A Survey of Markovian Behavioral Equivalences

Marco Bernardo

Università di Urbino "Carlo Bo" – Italy
Istituto di Scienze e Tecnologie dell'Informazione

Abstract. Markovian behavioral equivalences are a means to relate and manipulate the formal descriptions of systems with an underlying CTMC semantics. There are three fundamental approaches to their definition: bisimilarity, testing, and trace. In this paper we survey the major results appeared in the literature about Markovian bisimilarity, Markovian testing equivalence, and Markovian trace equivalence. The objective is to compare these equivalences with respect to a number of criteria such as their discriminating power, the exactness of the CTMC-level aggregations they induce, the achievement of the congruence property, the existence of sound and complete axiomatizations, the existence of logical characterizations, and the existence of efficient verification algorithms.

1 Introduction

Performance-oriented notations provide the designer with the capability of building performance-aware system models, which can be used in the early development stages to predict the satisfiability of certain performance requirements as well as to choose among alternative designs on the basis of their expected QoS guarantees. These notations range from more theoretical ones – like queueing networks [38], stochastic Petri nets [1], and stochastic process algebras [32] – to more applicative ones – like formal modeling languages (Modest [12]), architectural description languages (Æmilia [5]), coordination languages (StoKlaim [22]), and object-oriented modeling languages (UML SPT/MARTE [48]).

An important feature shared by most of the performance-oriented notations mentioned above is that of providing behavioral models of the systems under construction. Given two such models, establishing whether they are equivalent amounts to establishing whether the systems they represent behave the same. What is needed is thus a notion of behavioral equivalence. This would be useful not only to relate models that are syntactically different, but also to manipulate models in a way that preserves their functional and performance properties.

Among the various proposals appeared in the literature [25], there are three fundamental approaches to the definition of behavioral equivalences: bisimilarity [43, 41], testing [21], and trace [33]. In the first approach, two models are considered to be equivalent if they are able to mimic each other's behavior stepwise. In the second approach, two models are considered to be equivalent if an external observer cannot distinguish between them, with the only way for the observer to compare their behaviors being to interact with them by means of

tests and look at their reactions. In the third approach, similarly to traditional automata theory, two models are considered to be equivalent if they are able to perform the same sequences of activities.

These three approaches, originally conceived for reasoning about functional aspects, have been subsequently extended to deal with non-functional aspects. As far as performance aspects are concerned, research has mainly concentrated on models of systems with an underlying continuous-time Markov chain (CTMC) semantics. The reason is that, due to their memoryless property, exponential distributions result in a simpler mathematical treatment without sacrificing expressiveness. In fact, besides being adequate for many real-life phenomena (like arrival processes and failure events), exponential distributions provide the most appropriate stochastic approximation if only the average duration of an activity is known, and proper combinations of them (called phase-type distributions) can approximate most of the general distributions arbitrarily closely.

This has resulted in the development of the Markovian versions of bisimilarity, testing equivalence, and trace equivalence, which will be surveyed in this paper by recalling from [32, 15, 31, 14, 18, 4, 23, 10, 30, 8, 7, 9, 49] their properties.

Although behavioral equivalences abstract from specific kinds of models, most of their properties can be better investigated and understood in a process algebraic framework [41, 33, 2, 6]. For this reason, the three Markovian behavioral equivalences mentioned above will be defined in this paper over Markovian process calculi. Many such calculi have been proposed in the literature, like TIPP [27], PEPA [32], MPA [16], EMPA_{gr} [10], Sπ [44], IMC [30], and PIOA [46]. They differ for the action representation – durational actions (TIPP, PEPA, MPA, EMPA_{gr}, Sπ, PIOA) vs. instantaneous actions separated from time passing (IMC) – as well as for the action synchronization discipline – symmetric (TIPP, MPA, Sπ, IMC), asymmetric (EMPA_{gr}, PIOA), or both (PEPA).

In this paper we shall start with a sequential Markovian process calculus (SMPC) with durational actions, which generates all the finite CTMCs with as few operators as possible: the null term, the action prefix operator, the alternative composition operator, and recursion. Then we shall add a parallel composition operator governed by an asymmetric action synchronization discipline, thus resulting in a concurrent Markovian process calculus (CMPC). We shall also address some syntax variations – like the inclusion of rewards, nondeterminism, and prioritized/weighted immediate actions – in order to present some useful variants of the considered equivalences.

Markovian bisimilarity, Markovian testing equivalence, and Markovian trace equivalence will be compared with respect to the following criteria:

1. *Discriminating power.* The three Markovian behavioral equivalences, together with some variants of Markovian trace equivalence, will be ordered according to a finer-than/coarser-than relation, thus providing information about the linear-time/branching-time spectrum in the Markovian case. As we shall see, similarly to what happens in the probabilistic case [36, 35], the Markovian spectrum is more condensed than the nondeterministic one [25].

2. *Exactness.* Each of the three Markovian behavioral equivalences induces an aggregation at the CTMC level. In general, a CTMC aggregation is exact whenever the transient/stationary probability of being in a macrostate of an aggregated CTMC is the sum of the transient/stationary probabilities of being in one of the constituent microstates of the original CTMC. This guarantees the preservation of the performance characteristics when going from the original CTMC to the aggregated one. As we shall see, all the three Markovian behavioral equivalences induce exact aggregations. In other words, the three approaches – bisimilarity, testing, trace – to the definition of behavioral equivalences are not only intuitively appropriate from the functional viewpoint, but also meaningful for performance evaluation purposes.
3. *Congruence.* A behavioral equivalence that is a congruence with respect to the typical process algebraic operators is particularly helpful in practice, as it supports compositional reasoning. This enables the compositional reduction of the model state space. As we shall see, Markovian bisimilarity and Markovian testing equivalence are congruences, whereas Markovian trace equivalence – unlike the nondeterministic case but similarly to the probabilistic one [36] – is not a congruence with respect to parallel composition.
4. *Axiomatization.* The axiomatization of a behavioral equivalence elucidates the fundamental equational laws on which the equivalence relies. This equational characterization is thus useful to understand what models can be related by the equivalence. Whenever it is sound and complete, the axiomatization gives rise to the specific rules of a deduction system – including reflexivity, symmetry, transitivity, and substitutivity – that can be exploited as a rewriting system to syntactically manipulate the models in a way that is consistent with the equivalence.
5. *Logical characterization.* The modal/temporal logic characterization of a behavioral equivalence shows what behavioral properties are preserved by the equivalence. This can be exploited to provide diagnostic information that explains why two models are not equivalent. As we shall see, Markovian bisimilarity preserves branching-time properties, while Markovian trace equivalence preserves linear-time properties.
6. *Verification complexity.* In order to be applicable in practice, a behavioral equivalence must be equipped with an efficient verification algorithm. As we shall see, not only Markovian bisimilarity but also Markovian testing and trace equivalences – unlike the nondeterministic case [37] but similarly to the probabilistic one [35] – are all decidable in polynomial time.

This paper is organized as follows. In Sect. 2 we introduce the syntax and the semantics for SMPC and CMPC. In Sect. 3 we study Markovian bisimilarity. In Sect. 4 we address some of its variants that include rewards, nondeterminism, and prioritized/weighted immediate actions. In Sect. 5 we present Markovian testing equivalence. In Sect. 6 we illustrate Markovian trace equivalence together with some other trace-based Markovian behavioral equivalences. Finally, in Sect. 7 we summarize the comparison of the Markovian behavioral equivalences based on the criteria explained above and we discuss some open problems.

