On the Coexistence of Exponential, Immediate and Passive Actions in EMPA

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

Abstract

EMPA is a process algebra supporting three different kinds of actions: exponentially timed actions, prioritised weighted immediate actions, and passive actions. We report some considerations and examples about their coexistence, with respect to both the resulting expressiveness and the underlying theory. We show that a good trade off between expressiveness and theory has been achieved: the expressiveness is considerable while the theory remains relatively simple.

1 Introduction

The development of the stochastic process algebra EMPA [3, 4] has been strongly influenced by the stochastic process algebras MTIPP [11] and PEPA [13], and by the formalism of GSPNs [1]. This is witnessed by the fact that in EMPA there are three different kinds of actions: exponentially timed actions (taken from MTIPP and PEPA), prioritised weighted immediate actions (similar to immediate transitions of GSPNs), and passive actions (basically taken from PEPA). Exponentially timed actions describe activities whose duration is exponentially distributed: these are the activities that are relevant from the performance point of view. Prioritised weighted immediate actions have duration zero: they model logical events as well as activities that are either irrelevant from the performance point of view or unboundedly faster than the others, and are useful to express both prioritised choices and probabilistic choices. Finally, passive actions have unspecified duration: they model activities waiting for the synchronisation with exponentially timed or immediate activities, and are useful to express nondeterministic choices.

The aim of this paper is to investigate the consequence of the coexistence of the three kinds of actions above, from the point of view of both the resulting expressiveness and the underlying theory. The main observation is that a considerable expressiveness is obtained while the underlying theory, i.e. the definition of the semantics and the notion of equivalence, is relatively simple.

The paper is organised as follows. In Section 2 we recall the syntax and the semantics of EMPA terms. In Section 3 we show that the coexistence of the different kinds of actions can be thought of as the coexistence of a nondeterministic kernel, a probabilistic kernel, a prioritised kernel and an exponentially timed kernel: this means that EMPA can be used as a classical process algebra, a probabilistic process algebra, a prioritised process algebra, or an exponentially timed process algebra. In Section 4 we study the kernel interaction via the binary operators: we stress that the alternative composition operator is parametric in the nature of the choice, and that some kinds of communication different from the client-server one are expressible in an indirect manner. In Section 5 we analyse the kernel interaction in the definition of the integrated interleaving semantics: we highlight the fact that the definition of this semantics can be kept reasonably simple thanks to the idea of potential move. In Section 6 we investigate the kernel interaction with respect to the notion of equivalence: the main result is that several proposals of equivalences for the various kernels can be combined together in an elegant way to give rise to a congruence. In Section 7 we report some examples of the expressiveness of EMPA and the usefulness of all its kernels. We conclude in Section 8 by reporting some remarks on further enhancements of the expressiveness of EMPA and the related consequences on the underlying theory.

2 Syntax and semantics of EMPA terms

In this section we briefly recall the syntax and the semantics of EMPA terms. The interested reader is referred to [3] for a comprehensive presentation.

Each EMPA action is a pair "$<a,\tilde{\lambda}>$" consisting of the *type* and the *rate* of the action. The rate indicates the speed at which the action occurs: 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 synchronising it with an active action of the same type.

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 $\mathbb{R}_+$.

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 *relabelling functions.*

Definition 2.1 The set $\mathcal{L}$ of EMPA *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 $\mathcal{L}$. ■

Since the *null term* "$\underline{0}$", the *prefix operator* "$<a,\tilde{\lambda}>.\_$", the *functional abstraction operator* "$\_/L$", and the *functional relabelling operator* "$\_[\varphi]$" are typical of classical process algebras, we concentrate on the 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 *alternative composition operator* "$\_+\_$" expresses a choice between two terms. The choice is based on the *race policy*: the action having the least duration is executed, and the corresponding term is selected. This implies that immediate actions take precedence over exponentially timed actions.

The *parallel composition operator* "$\_\parallel_S\_$" is based on two synchronisation disciplines. The synchronisation discipline on action types is the same as that of CSP [15]. The synchronisation discipline on action rates states that action $<a,\tilde{\lambda}>$ can be synchronised with action $<a,\tilde{\mu}>$ if and only if $\min(\tilde{\lambda},\tilde{\mu}) = *$, [1] and the rate of the resulting action is given by $\max(\tilde{\lambda},\tilde{\mu})$ up to normalisation. In other words, in a synchronisation at most one active action can be involved and its rate determines the rate of the resulting action, up to normalisation.

We conclude this brief overview of EMPA by introducing its semantics. The main problem to tackle 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 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.

The formal definition of the integrated interleaving semantics for EMPA is in Table 1. [2] 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 $Min : (ARate \times ARate) \longrightarrow\!\!\!\!\!\!\circ\;\; ARate$, which are defined in the fourth part of Table 1. [3]

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 normalisation of the rates of potential moves resulting from the synchronisation of the same active action with several independent or alternative passive actions is carried out through partial function $Norm : (AType \times ARate \times ARate \times \mathcal{M}u_{fin}(Act \times$

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

[2] We use "$\{\!|$" and "$|\!\}$" as brackets for multisets and "$\oplus$" as multiset union symbol, and we denote by $\mathcal{M}u_{fin}(S)$ the set of all the multisets over S, $M(s)$ the multiplicity of s, and $\pi_i(M)$ the multiset resulting from the projection of the tuples of $M \in \mathcal{M}u_{fin}(S_1 \times \ldots \times S_n)$ onto their i-th component.

[3] *Min* computes the minimum of a set of random variables, as required by the race policy. *Min* is extended to multisets of action rates having the same priority by assuming it is commutative and associative.

