, a, 1, [\underline{0}]_{\sim_{EMB}})$. If we relax Definition 9 to abstract from the priority level of immediate actions, then $E_1 \sim_{EMB} E_2$ but this new strong EMBE would be closed under neither the alternative composition operator nor the parallel composition operator. For example, $E_1 + <b, \infty_{1,1}>.\underline{0} \not\sim_{EMB} E_2 + <b, \infty_{1,1}>.\underline{0}$ and $E_1 \,\|_\emptyset <b, \infty_{1,1}>.\underline{0} \not\sim_{EMB} E_2 \,\|_\emptyset <b, \infty_{1,1}>.\underline{0}$ because the left-hand side terms can execute an action with type b while the right-hand side terms cannot. ■

Example 8. Consider terms E_1 and E_2 of Example 5. Then $E_1 \sim_{FP} E_2$ but $E_1 \not\sim_{EMB} E_2$ because $ERate(E_1, a, 1, [\underline{0}]_{\sim_{EMB}}) \neq ERate(E_2, a, 1, [\underline{0}]_{\sim_{EMB}})$. If we relax Definition 9 to consider execution probabilities instead of weights for immediate actions (see the notion of *relative bisimulation* proposed in [19]), then $E_1 \sim_{EMB} E_2$ but this new strong EMBE would be closed under neither the alternative composition operator nor the parallel composition operator. For example, $E_1 + <b, \infty_{1,1}>.\underline{0} \not\sim_{EMB} E_2 + <b, \infty_{1,1}>.\underline{0}$ and $E_1 \,\|_\emptyset <b, \infty_{1,1}>.\underline{0} \not\sim_{EMB} E_2 \,\|_\emptyset <b, \infty_{1,1}>.\underline{0}$ because the left-hand side terms can execute actions having type a with probability $1/2$ while the right-hand side terms can execute actions having type a with probability $2/3$. ■

As a matter of fact, it turns out that $\sim_{EMB}$, restricted to the set $\mathcal{E}_{-\tau\infty}$ of terms in $\mathcal{E}$ whose integrated semantic model has no immediate internal transitions, is the coarsest congruence contained in $\sim_{FP}$.

Theorem 12. *Let $E_1, E_2 \in \mathcal{E}_{-\tau\infty}$. Then $E_1 \sim_{EMB} E_2$ if and only if, for all $F \in \mathcal{G}$ and $S \subseteq AType - \{\tau\}$ such that $E_1 + F$, $E_2 + F$, $E_1 \,\|_S\, F$, $E_2 \,\|_S\, F \in \mathcal{E}_{-\tau\infty}$, it turns out $E_1 + F \sim_{FP} E_2 + F$ and $E_1 \,\|_S\, F \sim_{FP} E_2 \,\|_S\, F$.* ■

Example 9. Consider terms $E_1 \equiv <a, \infty_{1,1}>.A$ and $E_2 \equiv <a, \infty_{1,1}>.B$ where $A \equiv <\tau, \infty_{1,1}>.A$ and $B \equiv <\tau, \infty_{1,2}>.B$. Then $E_1 \not\sim_{EMB} E_2$ but there does not exist any context based on the alternative composition operator or the parallel composition operator that distinguishes E_1 and E_2 with respect to $\sim_{FP}$. ■

5 An Example: Dining Philosophers Problem

Suppose we are given n philosophers P_i $(0 \leq i \leq n-1)$ sat at a round table each with a plate in front of him, and n chopsticks C_i $(0 \leq i \leq n-1)$ each shared by two neighbor philosophers and used to get the rice at the center of the table. Let us denote by "$\_ +_n \_$" the sum modulo n, and let $think_i$ be the action type "P_i is thinking", pu_i (pu_{i+_n1}) be "P_i picks up C_i (C_{i+_n1})", eat_i be "P_i is eating", and pd_i (pd_{i+_n1}) be "P_i puts down C_i (C_{i+_n1})". The scenario can be described as follows:

- $DP_n \equiv (P_0 \,\|_\emptyset \ldots \|_\emptyset\, P_{n-1}) \,\|_{\{pu_i, pd_i | 0 \leq i \leq n-1\}} (C_0 \,\|_\emptyset \ldots \|_\emptyset\, C_{n-1})$
  - $P_i \stackrel{\Delta}{=} <think_i, *>.(<pu_i, *>.<pu_{i+_n1}, *>.P_i' + <pu_{i+_n1}, *>.<pu_i, *>.P_i')$
    $P_i' \stackrel{\Delta}{=} <eat_i, *>.<pd_i, *>.<pd_{i+_n1}, *>.P_i$

- $C_i \stackrel{\Delta}{=} <pu_i, *>.<pd_i, *>.C_i$

Since all the actions are passive, the system is purely nondeterministic: this is exactly the same description we would obtain with classical process algebras, so EMPA is a conservative extension of them.