2 Basic Markovian Process Calculi

In this section we introduce two basic Markovian process calculi with durational actions. The first one is a sequential Markovian process calculus (SMPC) that generates all the finite CTMCs with as few operators as possible: the null term, the action prefix operator, the alternative composition operator, and recursion. The second one is a concurrent Markovian process calculus (CMPC) as it additionally includes a parallel composition operator governed by an asymmetric action synchronization discipline. Then we introduce some notation concerned with the exit rates of the process terms and the attributes associated with their computations.

2.1 Syntax and Semantics for SMPC

In SMPC every action is durational, hence it is represented as a pair $<a, \lambda>$, where $a \in \mathit{Name}$ is the name of the action while $\lambda \in \mathbb{R}_{>0}$ is the rate of the exponential distribution quantifying the duration of the action. The average duration of an exponentially timed action is equal to the inverse of its rate.

Whenever several exponentially timed actions are enabled, the race policy is adopted, hence the fastest action is the one that is executed. As a consequence of this generative [26] selection mechanism, the execution probability of any enabled exponentially timed action is proportional to its rate and the average sojourn time associated with a process term is exponentially distributed with rate given by the sum of the rates of the actions enabled by the term.

We denote by $\mathit{Act}_{\mathrm{S}} = \mathit{Name} \times \mathbb{R}_{>0}$ the set of the actions of SMPC. Unlike standard process theory, where a distinguished symbol τ is used as the name of the invisible action, here we assume that all the actions are visible.

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

$$\begin{array}{|l} \hline P ::= \underline{0} \\ \quad \mid\ <a, \lambda>.P \\ \quad \mid\ P + P \\ \quad \mid\ X \\ \quad \mid\ \mathit{rec}\, X : P \\ \hline \end{array}$$

where X is a process variable. We denote by $\mathcal{P}_{\mathrm{S}}$ the set of the closed and guarded process terms of SMPC. ■

The semantics for SMPC is given by a state-transition model that can be defined in the usual operational style. However, unlike nondeterministic process calculi, idempotency no longer holds. In fact, a term like $<a, \lambda>.P + <a, \lambda>.P$ is not the same as $<a, \lambda>.P$, as the average sojourn time associated with the latter, i.e. $1/\lambda$, is twice the average sojourn time associated with the former, i.e. $1/(\lambda + \lambda)$. To keep the two terms distinct at the semantic level, it is necessary to take into account the multiplicity of each transition, intended as the number of different proofs for the derivation of the transition.

Therefore, the behavior of each process term $P \in \mathcal{P}_{\mathrm{S}}$ is given by a labeled multitransition system $[\![P]\!]$, whose states correspond to process terms and whose transitions – each of which has a multiplicity – are labeled with actions. From such a labeled multitransition system the CTMC underlying the process term can easily be retrieved by (i) discarding the action names from the transition labels and (ii) collapsing all the transitions between any two states into a single transition whose rate is the sum of the rates of the original transitions.

We now provide the semantic rules for the various operators of SMPC:

- Null term: $\underline{0}$ cannot execute any action, hence the corresponding labeled multitransition system is just a state with no transitions.
- Exponentially timed action prefix: $<a, \lambda>.P$ can execute an action of name a and rate λ and then behaves as P:

$$\boxed{<a, \lambda>.P \stackrel{a,\lambda}{\longrightarrow} P}$$

- Alternative composition: $P_1 + P_2$ behaves as either P_1 or P_2 depending on whether P_1 or P_2 executes an action first:

$$\boxed{\frac{P_1 \stackrel{a,\lambda}{\longrightarrow} P'}{P_1 + P_2 \stackrel{a,\lambda}{\longrightarrow} P'} \qquad \frac{P_2 \stackrel{a,\lambda}{\longrightarrow} P'}{P_1 + P_2 \stackrel{a,\lambda}{\longrightarrow} P'}}$$

- Recursion: $rec\, X : P$ behaves as P after replacing every occurrence of X with $rec\, X : P$:

$$\boxed{\frac{P\{(rec\, X : P)/X\} \stackrel{a,\lambda}{\longrightarrow} P'}{rec\, X : P \stackrel{a,\lambda}{\longrightarrow} P'}}$$

2.2 Syntax and Semantics for CMPC

CMPC extends SMPC with a parallel composition operator governed by an asymmetric action synchronization discipline, which is enforced on an explicit set of action names and makes use of passive actions. Multiway synchronizations are allowed provided that they involve at most one exponentially timed action, with all the other actions being passive.

A passive action is of the form $<a, *_w>$, where $w \in \mathbb{R}_{>0}$ is called weight and is used to quantify choices among passive actions with the same name. Every passive action has a duration that will become specified upon synchronization with an exponentially timed action having the same name.

Whenever several passive actions are enabled, the reactive [26] preselection policy is adopted. This means that, within every set of enabled passive actions with the same name, each such action is given an execution probability proportional to its weight. The choice between two enabled passive actions having different names is instead nondeterministic.

We denote by $Act_{\mathrm{C}} = Name \times Rate$ the set of the actions of CMPC, where $Rate = \mathbb{R}_{>0} \cup \{*_w \mid w \in \mathbb{R}_{>0}\}$ is the set of the action rates (ranged over by $\tilde{\lambda}$). As for SMPC, we assume that all the actions are visible.

Definition 2. *The set $\mathcal{L}_{\rm C}$ of the process terms of CMPC is generated by the following syntax:*

$$\begin{array}{l} P ::= \underline{0} \\ \quad \mid\ <a,\lambda>.P \\ \quad \mid\ <a,*_w>.P \\ \quad \mid\ P + P \\ \quad \mid\ P \parallel_S P \\ \quad \mid\ X \\ \quad \mid\ rec\ X : P \end{array}$$

where $S \subseteq Name$ and X is a process variable. We denote by $\mathcal{P}_{\rm C}$ the set of the closed and guarded process terms of CMPC. ■

Due to the memoryless property of exponential distributions and the fact that the probability that two concurrent exponentially timed actions terminate simultaneously is zero, the semantics for the parallel composition operator can be defined in the usual interleaving style like in the nondeterministic case. In fact, term $<a,\lambda>.\underline{0} \parallel_\emptyset <b,\mu>.\underline{0}$ behaves exactly like term $<a,\lambda>.<b,\mu>.\underline{0} + <b,\mu>.<a,\lambda>.\underline{0}$ as the execution of an exponentially timed action can be thought of as being started in the last state in which the action is enabled.