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

$$\begin{aligned}
&PM(\underline{0}) = \emptyset\\
&PM(<a,\tilde{\lambda}>.\,E) = \{\!|\,(<a,\tilde{\lambda}>,E)\,|\!\}\\
&PM(E/L) = \{\!|\,(<a,\tilde{\lambda}>,E'/L) \mid (<a,\tilde{\lambda}>,E') \in PM(E) \wedge a \notin L\,|\!\} \oplus\\
&\qquad\qquad\quad \{\!|\,(<\tau,\tilde{\lambda}>,E'/L) \mid (<a,\tilde{\lambda}>,E') \in PM(E) \wedge a \in L\,|\!\}\\
&PM(E\backslash H) = \{\!|\,(<a,\tilde{\lambda}>,E'\backslash H) \mid (<a,\tilde{\lambda}>,E') \in PM(E) \wedge \neg(a \in H \wedge \tilde{\lambda} = *)\,|\!\}\\
&PM(E[\varphi]) = \{\!|\,(<\varphi(a),\tilde{\lambda}>,E'[\varphi]) \mid (<a,\tilde{\lambda}>,E') \in PM(E)\,|\!\}\\
&PM(E_1 + E_2) = PM(E_1) \oplus PM(E_2)\\
&PM(E_1 \,\|_S\, E_2) = \{\!|\,(<a,\tilde{\lambda}>,E'_1 \,\|_S\, E_2) \mid a \notin S \wedge (<a,\tilde{\lambda}>,E'_1) \in PM(E_1)\,|\!\} \oplus\\
&\qquad\qquad\qquad\quad \{\!|\,(<a,\tilde{\lambda}>,E_1 \,\|_S\, E'_2) \mid a \notin S \wedge (<a,\tilde{\lambda}>,E'_2) \in PM(E_2)\,|\!\} \oplus\\
&\qquad\qquad\qquad\quad \{\!|\,(<a,\tilde{\gamma}>,E'_1 \,\|_S\, E'_2) \mid a \in S \wedge\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad (<a,\tilde{\lambda}>,E'_1) \in PM(E_1) \wedge\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad (<a,\tilde{\mu}>,E'_2) \in PM(E_2) \wedge\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad \tilde{\gamma} = \mathit{Norm}(a,\tilde{\lambda},\tilde{\mu},PM(E_1),PM(E_2))\,|\!\}\\
&PM(A) = PM(E) \quad \text{if } A \stackrel{\Delta}{=} E
\end{aligned}$$

$$\begin{aligned}
&\mathit{Select}(PM) = \{\!|\,(<a,\tilde{\lambda}>,E) \in PM \mid PL(<a,\tilde{\lambda}>) = -1 \vee\\
&\qquad\qquad\qquad\qquad\qquad\qquad \forall (<b,\tilde{\mu}>,E') \in PM.\, PL(<a,\tilde{\lambda}>) \geq PL(<b,\tilde{\mu}>)\,|\!\}\\
&\qquad PL(<a,*>) = -1 \quad PL(<a,\lambda>) = 0 \quad PL(<a,\infty_{l,w}>) = l
\end{aligned}$$

$$\begin{aligned}
&\mathit{Melt}(PM) = \{(<a,\tilde{\lambda}>,E) \mid (<a,\tilde{\mu}>,E) \in PM \wedge\\
&\qquad\qquad\qquad\qquad \tilde{\lambda} = \mathit{Min}\{\!|\,\tilde{\gamma} \mid (<a,\tilde{\gamma}>,E) \in PM \wedge PL(<a,\tilde{\gamma}>) = PL(<a,\tilde{\mu}>)\,|\!\}\}\\
&\qquad * \,\mathit{Min}\, * = * \quad \lambda \,\mathit{Min}\, \lambda' = \lambda + \lambda' \quad \infty_{l,w} \,\mathit{Min}\, \infty_{l,w'} = \infty_{l,w+w'}
\end{aligned}$$

$$\begin{aligned}
&\mathit{Norm}(a,\tilde{\lambda},\tilde{\mu},PM_1,PM_2) = \begin{cases} \mathit{Split}(\tilde{\lambda},1/(\pi_1(PM_2))(<a,*>)) & \text{if } \tilde{\mu} = *\\ \mathit{Split}(\tilde{\mu},1/(\pi_1(PM_1))(<a,*>)) & \text{if } \tilde{\lambda} = * \end{cases}\\
&\qquad \mathit{Split}(*,\alpha) = * \quad \mathit{Split}(\lambda,\alpha) = \lambda \cdot \alpha \quad \mathit{Split}(\infty_{l,w},\alpha) = \infty_{l,w\cdot\alpha}
\end{aligned}$$

Table 1 EMPA integrated interleaving semantics

$\mathcal{G}) \times \mathcal{M}u_{\mathit{fin}}(Act \times \mathcal{G})) \multimap ARate$ 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 order for a synchronisation to be permitted.

Definition 2.2 The *integrated operational interleaving semantics* of $E \in \mathcal{G}$ is the labelled transition system $\mathcal{I}[\![E]\!] = (\uparrow E, Act, \longrightarrow_E, E)$ where $\uparrow E$ is the set of states reachable from E via $\longrightarrow$, and $\longrightarrow_E$ is $\longrightarrow$ restricted to $\uparrow E \times Act \times \uparrow E$. ■

3 EMPA kernels

Due to the coexistence of exponentially timed actions, prioritised weighted immediate actions, and passive actions, EMPA can be viewed as being made out of four kernels (see Figure 1): a nondeterministic kernel, a probabilistic kernel, a prioritised kernel, and an exponentially timed kernel.

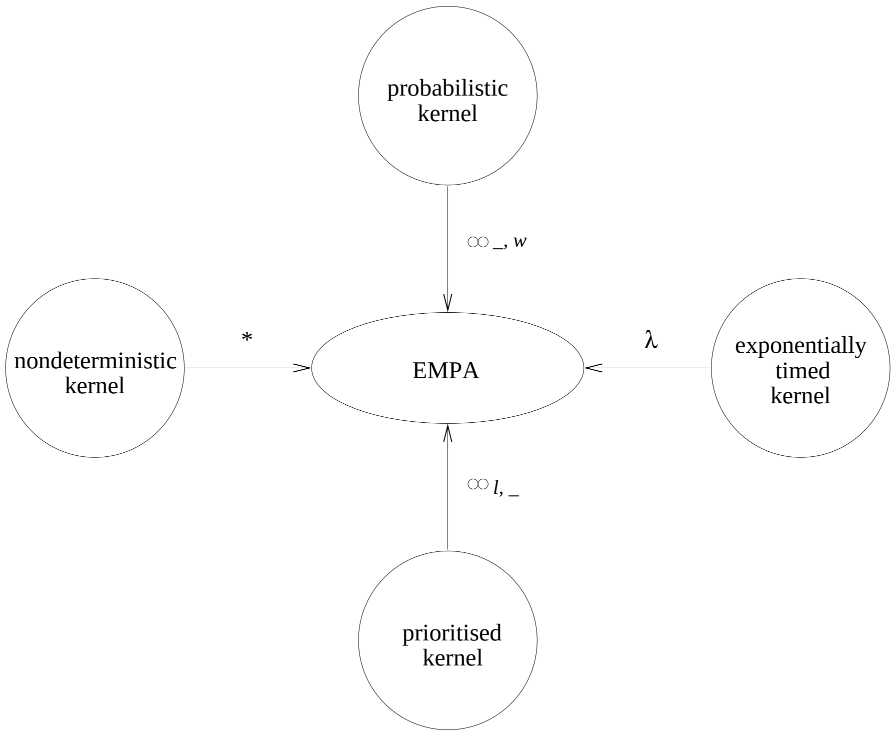

Figure 1 EMPA kernels

3.1 Nondeterministic kernel

The *nondeterministic kernel* EMPA$_{nd}$ is the sublanguage of EMPA obtained by considering only passive actions. Since the duration of passive actions is completely unspecified, EMPA$_{nd}$ is a classical process algebra and allows for pure nondeterminism.