As a naive solution to break the symmetry that may cause deadlock, we could introduce a precedence relation among philosophers by means of the priority levels of immediate actions, thus modifying the specification of P_i as follows:

$$\begin{aligned} P_i \stackrel{\Delta}{=}\ & <think_i, *>.<pu_i, \infty_{i+1,1}>.<pu_{i+_n1}, \infty_{i+1,1}>. \\ & <eat_i, *>.<pd_i, *>.<pd_{i+_n1}, *>.P_i \end{aligned}$$

To solve the problem in a more elegant and fair manner, we could use the randomized distributed algorithm of [15]: P_i flips a fair coin to choose between C_i and C_{i+1}, gets the chosen chopstick as soon as it becomes free, and gets the other chopstick if it is free, otherwise releases the chosen chopstick and flips the coin again. This algorithm can be easily described in EMPA through the weights of immediate actions by modifying the specification of P_i as follows:

$$\begin{aligned} P_i &\stackrel{\Delta}{=} <think_i, *>.P'_i \\ P'_i &\stackrel{\Delta}{=} <\tau, \infty_{1,1/2}>.<pu_i, *>.(<pu_{i+_n1}, *>.P''_i + <pd_i, *>.P'_i)+ \\ &\quad\ <\tau, \infty_{1,1/2}>.<pu_{i+_n1}, *>.(<pu_i, *>.P''_i + <pd_{i+_n1}, *>.P'_i) \\ P''_i &\stackrel{\Delta}{=} <eat_i, *>.<pd_i, *>.<pd_{i+_n1}, *>.P_i \end{aligned}$$

Now in the system nondeterministic and probabilistic aspects coexist, so EMPA can be viewed as a possible syntactical counterpart of formal models for randomized distributed computations such as those defined in [18].

Finally, by temporally closing the system, with EMPA we can even assess some performance indices like, e.g., the average time during which there is at least one philosopher eating, i.e. the chopstick utilization. The specification of P_i has to be modified as follows:

$$\begin{aligned} P_i &\stackrel{\Delta}{=} <think_i, \lambda_i>.P'_i \\ P'_i &\stackrel{\Delta}{=} <\tau, \infty_{1,1/2}>.<pu_i, \infty_{1,1}>.(<pu_{i+_n1}, \infty_{1,1}>.P''_i + <pd_i, \infty_{1,1}>.P'_i)+ \\ &\quad\ <\tau, \infty_{1,1/2}>.<pu_{i+_n1}, \infty_{1,1}>.(<pu_i, \infty_{1,1}>.P''_i + <pd_{i+_n1}, \infty_{1,1}>.P'_i) \\ P''_i &\stackrel{\Delta}{=} <eat_i, \mu_i>.<pd_i, \infty_{1,1}>.<pd_{i+_n1}, \infty_{1,1}>.P_i \end{aligned}$$

Observe that actions pu_i and pd_i have been modeled as immediate, because they are irrelevant from the performance evaluation point of view. Thus immediate actions provide a mechanism for *performance abstraction*, in the same way as action type τ provides a mechanism for functional abstraction.

6 Conclusion

In this paper we have proposed the stochastic process algebra EMPA in order to integrate functional and performance aspects in the modeling and analysis of concurrent systems. The development of EMPA has been influenced by the stochastic process algebras MTIPP [8] and PEPA [10], and by the formalism of GSPNs [1]. While designing EMPA, emphasis has been placed on expressiveness and formal semantics.

In particular, in EMPA action durations are mainly expressed by means of exponentially distributed random variables (like in MTIPP and PEPA), but it is also possible to express immediate actions (like in MTIPP) each of which is assigned a priority level and a weight (like in GSPNs), as well as actions whose duration is unspecified (like in classical process algebras). As a consequence, the expressiveness of EMPA is the sum of the expressiveness of a classical process algebra, a probabilistic process algebra (resulting in stratified models of probabilistic processes [7], though also reactive models are expressible via the interplay of immediate and passive actions), a prioritized process algebra (consisting of an extension of the approach proposed in [5]), and an exponentially timed process algebra. The price to pay for this enhanced expressiveness from the point of view of the underlying theory is relatively low: the idea of potential move in the definition of the integrated semantics is quite intuitive, and the notion of equivalence is simple and elegant.