We now provide the semantic rules for the additional operators of CMPC:

– Passive action prefix: $<a,*_w>.P$ can execute a passive action of name a and weight w and then behaves as P:

$$<a,*_w>.P \stackrel{a,*_w}{\longrightarrow} P$$

– Parallel composition: $P_1 \parallel_S P_2$ behaves as P_1 in parallel with P_2 as long as actions are executed whose names do not belong to S:

$$\frac{P_1 \stackrel{a,\tilde{\lambda}}{\longrightarrow} P_1' \quad a \notin S}{P_1 \parallel_S P_2 \stackrel{a,\tilde{\lambda}}{\longrightarrow} P_1' \parallel_S P_2} \qquad \frac{P_2 \stackrel{a,\tilde{\lambda}}{\longrightarrow} P_2' \quad a \notin S}{P_1 \parallel_S P_2 \stackrel{a,\tilde{\lambda}}{\longrightarrow} P_1 \parallel_S P_2'}$$

Generative-reactive synchronizations are forced between any exponentially timed action executed by one term and any passive action executed by the other term that have the same name belonging to S:

$$\frac{P_1 \stackrel{a,\lambda}{\longrightarrow} P_1' \quad P_2 \stackrel{a,*_w}{\longrightarrow} P_2' \quad a \in S}{P_1 \parallel_S P_2 \stackrel{a,\lambda \cdot \frac{w}{weight(P_2,a)}}{\longrightarrow} P_1' \parallel_S P_2'}$$

$$\frac{P_1 \stackrel{a,*_w}{\longrightarrow} P_1' \quad P_2 \stackrel{a,\lambda}{\longrightarrow} P_2' \quad a \in S}{P_1 \parallel_S P_2 \stackrel{a,\lambda \cdot \frac{w}{weight(P_1,a)}}{\longrightarrow} P_1' \parallel_S P_2'}$$

while reactive-reactive synchronizations are forced between any two passive actions executed by the two terms that have the same name belonging to S:

$$\frac{P_1 \stackrel{a,*_{w_1}}{\longrightarrow} P_1' \qquad P_2 \stackrel{a,*_{w_2}}{\longrightarrow} P_2' \qquad a \in S}{P_1 \parallel_S P_2 \stackrel{a,* \frac{w_1}{weight(P_1,a)} \cdot \frac{w_2}{weight(P_2,a)} \cdot (weight(P_1,a)+weight(P_2,a))}{\longrightarrow} P_1' \parallel_S P_2'}$$

where the weight of a process term P with respect to the passive actions of name a that P enables is defined as follows:

$$weight(P,a) = \sum\!\!\!\!\!\!\{\!\!\{ w \in \mathbb{R}_{>0} \mid \exists P' \in \mathcal{P}_{\rm C}.\, P \stackrel{a,*_w}{\longrightarrow} P' \}\!\!\}$$

We point out that the CTMC underlying a process term in $\mathcal{P}_{\rm C}$ can be retrieved only if its labeled multitransition system has no passive transitions. In this case we say that the process term is performance closed. We denote by $\mathcal{P}_{\rm C,pc}$ the set of the performance closed process terms of $\mathcal{P}_{\rm C}$. Note that $\mathcal{P}_{\rm S,pc} = \mathcal{P}_{\rm S}$.

2.3 Exit Rates and Computations of Process Terms

The Markovian behavioral equivalences that we shall define over SMPC and CMPC are based on concepts like the exit rates of the process terms and the traces, the probabilities, and the durations of their computations. Since these concepts will be used several times in the paper, we collect in this section the related notation.

The exit rate of a process term is the rate at which it is possible to leave the term. We distinguish among the rate at which the process term can execute actions of a given name that lead to a given set of terms, the total rate at which the process term can execute actions of a given name, and the total exit rate of the process term. The latter is the sum of the rates of all the actions that the process term can execute, and coincides with the reciprocal of the average sojourn time in the CTMC-level state corresponding to the process term whenever the process term is performance closed.

Since there are two kinds of actions – exponentially timed and passive – we consider a two-level definition of each variant of exit rate, where level 0 corresponds to exponentially timed actions and level -1 corresponds to passive actions.

Definition 3. *Let $P \in \mathcal{P}_{\rm C}$, $a \in Name$, $l \in \{0,-1\}$, and $C \subseteq \mathcal{P}_{\rm C}$. The exit rate of P when executing actions of name a and level l that lead to C is defined through the following non-negative real function:*

$$rate(P,a,l,C) = \begin{cases} \sum\!\!\!\!\!\!\{\!\!\{ \lambda \in \mathbb{R}_{>0} \mid \exists P' \in C.\, P \stackrel{a,\lambda}{\longrightarrow} P' \}\!\!\} & \text{if } l = 0 \\ \sum\!\!\!\!\!\!\{\!\!\{ w \in \mathbb{R}_{>0} \mid \exists P' \in C.\, P \stackrel{a,*_w}{\longrightarrow} P' \}\!\!\} & \text{if } l = -1 \end{cases}$$

where each summation is taken to be zero whenever its multiset is empty. ■

Definition 4. *Let $P \in \mathcal{P}_{\mathrm{C}}$ and $l \in \{0, -1\}$. The total exit rate of P at level l is defined through the following non-negative real function:*

$$\boxed{rate_{\mathrm{t}}(P, l) = \sum_{a \in Name} rate(P, a, l, \mathcal{P}_{\mathrm{C}})}$$

where $rate(P, a, l, \mathcal{P}_{\mathrm{C}})$ is the total exit rate of P with respect to a at level l. ■

A computation of a process term is a sequence of transitions that can be executed starting from the state corresponding to the term. The length of a computation is given by the number of transitions occurring in it. We say that two computations are independent of each other if it is not the case that one of them is a proper prefix of the other one. In the following, we denote by $\mathcal{C}_{\mathrm{f}}(P)$ and $\mathcal{I}_{\mathrm{f}}(P)$ the multisets of the finite-length computations and independent computations of $P \in \mathcal{P}_{\mathrm{C}}$. Below we inductively define the trace, the execution probability, and the duration of an element of $\mathcal{C}_{\mathrm{f}}(P)$, using symbol "$\circ$" to denote the sequence concatenation operator.

Definition 5. *Let $P \in \mathcal{P}_{\mathrm{C}}$ and $c \in \mathcal{C}_{\mathrm{f}}(P)$. The trace associated with the execution of c is the sequence of the action names labeling the transitions of c, which is defined by induction on the length of c through the following $Name^*$-valued function:*

$$\boxed{trace(c) = \begin{cases} \varepsilon & \text{if } length(c) = 0 \\ a \circ trace(c') & \text{if } c \equiv P \xrightarrow{a,\tilde{\lambda}} c' \end{cases}}$$

where ε is the empty trace. ■

Definition 6. *Let $P \in \mathcal{P}_{\mathrm{C,pc}}$ and $c \in \mathcal{C}_{\mathrm{f}}(P)$. The probability of executing c is the product of the execution probabilities of the transitions of c, which is defined by induction on the length of c through the following $\mathbf{R}_{]0,1]}$-valued function:*

$$\boxed{prob(c) = \begin{cases} 1 & \text{if } length(c) = 0 \\ \frac{\lambda}{rate_{\mathrm{t}}(P,0)} \cdot prob(c') & \text{if } c \equiv P \xrightarrow{a,\lambda} c' \end{cases}}$$

We also define the probability of executing a computation of C as:

$$\boxed{prob(C) = \sum_{c \in C} prob(c)}$$

for all $C \subseteq \mathcal{I}_{\mathrm{f}}(P)$. ■

Definition 7. *Let $P \in \mathcal{P}_{\mathrm{C,pc}}$ and $c \in \mathcal{C}_{\mathrm{f}}(P)$. The stepwise average duration of c is the sequence of the average sojourn times in the states traversed by c, which is defined by induction on the length of c through the following $(\mathbf{R}_{>0})^*$-valued function:*

$$\boxed{time_{\mathrm{a}}(c) = \begin{cases} \varepsilon & \text{if } length(c) = 0 \\ \frac{1}{rate_{\mathrm{t}}(P,0)} \circ time_{\mathrm{a}}(c') & \text{if } c \equiv P \xrightarrow{a,\lambda} c' \end{cases}}$$

where ε is the empty stepwise average duration. We also define the multiset of the computations of C whose stepwise average duration is not greater than θ as:

$$C_{\leq\theta} = \{\!|\, c \in C \mid length(c) \leq length(\theta) \wedge \forall i = 1, \ldots, length(c).\, time_{\rm a}(c)[i] \leq \theta[i] \,|\!\}$$

for all $C \subseteq \mathcal{C}_{\rm f}(P)$ and $\theta \in (\mathbb{R}_{>0})^*$. ■