In particular, the operators of EMPA$_{nd}$ are a mix of the operators of CCS [19] and CSP [15]: the functional abstraction operator coincides with the hiding operator of CSP, the temporal restriction operator reduces to the restriction operator of CCS, the functional relabelling operator coincides with the relabelling operator of CCS, and the parallel composition operator reduces to the parallel composition operator of CSP since the constraint on action rates is always satisfied in EMPA$_{nd}$.

3.2 Probabilistic kernel

The *probabilistic kernel* EMPA$_{pb,l}$ is the sublanguage of EMPA obtained by considering only immediate actions having the same priority level l. Since each such immediate action is given a weight, EMPA$_{pb,l}$ is a probabilistic process algebra.

In particular, EMPA$_{pb,l}$ is much more similar to WSCCS [25] than PCCS [9] because weights instead of probabilities are specified for operational convenience: the main difference between EMPA$_{pb,l}$ and WSCCS is that the former is not a synchronous calculus, i.e. the computation does not proceed in locksteps. Additionally, since in EMPA$_{pb,l}$ there are no passive actions, the temporal restriction operator has no effect, while the parallel composition operator also acts as a restriction operator because the synchronisation constraint on action rate is never satisfied. This means that in EMPA$_{pb,l}$ synchronisations are not allowed.

Following the classification of models of probabilistic processes proposed in [10], it is easily seen that EMPA$_{pb,l}$ does not produce reactive models because the choice among enabled actions having different types is probabilistic instead of nondeterministic. As a consequence, EMPA$_{pb,l}$ produces generative models because the relative frequency of performing actions having different types is explicitly described. More accurately, EMPA$_{pb,l}$ is even finer as it can produce stratified models where the intended relative frequencies of actions are preserved in a levelwise fashion in the presence of a restriction.

Example 3.1 Consider an operating system having three processes to be multiprogrammed: the garbage collector gc, user process up_1, and user process up_2. Suppose that each process is given 1/3 of the CPU cycles, and that this frequency must be preserved for the garbage collector whenever one of the user processes is denied further access to the machine due to a restriction context. The corresponding EMPA$_{pb,l}$ term is

$$\mathit{Sched} \stackrel{\Delta}{=} <gc,\infty_{l,1}>.\mathit{Sched} + <\tau,\infty_{l,2}>.(<up_1,\infty_{l,1}>.\mathit{Sched} + <up_2,\infty_{l,1}>.\mathit{Sched})$$

If we consider $\mathit{Sched} \parallel_{\{up_2\}} \underline{0}$, then we have that the execution probability of the action having type gc is still 1/3, as required. ■

3.3 Prioritised kernel

The *prioritised kernel* EMPA$_{pt,w}$ is the sublanguage of EMPA obtained by considering only immediate actions having the same weight w. Since each immediate action is given a priority level, EMPA$_{pt,w}$ is a prioritised process algebra. As in the case of the probabilistic kernel, the temporal restriction operator has no effect and the parallel composition operator also acts as a restriction operator.

It is worth noting that an explicit priority level is associated with each action. Therefore, from the syntactical point of view our approach is similar to the proposal of [7] where for each action type both a prioritised version and an unprioritised version are provided: the difference is that in [7] unprioritised actions are preempted only by internal prioritised actions in order to achieve compositionality. Our approach is different from that presented in [2], where the emphasis is on equational reasoning and the priority structure is imposed only within the scope of a priority operator, from that presented in [8], where a priority choice operator is explicitly defined, and from those presented in [24, 25], where priority is expressed as extremal probability. Concerning the important features when modelling priority according to [24], we have that:

- ☐ Like [7, 8, 24, 25], the priority relation of EMPA$_{pt,w}$ is globally dynamic, i.e. it may be the case that an action with type a has priority over an action with type b in one state, and the converse in some other state.
- ☐ Unlike [2], the priority relation of EMPA$_{pt,w}$ cannot define arbitrary partial orders because the levelled priority structure causes incomparable actions to have the same priority. As an example, it is not possible in EMPA$_{pt,w}$ to express the fact that, in a given state, action type a takes precedence over action types b and c while action type d takes precedence only over action type c.
- ☐ Like [7, 24, 25], the priority relation of EMPA$_{pt,w}$ is modelled transparently and intuitively.

3.4 Exponentially timed kernel

The *exponentially timed kernel* EMPA$_{et}$ is the sublanguage of EMPA obtained by considering only exponentially timed actions. As in the case of the probabilistic and the prioritised kernels, the temporal restriction operator has no effect and the parallel composition operator also acts as a restriction operator.

Since exponentially timed actions are given nonzero durations, they account for the passage of time thereby making it possible to model and analyse the system performance. In particular, performance models are formalised by means of homogeneous continuous-time Markov chains, because the durations of transitions are exponentially distributed and the corresponding rates are time-independent. Another important consequence of the restriction to exponentially distributed durations is that the memoryless property allows the integrated semantics to be defined in the interleaving style.

3.5 Comparison with other Markovian process algebras

From the fact that EMPA is composed of the four kernels shown in Figure 1, it follows that EMPA can be used as a classical process algebra, a probabilistic process algebra, a prioritised process algebra or an exponentially timed process algebra. This is not possible with other Markovian process algebras:

- ☐ MTIPP [11], MPA [6], the Markovian process algebra proposed in [5], and Sπ [21] contain only an exponentially timed kernel implemented by means of exponentially timed actions.
- ☐ PEPA [13] contains a probabilistic kernel implemented by means of weighted passive actions, and an exponentially timed kernel implemented by means of exponentially timed actions.
- ☐ The union of PMTIPP [22] and IMTIPP [12] contains a nondeterministic kernel implemented by means of immediate actions, a probabilistic kernel implemented by means of a probabilistic choice operator, and an exponentially timed kernel implemented by means of exponentially timed actions.