The usefulness of all the features above is illustrated both in the previous section and in [3] by means of examples based on:

- Queueing systems [13]. They are models largely used for performance evaluation purposes whenever the scenario under study can be described by means of the interaction between a population of customers requiring service from a set of servers. In [3] we have stressed the expressiveness of EMPA by representing queueing systems with scalable service rate and with customers requiring different service rates or having different priorities; we have also described queueing systems with forks and joins as well as queueing networks.
- Communication protocols. In [3] the alternating bit protocol has been described by means of EMPA. From the specification we have obtained the integrated semantic model (composed of 302 states), which has been then projected on the functional model (by discarding action rates) and the performance model (composed of only 33 states because many original states had only immediate transitions, and moreover the phase of ordinary lumping has been able to capture the symmetry of the protocol). The projected models have finally been analyzed to detect functional properties (such as the absence of deadlock) as well as performance measures (such as the protocol throughput and the channel utilization).

Finally, we recall that there are several open problems in the field of stochastic process algebras, as listed in [3]. Probably, the most challenging one is the extension to generally distributed durations. It is however important to point out that the limitation to exponentially distributed durations is (*i*) convenient because exponential timing allows for a Markovian analysis without resorting to time-costly simulations, and (*ii*) not so restrictive because many frequently occurring distributions are (or can be approximated by) phase-type distributions [17], and these are expressible in EMPA by means of the interplay of exponentially timed actions and weights of immediate actions.

References

1. M. Ajmone Marsan, G. Balbo, G. Conte, *"A Class of Generalized Stochastic Petri Nets for the Performance Evaluation of Multiprocessor Systems"*, in ACM Trans. on Computer Systems 2:143-172, 1984
2. J. Baeten, J. A. Bergstra, J. W. Klop, *"Syntax and Defining Equations for an Interrupt Mechanism in Process Algebra"*, in Fundamenta Informatica IX:127-168, 1986
3. M. Bernardo, L. Donatiello, R. Gorrieri, *"Integrating Performance and Functional Analysis of Concurrent Systems with EMPA"*, Technical Report UBLCS-95-14, University of Bologna (Italy), September 1995 (revised March 1996), available via anonymous ftp from `ftp.cs.unibo.it:/pub/TR/UBLCS`
4. P. Buchholz, *"Markovian Process Algebra: Composition and Equivalence"*, in Proc. of PAPM '94, Erlangen (Germany), pages 11-30, July 1994
5. R. Cleaveland, M. Hennessy, *"Priorities in Process Algebras"*, in Proc. of LICS '88, Edinburgh (UK), IEEE-CS Press, pages 193-202, July 1988
6. D. Ferrari, *"Considerations on the Insularity of Performance Evaluation"*, in IEEE Trans. on Software Engineering 12(6):678-683, June 1986
7. R. van Glabbeek, S. A. Smolka, B. Steffen, C. M. N. Tofts, *"Reactive, Generative and Stratified Models of Probabilistic Processes"*, in Proc. of LICS '90, Philadelphia (PA), IEEE-CS Press, pages 130-141, 1990
8. N. Götz, U. Herzog, M. Rettelbach, *"Multiprocessor and Distributed System Design: the Integration of Functional Specification and Performance Analysis Using Stochastic Process Algebras"*, in Proc. of PERFORMANCE '93, Rome (Italy), LNCS 729:121-146, September 1993
9. H. Hermanns, M. Rettelbach, *"Syntax, Semantics, Equivalences, and Axioms for MTIPP"*, in Proc. of PAPM '94, Erlangen (Germany), pages 71-87, July 1994
10. J. Hillston, *"A Compositional Approach to Performance Modelling"*, Ph.D. Thesis, University of Edinburgh (UK), March 1994
11. C. A. R. Hoare, *"Communicating Sequential Processes"*, Prentice Hall, 1985
12. J. G. Kemeny, J. L. Snell, *"Finite Markov Chains"*, Springer-Verlag, 1977
13. L. Kleinrock, *"Queueing Systems"*, Wiley, 1975
14. K. G. Larsen, A. Skou, *"Bisimulation through Probabilistic Testing"*, in Information and Computation 94(1):1-28, September 1991
15. D. Lehmann, M. Rabin, *"On the Advantage of Free Choice: A Symmetric and Fully Distributed Solution to the Dining Philosophers Problem"*, in Proc. of POPL '81, pages 133-138, 1981
16. R. Milner, *"Communication and Concurrency"*, Prentice Hall, 1989
17. M. F. Neuts, *"Matrix-Geometric Solutions in Stochastic Models - An Algorithmic Approach"*, John Hopkins University Press, 1981
18. R. Segala, *"Modeling and Verification of Randomized Distributed Real-Time Systems"*, Ph.D. Thesis, MIT, June 1995
19. C. Tofts, *"A Synchronous Calculus of Relative Frequency"*, in Proc. of CONCUR '90, Amsterdam (The Netherlands), LNCS 458:467-480, August 1990
20. Y. Yemini, J. Kurose, *"Towards the Unification of the Functional and Performance Analysis of Protocols, or Is the Alternating-Bit Protocol Really Correct?"*, in Protocol Specification, Testing and Verification II, 1982

This article was processed using the LaTeX macro package with LLNCS style