Definition 8. *Let $P \in \mathcal{P}_{\rm C,pc}$ and $c \in \mathcal{C}_{\rm f}(P)$. The stepwise duration of c is the sequence of the random variables quantifying the sojourn times in the states traversed by c, which is defined by induction on the length of c through the following random-variable-sequence-valued function:*

$$time_{\rm d}(c) = \begin{cases} \varepsilon & \text{if } length(c) = 0 \\ Exp_{rate_{\rm t}(P,0)} \circ time_{\rm d}(c') & \text{if } c \equiv P \xrightarrow{a,\lambda} c' \end{cases}$$

where ε is the empty stepwise duration while $Exp_{rate_{\rm t}(P,0)}$ is the exponentially distributed random variable with rate $rate_{\rm t}(P,0) \in \mathbb{R}_{>0}$. ■

Definition 9. *Let $P \in \mathcal{P}_{\rm C,pc}$, $C \subseteq \mathcal{I}_{\rm f}(P)$, and $\theta \in (\mathbb{R}_{>0})^*$. The probability distribution of executing a computation of C within a sequence θ of time units is given by:*

$$prob_{\rm d}(C,\theta) = \sum_{c \in C}^{length(c) \leq length(\theta)} prob(c) \cdot \prod_{i=1}^{length(c)} \Pr(time_{\rm d}(c)[i] \leq \theta[i])$$

where $\Pr(time_{\rm d}(c)[i] \leq \theta[i]) = 1 - {\rm e}^{-\theta[i]/time_{\rm a}(c)[i]}$ is the cumulative distribution function of the exponentially distributed random variable $time_{\rm d}(c)[i]$, whose expected value is $time_{\rm a}(c)[i]$. ■

We conclude by observing that the average duration (resp. duration) of a finite-length computation has been defined as the sequence of the average sojourn times (resp. of the random variables quantifying the sojourn times) in the states traversed by the computation. The same quantity could have been defined as the sum of the same basic ingredients, but this would not have been appropriate.

Example 1. Consider the two following process terms:

$$<g,\gamma>.<a,\lambda>.<b,\mu>.\underline{0} + <g,\gamma>.<a,\mu>.<d,\lambda>.\underline{0}$$
$$<g,\gamma>.<a,\lambda>.<d,\mu>.\underline{0} + <g,\gamma>.<a,\mu>.<b,\lambda>.\underline{0}$$

with $\lambda \neq \mu$ and $b \neq d$. Observed that the two terms have identical non-maximal computations, we further notice that the first term has the two following maximal computations each with probability 1/2:

$$c_{1,1} \equiv . \xrightarrow{g,\gamma} . \xrightarrow{a,\lambda} . \xrightarrow{b,\mu} .$$
$$c_{1,2} \equiv . \xrightarrow{g,\gamma} . \xrightarrow{a,\mu} . \xrightarrow{d,\lambda} .$$

while the second term has the two following maximal computations each with probability 1/2:

$$c_{2,1} \equiv . \xrightarrow{g,\gamma} . \xrightarrow{a,\lambda} . \xrightarrow{d,\mu} .$$
$$c_{2,2} \equiv . \xrightarrow{g,\gamma} . \xrightarrow{a,\mu} . \xrightarrow{b,\lambda} .$$

If the average duration were defined as the sum of the average sojourn times, then $c_{1,1}$ and $c_{2,2}$ would have the same trace $g \circ a \circ b$ and the same average duration $\frac{1}{2 \cdot \gamma} + \frac{1}{\lambda} + \frac{1}{\mu}$, and similarly $c_{1,2}$ and $c_{2,1}$ would have the same trace $g \circ a \circ d$ and the same average duration $\frac{1}{2 \cdot \gamma} + \frac{1}{\mu} + \frac{1}{\lambda}$. This would lead to conclude that the two terms are equivalent, whereas an external observer equipped with a button-pushing machine displaying the names of the actions that are performed and the instants at which they are performed [49] would be able to distinguish between the two terms. ■

3 Markovian Bisimilarity

Markovian bisimilarity considers two process terms to be equivalent whenever they are able to mimic each other's functional and performance behavior stepwise. In this section we provide the definition of Markovian bisimilarity over $\mathcal{P}_{\mathrm{C}}$ and we recall its properties from [32, 15, 31, 14, 18, 4, 23].

3.1 Equivalence Definition

The basic idea behind Markovian bisimilarity is that, whenever a process term can perform actions with a certain name that reach a certain set of terms at a certain speed, then any process term equivalent to the given one has to be able to respond with actions with the same name that reach an equivalent set of terms at the same speed. This can be formalized through the comparison of the process term exit rates when executing actions of the same name (and level) that lead to the same set of equivalent terms.

Definition 10. *An equivalence relation $\mathcal{B} \subseteq \mathcal{P}_{\mathrm{C}} \times \mathcal{P}_{\mathrm{C}}$ is a Markovian bisimulation iff, whenever $(P_1, P_2) \in \mathcal{B}$, then for all action names $a \in Name$, levels $l \in \{0, -1\}$, and equivalence classes $C \in \mathcal{P}_{\mathrm{C}}/\mathcal{B}$:*

$$rate(P_1, a, l, C) = rate(P_2, a, l, C)$$ ■

Since the union of all the Markovian bisimulations can be proved to be the largest Markovian bisimulation, the definition below follows.

Definition 11. *Markovian bisimilarity, denoted by $\sim_{\mathrm{MB}}$, is the union of all the Markovian bisimulations.* ■

Obviously, $\sim_{\mathrm{MB}}$ is strictly finer than classical bisimilarity [43, 41] and probabilistic bisimilarity [40]. We conclude with an easy-to-check necessary condition.

Proposition 1. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C}}$. Whenever $P_1 \sim_{\mathrm{MB}} P_2$, then for all $a \in Name$ and $l \in \{0, -1\}$:*

$$rate(P_1, a, l, \mathcal{P}_{\mathrm{C}}) = rate(P_2, a, l, \mathcal{P}_{\mathrm{C}})$$ ■

3.2 Exactness

Markovian bisimilarity is consistent with an exact aggregation for CTMCs that is known under the name of ordinary lumping.

Definition 12. *A partition $\mathcal{O}$ of the state space of a CTMC is an ordinary lumping iff, whenever $s_1, s_2 \in O$ for some $O \in \mathcal{O}$, then for all $O' \in \mathcal{O}$:*

$$\sum\{\!|\, \lambda \in \mathbb{R}_{>0} \mid \exists s' \in O'.\, s_1 \xrightarrow{\lambda} s' \,|\!\} = \sum\{\!|\, \lambda \in \mathbb{R}_{>0} \mid \exists s' \in O'.\, s_2 \xrightarrow{\lambda} s' \,|\!\}$$ ■

Theorem 1. *The CTMC-level aggregation induced by $\sim_{\rm MB}$ is an ordinary lumping.* ■

Corollary 1. *The CTMC-level aggregation induced by $\sim_{\rm MB}$ is exact.* ■

3.3 Congruence

Markovian bisimilarity is a congruence with respect to all the operators of CMPC.