In summary, the expressiveness of EMPA is the sum of the expressiveness of a classical, a probabilistic, a prioritised and an exponentially timed process algebra. It is however worth pointing out that EMPA$_{pb,l}$, EMPA$_{pt,w}$ and EMPA$_{et}$ do not allow for synchronisations. This emphasises the "gluing" role played by EMPA$_{nd}$.

4 Kernel interaction via binary operators

In this section we examine how kernels interact when terms enabling actions of different kinds are combined through the binary operators of EMPA, i.e. the alternative composition operator and the parallel composition operator.

4.1 Alternative composition operator

The nature of the choice expressed by the alternative composition operator heavily depends upon the involved actions. In its simpler form, the alternative composition operator describes a choice between two actions (executable by two alternative terms) whose nature is:

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

It is worth noting that EMPA has only one alternative composition operator for modelling all of these kinds of choice. In other words, the alternative composition operator of EMPA is parametric in the nature of the choice. Among the Markovian process algebras comprising more than one kernel, such a feature is present both in PEPA [13] and IMTIPP [12], while in PMTIPP [22] a probabilistic choice operator is explicitly introduced.

Coming back to the classification of models of probabilistic processes proposed in [10], we now realise that reactive models, which cannot be described in the probabilistic kernel, can be expressed by means of the interplay of the nondeterministic kernel and the probabilistic kernel.

Example 4.1 Suppose that two people want to flip a coin in order to make a decision: only one of them actually flips the coin, and the outcome of the coin toss is a head with probability 1/2, a tail with probability 1/2. This scenario can be modelled as follows:

$$\begin{aligned} \mathit{GetCoin} &\triangleq <\mathit{person}_1, *>.\,\mathit{FlipCoin} + <\mathit{person}_2, *>.\,\mathit{FlipCoin} \\ \mathit{FlipCoin} &\triangleq <\mathit{flip},\infty_{l,1/2}>.\,\mathit{Head} + <\mathit{flip},\infty_{l,1/2}>.\,\mathit{Tail} \end{aligned}$$

The underlying model is reactive because the relative frequency with which the two people get the coin is not specified, i.e. it is left to the environment, while the probability of the outcome of the coin toss is governed by the system. ■

We conclude by observing that the capability of expressing nondeterministic and explicit probabilistic choices is essential to cope with those cases in which the race policy is not adequate, i.e. those cases in which the choice between two actions is made independently of their durations: one of the two actions is chosen (either nondeterministically or probabilistically), and then that action is executed. As a matter of fact, the adoption of the race policy seems to be appropriate for the parallel composition operator because the competing actions do not share any resource, while in the case of the alternative composition operator the choice phase and the execution phase should be clearly separated because only one resource is available for all the actions involved.

4.2 Parallel composition operator

In the case of the alternative composition operator, the interaction of different kernels takes place only in implicit prioritised choices. In the case of the parallel composition operator, instead, the interaction among kernels is more evident. In particular, the nondeterministic kernel plays a prominent role because the synchronisation discipline on action rates requires that at most one active action can be involved, while all the other involved actions must be passive.

The main consequence is that only client-server communications are directly expressible: the rate of the action resulting from a synchronisation is determined by the rate of the only possible active action involved in the synchronisation itself. This choice has been made due to its simplicity, since it avoids the need to define the rate of the action deriving from the synchronisation of two active actions. Also, this choice has been made due to its modularity. When modelling an n-way synchronisation, only the designer of the active component must know the rate of the synchronisation, while the other $n-1$ designers can get rid of it by leaving it unspecified through passive actions. Furthermore, possible subsequent changes of the rate affect only one component.

To compute correctly the rate of the action resulting from a synchronisation according to the bounded capacity assumption [14], a normalisation is required that takes into account the number of alternative or independent passive actions that can be synchronised with the active action at hand.

Example 4.2 A queueing system $M/M/2/2$ with arrival rate λ and service rate μ [16] can be represented as follows:

□ $\mathit{System}_{M/M/2/2} \triangleq \mathit{Arrivals} \parallel_{\{a\}} \mathit{Servers}_2$:

- ▷ $\mathit{Arrivals} \triangleq <a,\lambda>.\,\mathit{Arrivals}$;
- ▷ $\mathit{Servers}_2 \triangleq S \parallel_{\emptyset} S$:

$\diamond$ $S \triangleq <a, *>. <s,\mu>. S$.

The normalisation operates in such a way that in $\mathcal{I}[\![System_{M/M/2/2}]\!]$ the two transitions leaving the initial state have rate $\lambda/2$, so the exit rate λ of the involved active component *Arrivals* is preserved. Such a normalisation is completely transparent to the user in EMPA, PEPA [13] and Sπ [21] since it is embodied in the semantic rules. On the contrary, in the case of MTIPP [11] no normalisation is carried out because the bounded capacity assumption is not made. As a consequence, it is responsibility of the designer to define the rates of actions with type a in both terms S so that their sum is 1, otherwise the expected underlying Markov chain would not be obtained. This problem has been alleviated in IMTIPP [12] and PMTIPP [22] by means of the introduction of immediate actions and a probabilistic choice operator. ■

Despite of the fact that client-server communications frequently occur in computing systems, it would be useful to be able to describe other kinds of communication, as recognised in [14]. Some of them can be described in other Markovian process algebras: for example, flexible client server-communications in MTIPP [11], and patient communications in PEPA [13] and Sπ [21]. Concerning the flexible client-server communication, it can be modelled indirectly in EMPA.

Example 4.3 Assume that a queueing system $M/M/1/q$ with arrival rate λ provides service at a speed depending on the number of customers in the queue. This queueing system can be modelled as follows:

$\square$ $SSRSystem_{M/M/1/q} \triangleq Arrivals \parallel_{\{a\}} (Queue_0 \parallel_{\{d_h | 1 \leq h \leq q-1\}} Server)$:

$\triangleright$ $Arrivals \triangleq <a,\lambda>. Arrivals$;

$\triangleright$ $Queue_0 \triangleq <a, *>. Queue_1$,
$Queue_h \triangleq <a, *>. Queue_{h+1} + <d_h, *>. Queue_{h-1}, \; 0 < h < q-1$,
$Queue_{q-1} \triangleq <d_{q-1}, *>. Queue_{q-2}$;

$\triangleright$ $Server \triangleq <d_1,\infty_{1,1}>. Server_1 + \ldots + <d_{q-1},\infty_{1,1}>. Server_{q-1}$:

$\diamond$ $Server_h \triangleq <s,sf(h) \cdot \mu>. Server, \; 1 \leq h \leq q-1$

where μ is the basic service rate and $sf : \mathbb{N}_+ \longrightarrow \mathbb{R}_+$ describes the scaling factor. ■

Other kinds of communications are listed in [14]: polite communications, impolite communications and timed synchronisations. The following examples shows that they all are expressible indirectly with EMPA obtaining the expected underlying Markov chain.

Example 4.4 Consider two people speaking at the phone in a polite way. This scenario can be modelled as follows:

$\square$ $PPCall \triangleq Caller \parallel_{\{talk\}} Callee$:

$\triangleright$ $Caller \triangleq <talk,\lambda>. <talk, *>. \underline{0}$;

$\triangleright$ $\mathit{Callee} \stackrel{\Delta}{=} <\mathit{talk}, *>. <\mathit{talk},\mu>. \underline{0}$.

The underlying Markov chain comprises three states. The sojourn times of the first two states are exponentially distributed with rate λ and μ, respectively, and this is consistent with the fact that only one person at a time speaks. ■

Example 4.5 Consider two people speaking at the phone in an impolite way. This scenario can be modelled as follows:

$\square$ $\mathit{IPCall} \stackrel{\Delta}{=} (\mathit{Person}_1 \parallel_{\{\mathit{hang}\}} \mathit{Person}_2) \backslash \{\mathit{hang}\}$:

$\triangleright$ $\mathit{Person}_1 \stackrel{\Delta}{=} <\mathit{talk},\lambda_1>. <\mathit{hang},\infty_{1,1}>. \underline{0} + <\mathit{hang}, *>. \underline{0}$;

$\triangleright$ $\mathit{Person}_2 \stackrel{\Delta}{=} <\mathit{talk},\lambda_2>. <\mathit{hang},\infty_{1,1}>. \underline{0} + <\mathit{hang}, *>. \underline{0}$.

The underlying Markov chain comprises two states. The sojourn time of the first state is exponentially distributed with rate $\lambda_1 + \lambda_2$, and this is consistent with the fact that the two people speak simultaneously and the phone call finishes as soon as one of them stops speaking. Note the use of the temporal restriction operator: it avoids the synchronisation between the two passive actions whose type is *hang*, i.e. the fact that the two people hang without speaking. ■

Example 4.6 Consider two polite people eating a meal together: neither will start eating until both have been served their meal, then both proceed to eat at their own speed, and neither will leave the table until both have finished. This scenario can be modelled as follows:

$\square$ $\mathit{Meal} \stackrel{\Delta}{=} (\mathit{Person}_1 \parallel_{\{\mathit{start},\mathit{leave}\}} \mathit{Person}_2)$:

$\triangleright$ $\mathit{Person}_1 \stackrel{\Delta}{=} <\mathit{start},\infty_{1,1}>. <\mathit{eat},\lambda_1>. <\mathit{leave},\infty_{1,1}>. \underline{0}$;

$\triangleright$ $\mathit{Person}_2 \stackrel{\Delta}{=} <\mathit{start}, *>. <\mathit{eat},\lambda_2>. <\mathit{leave}, *>. \underline{0}$.

The underlying Markov chain comprises four states. The sojourn time of the first state is exponentially distributed with rate $\lambda_1 + \lambda_2$ (meaning that both people are eating), while the sojourn times of the second and the third state are exponentially distributed with rate λ_1 and λ_2 respectively (meaning that only one person is eating while the other person has finished and is waiting). This Markov chain describes the phase-type distribution [20] that results from the product of two independent exponential distributions, or equivalently the phase-type distributed random variable computed as the maximum of two independent exponentially distributed random variables. ■

5 Kernel interaction via integrated semantics

In this section we examine the impact of each kernel on the integrated interleaving semantics for EMPA, and the way in which such kernels interact.

The basic idea underlying the definition of the semantics in order to cope with the coexistence of different kernels, is to derive the transitions of a term only after computing the multiset of the potential moves of that term. This is motivated in our 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.

The impact of each kernel on the semantics is witnessed by the auxiliary functions used in the semantics itself:

- □ Function *Select* handles the prioritised kernel. Given a multiset of potential moves, it discards those potential moves having lower priority.
- □ Function *Melt* handles both the probabilistic kernel and the exponentially timed kernel. The reason why we compute multisets of potential moves instead of simple sets is to cope with terms like $<a,\lambda>.E + <a,\lambda>.E$ and $<a,\infty_{l,w}>.E + <a,\infty_{l,w}>.E$ since they are not equivalent to $<a,\lambda>.E$ and $<a,\infty_{l,w}>.E$ respectively. Our solution to the problem above, which is based on taking into account the multiplicity of potential moves and then merging through *Melt* those potential moves having the same action type and priority as well as the same derivative term, generates transition systems labelled on both the type and the rate of the actions. As a consequence, our solution differs both from those of MTIPP [11], the Markovian process algebra proposed in [5], and Sπ [21] because it does not burden transitions with auxiliary labels, and from that of PEPA [13] because it generates graphs in place of multigraphs. Graphs whose transitions have no auxiliary labels and have multiplicity one are generated with a different approach also in MPA [6].
- □ Function *Norm* handles the interplay of the probabilistic and the exponentially timed kernels with the nondeterministic kernel. It allows for the determination of rates resulting from synchronisations according to the bounded capacity assumption [14].

We conclude by observing that the idea of potential move is quite intuitive and keeps the definition of the integrated semantics relatively simple. Of course, the formal definition of the semantics for EMPA is more complex than the definition of the semantics for e.g. MTIPP [11] and PEPA [13]. However, if one has understood the intuition behind auxiliary functions, then the rules in the first part and the second part of Table 1 are easily readable.

6 Kernel interaction via equivalence

In this section we show how the notion of bisimulation equivalence we have developed for EMPA has been built starting from considerations about the individual kernels:

- □ The exponentially timed kernel and the probabilistic kernel should be treated according to the notion of *probabilistic bisimulation* [17], 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.
  - ▷ In the case of the exponentially timed kernel, the notion of probabilistic bisimulation must be refined by requiring additionally that two bisimilar terms

have identically distributed sojourn times. For example, if we consider terms $E_1 \equiv <a,\lambda>.\,F + <a,\mu>.\,G$ and $E_2 \equiv <a,2 \cdot \lambda>.\,F + <a,2 \cdot \mu>.\,G$, then both transitions labelled with a,λ and $a,2 \cdot \lambda$ have execution probability $\lambda/(\lambda + \mu)$, and both transitions labelled with a,μ and $a,2 \cdot \mu$ have execution probability $\mu/(\lambda+\mu)$, but the average sojourn time of E_1 is twice the average sojourn time of E_2. Due to the race policy, requiring that two bisimilar terms have identically distributed sojourn 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 exit rate to reach the same equivalence class by executing exponentially timed actions of the same type. For example, it must hold that

$$<a,\lambda>.\,F + <a,\mu>.\,F \sim <a,\lambda + \mu>.\,F$$

This coincides with the notion of *Markovian bisimulation* [11, 13, 6].