Theorem 2. *Let $P_1, P_2 \in \mathcal{P}_{\rm C}$. Whenever $P_1 \sim_{\rm MB} P_2$, then:*

1. *$<a, \tilde{\lambda}>.P_1 \sim_{\rm MB} <a, \tilde{\lambda}>.P_2$ for all $<a, \tilde{\lambda}> \in Act_{\rm C}$.*
2. *$P_1 + P \sim_{\rm MB} P_2 + P$ and $P + P_1 \sim_{\rm MB} P + P_2$ for all $P \in \mathcal{P}_{\rm C}$.*
3. *$P_1 \parallel_S P \sim_{\rm MB} P_2 \parallel_S P$ and $P \parallel_S P_1 \sim_{\rm MB} P \parallel_S P_2$ for all $S \subseteq Name$ and $P \in \mathcal{P}_{\rm C}$.* ■

As far as recursion is concerned, we need to extend $\sim_{\rm MB}$ to open process terms. These are terms containing free process variables, i.e. variables not occurring within the scope of a *rec* binder.

Definition 13. *Let $P_1, P_2 \in \mathcal{L}_{\rm C}$ containing a free process variable X. We define $P_1 \sim_{\rm MB} P_2$ iff $P_1\{Q/X\} \sim_{\rm MB} P_2\{Q/X\}$ for all $Q \in \mathcal{P}_{\rm C}$.* ■

Theorem 3. *Let $P_1, P_2 \in \mathcal{L}_{\rm C}$ containing a free process variable X. Whenever $P_1 \sim_{\rm MB} P_2$, then $rec\, X : P_1 \sim_{\rm MB} rec\, X : P_2$.* ■

3.4 Axiomatization

Markovian bisimilarity has a sound and complete axiomatization over $\mathcal{P}_{\rm C}$, whose specific axioms are shown Table 1. This includes three basic laws for alternative composition, two laws characterizing the race policy and the reactive preselection policy, an expansion law for parallel composition, and three laws for recursion. As far as $\mathcal{A}_6^{\rm MB}$ is concerned, I and J are finite index sets (if empty, the related summations are taken to be $\underline{0}$). The validity of this law is a consequence of the memoryless property of the exponentially distributed durations and of the fact that the probability that two concurrent exponentially timed actions terminate simultaneously is zero, which allows the semantics for the parallel composition operator to be defined in the usual interleaving style.

Theorem 4. *The deduction system $DED(\mathcal{A}^{\rm MB})$ is sound and complete for $\sim_{\rm MB}$ over $\mathcal{P}_{\rm C}$, i.e. for all $P_1, P_2 \in \mathcal{P}_{\rm C}$:*

$$P_1 \sim_{\rm MB} P_2 \iff \mathcal{A}^{\rm MB} \vdash P_1 = P_2$$ ■

$$
\begin{array}{ll}
(\mathcal{A}_1^{\mathrm{MB}}) & P_1 + P_2 = P_2 + P_1 \\
(\mathcal{A}_2^{\mathrm{MB}}) & (P_1 + P_2) + P_3 = P_1 + (P_2 + P_3) \\
(\mathcal{A}_3^{\mathrm{MB}}) & P + \underline{0} = P \\
(\mathcal{A}_4^{\mathrm{MB}}) & <a, \lambda_1>.P + <a, \lambda_2>.P = <a, \lambda_1 + \lambda_2>.P \\
(\mathcal{A}_5^{\mathrm{MB}}) & <a, *_{w_1}>.P + <a, *_{w_2}>.P = <a, *_{w_1 + w_2}>.P \\
(\mathcal{A}_6^{\mathrm{MB}}) & \sum_{i \in I} <a_i, \tilde{\lambda}_i>.P_{1,i} \parallel_S \sum_{j \in J} <b_j, \tilde{\mu}_j>.P_{2,j} = \\
& \quad \sum_{k \in I, a_k \notin S} <a_k, \tilde{\lambda}_k>.\left(P_{1,k} \parallel_S \sum_{j \in J} <b_j, \tilde{\mu}_j>.P_{2,j} \right) + \\
& \quad \sum_{h \in J, b_h \notin S} <b_h, \tilde{\mu}_h>.\left(\sum_{i \in I} <a_i, \tilde{\lambda}_i>.P_{1,i} \parallel_S P_{2,h} \right) + \\
& \quad \sum_{k \in I, a_k \in S, \tilde{\lambda}_k \in \mathbb{R}_{>0}} \; \sum_{h \in J, b_h = a_k, \tilde{\mu}_h = *_{w_h}} <a_k, \tilde{\lambda}_k \cdot \frac{w_h}{weight(P_2, b_h)}>.(P_{1,k} \parallel_S P_{2,h}) + \\
& \quad \sum_{h \in J, b_h \in S, \tilde{\mu}_h \in \mathbb{R}_{>0}} \; \sum_{k \in I, a_k = b_h, \tilde{\lambda}_k = *_{v_k}} <b_h, \tilde{\mu}_h \cdot \frac{v_k}{weight(P_1, a_k)}>.(P_{1,k} \parallel_S P_{2,h}) + \\
& \quad \sum_{k \in I, a_k \in S, \tilde{\lambda}_k = *_{v_k}} \; \sum_{h \in J, b_h = a_k, \tilde{\mu}_h = *_{w_h}} \\
& \qquad <a_k, *_{\frac{v_k}{weight(P_1, a_k)} \cdot \frac{w_h}{weight(P_2, b_h)} \cdot (weight(P_1, a_k) + weight(P_2, b_h))}>.(P_{1,k} \parallel_S P_{2,h}) \\
(\mathcal{A}_7^{\mathrm{MB}}) & rec\, X : P = rec\, Y : P\{Y/X\} \qquad \text{if } Y \text{ not in } P \\
(\mathcal{A}_8^{\mathrm{MB}}) & rec\, X : P = P\{(rec\, X : P)/X\} \\
(\mathcal{A}_9^{\mathrm{MB}}) & Q = P\{Q/X\} \Rightarrow Q = rec\, X : P
\end{array}
$$

Table 1. Axiomatization of $\sim_{\mathrm{MB}}$ over $\mathcal{P}_{\mathrm{C}}$

3.5 Logical Characterization

Markovian bisimilarity has a modal logic characterization over $\mathcal{P}_{\mathrm{C,pc}}$ based on a Markovian variant of the Hennessy-Milner logic [29], in which the diamond operator is decorated with a rate lower bound.

Definition 14. *The set of the formulas of* $\mathrm{HML}_{\mathrm{MB}}$ *is generated by the following syntax:*

$$\boxed{\phi ::= \mathrm{true} \mid \neg\phi \mid \phi \wedge \phi \mid \nabla_a \mid \langle a \rangle_\lambda \phi}$$

where $a \in Name$ *and* $\lambda \in \mathbb{R}_{>0}$. ■

Definition 15. *The satisfaction relation* $\models_{\mathrm{MB}}$ *of* $\mathrm{HML}_{\mathrm{MB}}$ *over* $\mathcal{P}_{\mathrm{C,pc}}$ *is defined by structural induction as follows:*

$$
\begin{array}{|ll|}
\hline
P \models_{\mathrm{MB}} \mathrm{true} & \\
P \models_{\mathrm{MB}} \neg\phi & \text{if } P \not\models_{\mathrm{MB}} \phi \\
P \models_{\mathrm{MB}} \phi_1 \wedge \phi_2 & \text{if } P \models_{\mathrm{MB}} \phi_1 \text{ and } P \models_{\mathrm{MB}} \phi_2 \\
P \models_{\mathrm{MB}} \nabla_a & \text{if } rate(P, a, 0, \mathcal{P}_{\mathrm{C}}) = 0 \\
P \models_{\mathrm{MB}} \langle a \rangle_\lambda \phi & \text{if } rate(P, a, 0, sat(\phi)) \geq \lambda \\
\hline
\end{array}
$$

where $sat(\phi) = \{P' \in \mathcal{P}_{\mathrm{C}} \mid P' \models_{\mathrm{MB}} \phi\}$. ■

Theorem 5. *Let* $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. *Then:*

$$P_1 \sim_{\mathrm{MB}} P_2 \iff (\forall \phi \in \mathrm{HML}_{\mathrm{MB}}.\, P_1 \models_{\mathrm{MB}} \phi \iff P_2 \models_{\mathrm{MB}} \phi) \qquad \blacksquare$$

We also mention that Markovian bisimilarity has a temporal logic characterization based on the branching-time logic CSL [3]. Besides the usual propositional connectives, this logic comprises:

- a probability operator, which replaces the universal and existential computation quantifiers and allows to refer to the probability of performing a computation that satisfies a certain formula;
- a time-bounded next operator, which expresses that the next state is reached within a certain amount of time and a certain formula holds in it;
- a time-bounded until operator, which expresses that a certain formula is satisfied at some instant no later than a certain amount of time and at all preceding instants another given formula holds;
- a steady-state operator, which enables to reason about the probability to be in the long run in some state that satisfies a certain formula.

3.6 Verification Complexity

Markovian bisimilarity can be decided in polynomial time by means of a partition refinement algorithm in the style of [42]. This algorithm exploits the fact that $\sim_{\mathrm{MB}}$ can be characterized as a fixed point of successively finer relations. In fact we have:

$$\sim_{\mathrm{MB}} = \bigcap_{i \in \mathbf{N}} \sim_{\mathrm{MB},i}$$

where $\sim_{\mathrm{MB},0} = \mathcal{P}_{\mathrm{C}} \times \mathcal{P}_{\mathrm{C}}$ and $\sim_{\mathrm{MB},i}$ for $i \geq 1$ is defined as follows: whenever $(P_1, P_2) \in \sim_{\mathrm{MB},i}$, then for all $a \in \mathit{Name}$, $l \in \{0, -1\}$, and $C \in \mathcal{P}_{\mathrm{C}} / \sim_{\mathrm{MB},i-1}$:

$$\mathit{rate}(P_1, a, l, C) = \mathit{rate}(P_2, a, l, C)$$

In other words, $\sim_{\mathrm{MB},0}$ induces a trivial partition with a single equivalence class that coincides with $\mathcal{P}_{\mathrm{C}}$, $\sim_{\mathrm{MB},1}$ refines the previous partition by creating an equivalence class for each set of terms that satisfy the necessary condition stated by Prop. 1, and so on.

The algorithm to check whether $P_1 \sim_{\mathrm{MB}} P_2$ thus proceeds as follows:

1. Build a partition with a single class including all the states of the disjoint union of $[\![P_1]\!]$ and $[\![P_2]\!]$, then initialize a list of splitters with this class.
2. Refine the current partition by splitting each of its classes according to the exit rates towards one of the splitters, then remove this splitter from the list.
3. For each split class, insert into the list of splitters all the resulting subclasses except for the largest one.
4. If the list of splitters is empty, return yes/no depending on whether the initial state of $[\![P_1]\!]$ and the initial state of $[\![P_2]\!]$ belong to the same class or not, otherwise go back to the refinement step.

The time complexity is $O(m \cdot \log n)$, where n is the number of states and m is the number of transitions of the disjoint union of $[\![P_1]\!]$ and $[\![P_2]\!]$. To achieve this complexity it is necessary to resort to a splay tree when representing the subclasses arising from the splitting of a class.

4 Variants of Markovian Bisimilarity

Due to the nice properties presented in the previous section, Markovian bisimilarity has received more attention than Markovian testing and trace equivalences. For this reason, some useful variants of it have appeared in the literature, three of which will be recalled in this section.

The first one – reward Markovian bisimilarity [10] – takes rewards into account in order to allow process terms to be compositionally manipulated in a way that is sensitive to specific performance measures. The second one – interactive Markovian bisimilarity [30] – stems from the interaction of nondeterministic process calculi and CTMCs and deals with both Markovian branchings and nondeterministic branchings. The third one – extended Markovian bisimilarity [8] – considers immediate actions à la GSPN [1] and deals with both Markovian branchings and prioritized/probabilistic branchings.

4.1 Reward Markovian Bisimilarity

Although Markovian bisimilarity is consistent with ordinary lumping, which is an exact aggregation, it may happen that specific performance measures distinguish between ordinarily lumpable states by ascribing them a different meaning.

The usual approach to the specification of performance measures for CTMC-based models relies on reward structures [34]. This requires associating real numbers with model behaviors and activities, which are then transferred to the proper states (as yield rewards) and transitions (as bonus rewards) of the underlying CTMC, respectively. A yield reward expresses the rate at which a gain (or a loss, if the number is negative) is accumulated while sojourning in the related state. By contrast, a bonus reward specifies the instantaneous gain (or loss) implied by the execution of the related transition.

The instant-of-time value of a performance measure specified through a reward structure is computed as follows for a CTMC:

$$\boxed{\sum_{s} yr(s) \cdot \pi(s) + \sum_{s \xrightarrow{\lambda} s'} br(s, \lambda, s') \cdot \phi(s, \lambda, s')}$$

where:

- $yr(s)$ is the yield reward associated with state s.
- $\pi(s)$ is the probability of being in state s at the considered instant of time.
- $br(s, \lambda, s')$ is the bonus reward associated with transition $s \xrightarrow{\lambda} s'$.
- $\phi(s, \lambda, s')$ is the frequency of transition $s \xrightarrow{\lambda} s'$ at the considered instant of time, which is given by $\pi(s) \cdot \lambda$.

In this setting, ascribing a different meaning to ordinarily lumpable states amounts to giving different rewards to such states or their outgoing transitions. In order to manipulate process terms in a performance-sensitive way, the definition of Markovian bisimilarity can be modified by taking rewards into account.

Before doing that, we need to extend CMPC with rewards. While bonus rewards can naturally be associated with actions, the treatment of yield rewards is not trivial because in process calculi the concept of state is implicit. An approach that can be followed is to associate yield rewards with actions too, with the yield reward of a state being given by the sum of the yield rewards associated with the actions enabled in that state (additivity assumption).

Therefore, we derive from CMPC a new calculus, which we call concurrent reward Markovian process calculus (CRMPC):

– The syntax is extended as follows:

$$\begin{array}{l} P ::= \underline{0} \\ \quad \mid \ <a, \lambda, yr, br>.P \\ \quad \mid \ <a, *_w, *, *>.P \\ \quad \mid \ P + P \\ \quad \mid \ P \parallel_S P \\ \quad \mid \ X \\ \quad \mid \ rec\, X : P \end{array}$$

where $yr \in \mathbb{R}$ is the yield reward and $br \in \mathbb{R}$ is the bonus reward. We denote by $\mathcal{P}_{\mathrm{CR}}$ the set of the closed and guarded process terms of CRMPC.