▷ In the case of the probabilistic kernel, 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

$$<a,\infty_{l,w_1}>.\,F + <a,\infty_{l,w_2}>.\,F \sim <a,\infty_{l,w_1+w_2}>.\,F$$

This coincides with the notion of *direct bisimulation* [25].

□ The nondeterministic kernel should be treated by following the classical notion of bisimulation [19]. 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

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

□ Concerning the prioritised kernel, it might seem useful to be able to write equations like

$$\begin{array}{rcll} <a,\lambda>.\,E + <b,\infty_{l,w}>.\,F & \sim & <b,\infty_{l,w}>.\,F & \\ <c,\infty_{l,w}>.\,E + <d,\infty_{l',w'}>.\,F & \sim & <d,\infty_{l',w'}>.\,F & \text{if } l' > l \end{array}$$

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

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

whose semantic rules are those in Table 1 except that the first rule becomes

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

and the following rule for Θ is introduced

$$PM(\Theta(E)) = \mathit{Select}(PM(E))$$

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

To keep the definition of equivalence as simple as possible, we formalise the key concept of *conditional exit rate* in a uniform way for all kinds of action by means of partial function $\mathit{ERate} : (\mathcal{G}_\Theta \times \mathit{AType} \times \mathit{PLSet} \times \mathcal{P}(\mathcal{G}_\Theta)) \,\circ\!\!\!\longrightarrow \mathit{ARate}$ 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 6.1 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 that

$$ERate(E_1,a,l,C) = ERate(E_2,a,l,C)$$ ■

Definition 6.2 Let $E_1,E_2 \in \mathcal{G}_\Theta$. We say that E_1 is *strongly extended Markovian bisimulation equivalent (strongly EMBE)* to E_2, written $E_1 \sim_{EMB} E_2$, if and only if there exists a strong EMB $\mathcal{B}$ such that $(E_1,E_2) \in \mathcal{B}$. ■

It is worth noting that, despite of the presence of four different kernels, the definition of $\sim_{EMB}$ is compact and elegant. Furthermore, compositionality is achieved because in [3] we have proved that $\sim_{EMB}$ is a congruence.

7 Examples

In this section we illustrate some examples that should highlight the expressive power of EMPA achieved by means of the coexistence of the four kernels.

7.1 Phase-type distributions

The limitation to exponential durations is motivated by the fact that it allows for a Markovian analysis. On the other hand, many frequently occurring distributions are given (or can be approximated) by series-parallel combinations of exponential distributions. Such combinations, called phase-type distributions [20], can be modelled in EMPA through the interplay of the exponentially timed kernel and the probabilistic kernel:

- □ An exponential distribution with rate $\lambda \in \mathbb{R}_+$ can be modelled by means of term

$$Exp_\lambda \triangleq <a,\lambda>.\underline{0}$$

- □ An n-stage hypoexponential distribution with rates $\lambda_i \in \mathbb{R}_+, 1 \leq i \leq n$, can be modelled by means of the set of inductively defined terms

$$\begin{aligned} Hypoexp_{m,\lambda_1,\dots,\lambda_m} &\triangleq <a,\lambda_1>.Hypoexp_{m-1,\lambda_2,\dots,\lambda_m},\ 2 \leq m \leq n, \\ Hypoexp_{1,\lambda} &\triangleq Exp_\lambda \end{aligned}$$

- □ An n-stage hyperexponential distribution with rates $\lambda_i \in \mathbb{R}_+, 1 \leq i \leq n$, and branching probabilities $p_i \in \mathbb{R}_{]0,1]}, 1 \leq i \leq n$, where $\sum_{i=1}^n p_i = 1$, can be modelled by means of the set of inductively defined terms

$$\begin{aligned} Hyperexp_{n,\lambda_1,\dots,\lambda_n,p_1,\dots,p_n} &\triangleq H_{n,\lambda_1,\dots,\lambda_n,p_1,\dots,p_n}, \\ H_{m,\lambda_1,\dots,\lambda_m,p_1,\dots,p_m} &\triangleq H_{m-1,\lambda_1,\dots,\lambda_{m-1},p_1,\dots,p_{m-1}} + <a,\infty_{1,p_m}>.Exp_{\lambda_m},\ 2 \leq m \leq n, \\ H_{1,\lambda,p} &\triangleq <a,\infty_{1,p}>.Exp_\lambda \end{aligned}$$

□ An n-stage Coxian distribution with rates $\lambda_i \in \mathbb{R}_+, 1 \leq i \leq n$, and branching probabilities $p_i,q_i \in \mathbb{R}_{]0,1]}$ where $p_i + q_i = 1, 1 \leq i \leq n-1$, can be modelled by means of the set of inductively defined terms

$$\begin{aligned} \mathit{Cox}_{m,\lambda_1,\dots,\lambda_m,p_1,\dots,p_{m-1},q_1,\dots,q_{m-1}} &\stackrel{\Delta}{=} {<}a,\lambda_1{>}.\,({<}a,\infty_{1,q_1}{>}.\,\underline{0} + {<}a,\infty_{1,p_1}{>}.\\ &\qquad \mathit{Cox}_{m-1,\lambda_2,\dots,\lambda_m,p_2,\dots,p_{m-1},q_2,\dots,q_{m-1}}),\ 2 \leq m \leq n,\\ \mathit{Cox}_{1,\lambda} &\stackrel{\Delta}{=} \mathit{Exp}_\lambda \end{aligned}$$

It must be noticed that in EMPA phase-type distributions cannot be described in a direct manner because they require the interplay of several exponentially timed and immediate actions. As a consequence, they have to be used carefully due to the lack of atomicity. For example, if we consider term $\mathit{Exp}_\lambda + \mathit{Hyperexp}_{2,\lambda_1,\lambda_2,p_1,p_2}$ then we realise that the right-hand side term takes precedence over the left-hand side term, so the whole term cannot be used to express the choice between an activity whose duration is exponentially distributed and another activity whose duration is hyperexponentially distributed. To overcome this drawback, the system designer should be enabled to describe directly each kind of distribution, or the race policy should be somehow bypassed. The solution to this problem is left for future research.