– The semantic rules are modified accordingly. In particular, the synchronization rules change as follows:

$$\frac{P_1 \xrightarrow{a,\lambda,yr,br} P_1' \qquad P_2 \xrightarrow{a,*_w,*,*} P_2' \qquad a \in S}{P_1 \parallel_S P_2 \xrightarrow{a,\lambda \cdot \frac{w}{weight(P_2,a)}, yr \cdot \frac{w}{weight(P_2,a)}, br} P_1' \parallel_S P_2'}$$

$$\frac{P_1 \xrightarrow{a,*_w,*,*} P_1' \qquad P_2 \xrightarrow{a,\lambda,yr,br} P_2' \qquad a \in S}{P_1 \parallel_S P_2 \xrightarrow{a,\lambda \cdot \frac{w}{weight(P_1,a)}, yr \cdot \frac{w}{weight(P_1,a)}, br} P_1' \parallel_S P_2'}$$

$$\frac{P_1 \xrightarrow{a,*_{w_1},*,*} P_1' \qquad P_2 \xrightarrow{a,*_{w_2},*,*} P_2' \qquad a \in S}{P_1 \parallel_S P_2 \xrightarrow{a,* \frac{w_1}{weight(P_1,a)} \cdot \frac{w_2}{weight(P_2,a)} \cdot (weight(P_1,a) + weight(P_2,a)),*,*} P_1' \parallel_S P_2'}$$

Note that the yield rewards are subject to the same normalization as the rates, because they are strictly related to the sojourn time in the states. By contrast, no normalization is needed for the bonus rewards, as they are earned upon the execution of the transitions.

We now introduce the concept of exit reward, on the basis of which we shall define the performance-sensitive variant of Markovian bisimilarity.

Definition 16. *Let $P \in \mathcal{P}_{\mathrm{CR}}$, $a \in Name$, $l \in \{0, -1\}$, and $C \subseteq \mathcal{P}_{\mathrm{CR}}$. The exit reward of P when executing actions of name a and level l that lead to C is*

defined through the following real function:

$$reward(P,a,l,C) = \begin{cases} \sum\{\!\!\{\, yr+\lambda\cdot br \in \mathbb{R} \mid \exists P' \in C.\, P \xrightarrow{a,\lambda,yr,br} P' \,\}\!\!\} & \text{if } l = 0 \\ 0 & \text{if } l = -1 \end{cases}$$

where the summation is taken to be zero whenever its multiset is empty. ■

Definition 17. *An equivalence relation $\mathcal{B} \subseteq \mathcal{P}_{\mathrm{CR}} \times \mathcal{P}_{\mathrm{CR}}$ is a reward Markovian bisimulation iff, whenever $(P_1, P_2) \in \mathcal{B}$, then for all action names $a \in Name$, levels $l \in \{0, -1\}$, and equivalence classes $C \in \mathcal{P}_{\mathrm{CR}}/\mathcal{B}$:*

$$rate(P_1,a,l,C) = rate(P_2,a,l,C)$$
$$reward(P_1,a,l,C) = reward(P_2,a,l,C)$$

■

Definition 18. *Reward Markovian bisimilarity, denoted by $\sim_{\mathrm{RMB}}$, is the union of all the reward Markovian bisimulations.* ■

$\sim_{\mathrm{RMB}}$ enjoys the same properties as $\sim_{\mathrm{MB}}$. The characterizing axioms are:

$$\begin{array}{c} <a,\lambda_1,yr_1,br_1>.P + <a,\lambda_2,yr_2,br_2>.P = \\ <a,\lambda_1+\lambda_2,yr_1+yr_2,\frac{\lambda_1}{\lambda_1+\lambda_2}\cdot br_1 + \frac{\lambda_2}{\lambda_1+\lambda_2}\cdot br_2>.P \\ <a,*_{w_1},*,*>.P + <a,*_{w_2},*,*>.P = <a,*_{w_1+w_2},*,*>.P \end{array}$$

or equivalently:

$$\begin{array}{c} <a,\lambda,yr,br>.P = <a,\lambda,yr+\lambda\cdot br,0>.P \\ <a,\lambda_1,yr_1,0>.P + <a,\lambda_2,yr_2,0>.P = <a,\lambda_1+\lambda_2,yr_1+yr_2,0>.P \\ <a,*_{w_1},*,*>.P + <a,*_{w_2},*,*>.P = <a,*_{w_1+w_2},*,*>.P \end{array}$$

or equivalently:

$$\begin{array}{c} <a,\lambda,yr,br>.P = <a,\lambda,0,br+\frac{yr}{\lambda}>.P \\ <a,\lambda_1,0,br_1>.P + <a,\lambda_2,0,br_2>.P = <a,\lambda_1+\lambda_2,0,\frac{\lambda_1}{\lambda_1+\lambda_2}\cdot br_1 + \frac{\lambda_2}{\lambda_1+\lambda_2}\cdot br_2>.P \\ <a,*_{w_1},*,*>.P + <a,*_{w_2},*,*>.P = <a,*_{w_1+w_2},*,*>.P \end{array}$$

4.2 Interactive Markovian Bisimilarity

Markovian bisimilarity deals with fully probabilistic models. However, it may happen that not all the details of a model are known in the early design stages. In that case, nondeterminism is a useful abstraction.

A way to recover nondeterminism in a Markovian framework is to combine nondeterministic process calculi and CTMCs. This can be accomplished by replacing durational actions with two distinct prefix operators – one for interacting actions that take no time and one for exponential delays – with the inter-process communication being implemented through the synchronization of visible interacting actions. Since an invisible action – which we denote by τ as usual – takes no time and cannot be prevented by any synchronization constraint, we assume maximal progress, i.e. τ-actions take precedence over time passing.

We now derive from CMPC a new calculus, which we call concurrent interactive Markovian process calculus (CIMPC):

– The syntax is modified as follows:

$$\boxed{\begin{array}{l} P ::= \underline{0} \\ \quad \mid\ a.P \\ \quad \mid\ (\lambda).P \\ \quad \mid\ P + P \\ \quad \mid\ P \parallel_S P \\ \quad \mid\ X \\ \quad \mid\ \mathit{rec}\, X : P \end{array}}$$

where $a \in \mathit{Name} \cup \{\tau\}$. We denote by $\mathcal{P}_{\mathrm{CI}}$ the set of the closed and guarded process terms of CIMPC.

– Two transition relations are necessary: one for interacting actions and one for exponential delays. Besides handling recursion, the semantic rules for interacting actions include:

$$\boxed{\begin{array}{c} a.P \stackrel{a}{\longrightarrow}_{\mathrm{I}} P \\[2ex] \dfrac{P_1 \stackrel{a}{\longrightarrow}_{\mathrm{I}} P'}{P_1 + P_2 \stackrel{a}{\longrightarrow}_{\mathrm{I}} P'} \qquad \dfrac{P_2 \stackrel{a}{\longrightarrow}_{\mathrm{I}} P'}{P_1 + P_2 \stackrel{a}{\longrightarrow}_{\mathrm{I}} P'} \\[2ex] \dfrac{P_1 \stackrel{a}{\longrightarrow}_{\mathrm{I}} P_1' \quad a \notin S}{P_1 \parallel_S P_2 \stackrel{a}{\longrightarrow}_{\mathrm{I}} P_1' \parallel_S P_2} \qquad \dfrac{P_2 \stackrel{a}{\longrightarrow}_{\mathrm{I}} P_2' \quad a \notin S}{P_1 \parallel_S P_2 \stackrel{a}{\longrightarrow}_{\mathrm{I}} P_1 \parallel_S P_2'} \\[2ex] \dfrac{P_1 \stackrel{a}{\longrightarrow}_{\mathrm{I}} P_1' \quad P_2 \stackrel{a}{\longrightarrow}_{\mathrm{I}} P_2' \quad a \in S}{P_1 \parallel_S P_2 \stackrel{a}{\longrightarrow}_{\mathrm{I}} P_1' \parallel_S P_2'} \end{array}}$$