7.2 Prioritised queueing systems

Assume that a queueing system $M/M/1/q$ with arrival rate λ and service rate μ must serve two different types of customers. Red customers are assigned a priority level $r > b$, where b is the priority level assigned to black customers. There are two cases.

In the first case, we assume that the priority mechanism only affects the queueing discipline, i.e. we assume that possible preemption of the customer being served cannot be exercised. This queueing system can be modelled as follows:

□ $\mathit{PSystem}_{M/M/1/q} \stackrel{\Delta}{=} \mathit{Arrivals} \parallel_{\{a_r,a_b\}} (\mathit{Queue}_{0,0} \parallel_{\{d_r,d_b\}} \mathit{Server})$:

▷ $\mathit{Arrivals} \stackrel{\Delta}{=} {<}a_r,\lambda/2{>}.\,\mathit{Arrivals} + {<}a_b,\lambda/2{>}.\,\mathit{Arrivals}$;

▷ $$\begin{aligned} \mathit{Queue}_{0,0} &\stackrel{\Delta}{=} {<}a_r,*{>}.\,\mathit{Queue}_{1,0} + {<}a_b,*{>}.\,\mathit{Queue}_{0,1},\\ \mathit{Queue}_{i,0} &\stackrel{\Delta}{=} {<}a_r,*{>}.\,\mathit{Queue}_{i+1,0} + {<}a_b,*{>}.\,\mathit{Queue}_{i,1} +\\ &\quad {<}d_r,*{>}.\,\mathit{Queue}_{i-1,0},\ 0 < i < q-1,\\ \mathit{Queue}_{0,j} &\stackrel{\Delta}{=} {<}a_r,*{>}.\,\mathit{Queue}_{1,j} + {<}a_b,*{>}.\,\mathit{Queue}_{0,j+1} +\\ &\quad {<}d_b,*{>}.\,\mathit{Queue}_{0,j-1},\ 0 < j < q-1,\\ \mathit{Queue}_{i,j} &\stackrel{\Delta}{=} {<}a_r,*{>}.\,\mathit{Queue}_{i+1,j} + {<}a_b,*{>}.\,\mathit{Queue}_{i,j+1} +\\ &\quad {<}d_r,*{>}.\,\mathit{Queue}_{i-1,j} + {<}d_b,*{>}.\,\mathit{Queue}_{i,j-1},\ 0 < i,j \wedge i+j < q-1,\\ \mathit{Queue}_{q-1,0} &\stackrel{\Delta}{=} {<}d_r,*{>}.\,\mathit{Queue}_{q-2,0},\\ \mathit{Queue}_{0,q-1} &\stackrel{\Delta}{=} {<}d_b,*{>}.\,\mathit{Queue}_{0,q-2},\\ \mathit{Queue}_{i,j} &\stackrel{\Delta}{=} {<}d_r,*{>}.\,\mathit{Queue}_{i-1,j} + {<}d_b,*{>}.\,\mathit{Queue}_{i,j-1},\ 0 < i,j \wedge i+j = q-1; \end{aligned}$$

▷ $\mathit{Server} \stackrel{\Delta}{=} {<}d_r,\infty_{r,1}{>}.{<}s,\mu{>}.\,\mathit{Server} + {<}d_b,\infty_{b,1}{>}.{<}s,\mu{>}.\,\mathit{Server}$.

Note that the precedence of red customers over black ones has been enforced by means of the two immediate actions with different priority levels occurring in *Server*.

In the second case, we assume that preemption of a black customer being served can be exercised by red customers. This queueing system can be modelled as follows (*Arrivals* and $Queue_{i,j}$ are omitted since they stay the same):

- □ $PPSystem_{M/M/1/q} \stackrel{\Delta}{=} Arrivals \parallel_{\{a_r,a_b\}} (Queue_{0,0} \parallel_{\{d_r,d_b\}} PServer)$:
  - ▷ $PServer \stackrel{\Delta}{=} <d_r,\infty_{r,1}>.PServer_r + <d_b,\infty_{b,1}>.PServer_b$:
    - ◇ $PServer_r \stackrel{\Delta}{=} <s,\mu>.PServer$;
    - ◇ $PServer_b \stackrel{\Delta}{=} <s,\mu>.PServer + <d_r,\infty_{r,1}>.<s,\mu>.PServer_b$.

Note that, due to the memoryless property of the exponential distribution, there is no difference between the preemptive-restart policy (i.e., the preempted customer restarts from the beginning) and the preemptive-resume policy (i.e., the preempted customer resumes from the point at which it has been interrupted).

7.3 Queueing networks

Consider an open queueing network [16] composed of n queueing systems $M/M/1/q$ with service rates $\mu_1,\mu_2,\ldots,\mu_n$, respectively. Assume that there are n external sources of customers with rates $\lambda_1,\lambda_2,\ldots,\lambda_n$, respectively. Let us denote by $r_{i,j}$ and $p_{i,j}$ the routing action type and the routing probability, respectively, from queueing system i to queueing system j or the outside ($j = n + 1$). This queueing network can be modelled as follows:

- □ $QN \stackrel{\Delta}{=} QS_1 \parallel_{R_2} QS_2 \parallel_{R_3} \ldots \parallel_{R_n} QS_n$:
  - ▷ $QS_i \stackrel{\Delta}{=} Arrivals_i \parallel_{\{a_i\}} (Queue_{i,0} \parallel_{\{d_i,r_{i,i}\}} Server_i),\ 1 \leq i \leq n$:
    - ◇ $Arrivals_i \stackrel{\Delta}{=} <a_i,\lambda_i>.Arrivals_i$;
    - ◇ $Queue_{i,0} \stackrel{\Delta}{=} <a_i,*>.Queue_{i,1} + <r_{1,i},*>.Queue_{i,1} + \ldots + <r_{n,i},*>.Queue_{i,1}$,
      $Queue_{i,h} \stackrel{\Delta}{=} <a_i,*>.Queue_{i,h+1} + <r_{1,i},*>.Queue_{i,h+1} + \ldots +$
      $<r_{n,i},*>.Queue_{i,h+1} + <d_i,*>.Queue_{i,h-1},\ 0 < h < q-1$,
      $Queue_{i,q-1} \stackrel{\Delta}{=} <d_i,*>.Queue_{i,q-2}$:
    - ◇ $Server_i \stackrel{\Delta}{=} <d_i,\infty_{1,1}>.<s_i,\mu_i>.Router_i$:
      - ○ $Router_i \stackrel{\Delta}{=} <r_{i,1},\infty_{1,p_{i,1}}>.Server_i + \ldots + <r_{i,n+1},\infty_{1,p_{i,n+1}}>.Server_i$;
  - ▷ $R_j = \{r_{i,j},r_{j,i} \mid 1 \leq i < j\},\ 2 \leq j \leq n$.

Note that is has been very easy to model the routing of customers by means of the weights of immediate actions.

7.4 Dining philosophers

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 denoted 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:

□ $DP_n \stackrel{\Delta}{=} (P_0 \parallel_\emptyset \ldots \parallel_\emptyset P_{n-1}) \parallel_{\{pu_i, pd_i \mid 0 \leq i \leq n-1\}} (C_0 \parallel_\emptyset \ldots \parallel_\emptyset C_{n-1})$:

▷ $P_i \stackrel{\Delta}{=} <think_i, *>.(<pu_i, *>.<pu_{i+_n1}, *>.P_i' + <pu_{i+_n1}, *>.<pu_i, *>.P_i'),$
$P_i' \stackrel{\Delta}{=} <eat_i, *>.<pd_i, *>.<pd_{i+_n1}, *>.P_i;$

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

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:

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

To solve the problem in a more elegant and fair manner, we could use the randomised distributed algorithm of [18]: 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 through the weights of immediate actions of EMPA 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 randomised distributed computations such as those in [23].

Finally, by temporally closing the system, 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 utilisation. 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 modelled 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.

8 Conclusion

In this paper we have analysed the coexistence of the four kernels composing EMPA. The main result is that the expressiveness of EMPA is considerable, because it can be viewed as the sum of the expressiveness of a classical process algebra, a probabilistic process algebra, a prioritised process algebra and an exponentially timed process algebra, while the underlying theory is relatively simple, since the idea of potential move is intuitive and the notion of equivalence is elegant.

The problem we are currently investigating is how to obtain the maximum expressive power. This involves considering several aspects:

- ☐ The exponentially timed kernel should be extended to general distributions, possibly remaining in an interleaving framework. To achieve this, we have to understand which information must be added to states and transitions of the integrated semantics, how to define the performance semantics, and how to extend the notion of equivalence. We are afraid that the enhanced expressiveness obtained by means of general distributions cannot be traded with the complexity of the underlying theory.
- ☐ The adequacy of the race policy should be verified. It seems to be appropriate for the parallel composition operator, while in the case of the alternative composition operator a preselection policy should be adopted.
- ☐ The structure of the actions should be redesigned. For the time being, priority levels and weights can be attached only to immediate actions. This is due to the role played by GSPNs [1] during the development of EMPA. A more careful (and less questionable) design of the structure of the action should comprise the type, the duration, the priority level and the weight (required by the preselection policy mentioned above).
- ☐ The synchronisation discipline on action rates should be more general. At the moment, only client-server synchronisations can be directly expressed. Recalling that the alternative composition operator is parametric in the nature of the choice, it would be nice to introduce a parallel composition operator which is parametric in the synchronisation discipline on action rates.

Acknowledgements

We are grateful to Roberto Gorrieri and Holger Hermanns for the valuable discussions. This research has been partially funded by MURST and CNR.

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 Informaticae 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] M. Bernardo, R. Gorrieri, *"Extended Markovian Process Algebra"*, in Proc. of CONCUR '96, LNCS 1119:315-330, Pisa (Italy), August 1996

[5] E. Brinksma, J.-P. Katoen, R. Langerak, D. Latella, *"A Stochastic Causality-Based Process Algebra"*, in Proc. of PAPM '95, Computer Journal 38(7):553-565, Edinburgh (UK), June 1995

[6] P. Buchholz, *"Markovian Process Algebra: Composition and Equivalence"*, in Proc. of PAPM '94, pages 11-30, Erlangen (Germany), July 1994

[7] R. Cleaveland, M. Hennessy, *"Priorities in Process Algebras"*, in Information and Computation 87(1/2):58-77, 1990

[8] J. Camilleri, G. Winskel, *"CCS with Priority Choice"*, in Information and Computation 116:26-37, 1995

[9] A. Giacalone, C. C. Jou, S. A. Smolka, *"Algebraic Reasoning for Probabilistic Concurrent Systems"*, in Proc. of Working Conf. on Programming Concepts and Methods, Sea of Gallilee (Israel), 1990

[10] R. van Glabbeek, S. A. Smolka, B. Steffen, *"Reactive, Generative and Stratified Models of Probabilistic Processes"*, in Information and Computation 121:59-80, 1995

[11] H. Hermanns, M. Rettelbach, *"Syntax, Semantics, Equivalences, and Axioms for MTIPP"*, in Proc. of PAPM '94, pages 71-87, Erlangen (Germany), July 1994

[12] H. Hermanns, M. Rettelbach, T. Weiß, *"Formal Characterisation of Immediate Actions in SPA with Nondeterministic Branching"*, in Proc. of PAPM '95, Computer Journal 38(7):530-541, Edinburgh (UK), June 1995

[13] J. Hillston, *"A Compositional Approach to Performance Modelling"*, Ph.D. Thesis, University of Edinburgh (UK), March 1994

[14] J. Hillston, *"The Nature of Synchronisation"*, in Proc. of PAPM '94, pages 51-70, Erlangen (Germany), July 1994

[15] C. A. R. Hoare, *"Communicating Sequential Processes"*, Prentice Hall, 1985

[16] L. Kleinrock, *"Queueing Systems"*, Wiley, 1975

[17] K. G. Larsen, A. Skou, *"Bisimulation through Probabilistic Testing"*, in Information and Computation 94(1):1-28, 1991

[18] 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

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

[20] M. F. Neuts, *"Matrix-Geometric Solutions in Stochastic Models - An Algorithmic Approach"*, John Hopkins University Press, 1981

[21] C. Priami, *"Stochastic π-Calculus"*, in Proc. of PAPM '95, Computer Journal 38(7):578-589, Edinburgh (UK), June 1995

[22] M. Rettelbach, *"Probabilistic Branching in Markovian Process Algebras"*, in Proc. of PAPM '95, Computer Journal 38(7):590-599, Edinburgh (UK), June 1995

[23] R. Segala, *"Modeling and Verification of Randomized Distributed Real-Time Systems"*, Ph.D. Thesis, MIT, June 1995

[24] S. A. Smolka, B. Steffen, *"Priority as Extremal Probability"*, in Proc. of CONCUR '90, LNCS 458:456-466, Amsterdam (The Netherlands), August 1990

[25] C. M. N. Tofts, *"A Synchronous Calculus of Relative Frequency"*, in Proc. of CONCUR '90, LNCS 458:467-480, Amsterdam (The Netherlands), August 1990