while the semantic rules for exponential delays include:

$$\boxed{\begin{array}{c} (\lambda).P \stackrel{\lambda}{\longrightarrow}_{\mathrm{M}} P \\[2ex] \dfrac{P_1 \stackrel{\lambda}{\longrightarrow}_{\mathrm{M}} P'}{P_1 + P_2 \stackrel{\lambda}{\longrightarrow}_{\mathrm{M}} P'} \qquad \dfrac{P_2 \stackrel{\lambda}{\longrightarrow}_{\mathrm{M}} P'}{P_1 + P_2 \stackrel{\lambda}{\longrightarrow}_{\mathrm{M}} P'} \\[2ex] \dfrac{P_1 \stackrel{\lambda}{\longrightarrow}_{\mathrm{M}} P_1'}{P_1 \parallel_S P_2 \stackrel{\lambda}{\longrightarrow}_{\mathrm{M}} P_1' \parallel_S P_2} \qquad \dfrac{P_2 \stackrel{\lambda}{\longrightarrow}_{\mathrm{M}} P_2'}{P_1 \parallel_S P_2 \stackrel{\lambda}{\longrightarrow}_{\mathrm{M}} P_1 \parallel_S P_2'} \end{array}}$$

Before defining the interactive variant of Markovian bisimilarity, we adapt to this setting the concept of exit rate.

Definition 19. *Let $P \in \mathcal{P}_{\mathrm{CI}}$ and $C \subseteq \mathcal{P}_{\mathrm{CI}}$. The exit rate of P towards C is defined through the following non-negative real function:*

$$\boxed{\mathit{rate}_{\mathrm{M}}(P, C) = \textstyle\sum \{\!|\, \lambda \in \mathbb{R}_{>0} \mid \exists P' \in C.\, P \stackrel{\lambda}{\longrightarrow}_{\mathrm{M}} P' \,|\!\}}$$

where the summation is taken to be zero whenever its multiset is empty. ■

Definition 20. *An equivalence relation $\mathcal{B} \subseteq \mathcal{P}_{\rm CI} \times \mathcal{P}_{\rm CI}$ is an interactive Markovian bisimulation iff, whenever $(P_1, P_2) \in \mathcal{B}$, then for all action names $a \in Name \cup \{\tau\}$ and equivalence classes $C \in \mathcal{P}_{\rm CI}/\mathcal{B}$:*

- $P_1 \stackrel{a}{\longrightarrow}_{\rm I} P_1'$ *implies* $P_2 \stackrel{a}{\longrightarrow}_{\rm I} P_2'$ *for some* P_2' *with* $(P_1', P_2') \in \mathcal{B}$.
- $P_1 \stackrel{\tau}{\not\longrightarrow}_{\rm I}$ *implies* $rate_{\rm M}(P_1, C) = rate_{\rm M}(P_2, C)$. ■

Definition 21. *Interactive Markovian bisimilarity, denoted by $\sim_{\rm IMB}$, is the union of all the interactive Markovian bisimulations.* ■

$\sim_{\rm IMB}$ enjoys properties similar to those of $\sim_{\rm MB}$. The characterizing axioms are:

$$\boxed{\begin{array}{c} a.P + a.P = a.P \\ (\lambda_1).P + (\lambda_2).P = (\lambda_1 + \lambda_2).P \\ \tau.P + (\lambda).Q = \tau.P \end{array}}$$

which encode idempotency, race policy, and maximal progress, respectively.

Since τ-actions are invisible and take no time, when comparing process terms they should be abstracted away. This can be achieved by weakening $\sim_{\rm IMB}$ in such a way that, after any non-pre-emptable exponential delay, all the states that can evolve via a finite sequence of τ-transitions to a given class are skipped.

Definition 22. *Let $C \subseteq \mathcal{P}_{\rm CI}$. The internal backward closure of C is defined as follows:*

$$C_\tau = \{P' \in \mathcal{P}_{\rm CI} \mid \exists P \in C.\, P' \stackrel{\tau^*}{\Longrightarrow}_{\rm I} P\}$$

where $P' \stackrel{\tau^*}{\Longrightarrow}_{\rm I} P$ means that P' can evolve to P after a finite sequence of zero or more τ-transitions. ■

Definition 23. *An equivalence relation $\mathcal{B} \subseteq \mathcal{P}_{\rm CI} \times \mathcal{P}_{\rm CI}$ is a weak interactive Markovian bisimulation iff, whenever $(P_1, P_2) \in \mathcal{B}$, then for all action names $a \in Name$ and equivalence classes $C \in \mathcal{P}_{\rm CI}/\mathcal{B}$:*

- $P_1 \stackrel{a}{\longrightarrow}_{\rm I} P_1'$ *implies* $P_2 \stackrel{\tau^* a \tau^*}{\Longrightarrow}_{\rm I} P_2'$ *for some* P_2' *with* $(P_1', P_2') \in \mathcal{B}$.
- $P_1 \stackrel{\tau}{\longrightarrow}_{\rm I} P_1'$ *implies* $P_2 \stackrel{\tau^*}{\Longrightarrow}_{\rm I} P_2'$ *for some* P_2' *with* $(P_1', P_2') \in \mathcal{B}$.
- $P_1 \stackrel{\tau^*}{\Longrightarrow}_{\rm I} P_1' \stackrel{\tau}{\not\longrightarrow}_{\rm I}$ *implies* $P_2 \stackrel{\tau^*}{\Longrightarrow}_{\rm I} P_2' \stackrel{\tau}{\not\longrightarrow}_{\rm I}$ *for some* P_2' *with* $rate_{\rm M}(P_1', C_\tau) = rate_{\rm M}(P_2', C_\tau)$. ■

Definition 24. *Weak interactive Markovian bisimilarity, denoted by $\approx_{\rm IMB}$, is the union of all the weak interactive Markovian bisimulations.* ■

$\approx_{\rm IMB}$ is strictly coarser than $\sim_{\rm IMB}$ but it is not a congruence with respect to alternative composition. As usual, initial τ-actions need a different treatment.

Definition 25. *Let $P_1, P_2 \in \mathcal{P}_{\rm CI}$. We say that P_1 is weakly interactive Markovian bisimulation congruent to P_2, written $P_1 \simeq_{\rm IMB} P_2$, iff for all action names $a \in Name \cup \{\tau\}$ and equivalence classes $C \in \mathcal{P}_{\rm CI}/\approx_{\rm IMB}$:*

– $P_1 \xrightarrow{a}_{\rm I} P_1'$ *implies* $P_2 \xRightarrow{\tau^* a \tau^*}_{\rm I} P_2'$ *for some* P_2' *with* $P_1' \approx_{\rm IMB} P_2'$.
– $P_2 \xrightarrow{a}_{\rm I} P_2'$ *implies* $P_1 \xRightarrow{\tau^* a \tau^*}_{\rm I} P_1'$ *for some* P_1' *with* $P_1' \approx_{\rm IMB} P_2'$.
– $P_1 \not\xrightarrow{\tau}_{\rm I}$ *iff* $P_2 \not\xrightarrow{\tau}_{\rm I}$.
– $P_1 \not\xrightarrow{\tau}_{\rm I}$ *implies* $rate_{\rm M}(P_1, C) = rate_{\rm M}(P_2, C)$. ■

It turns out $\sim_{\rm IMB} \subset \simeq_{\rm IMB} \subset \approx_{\rm IMB}$ with $\simeq_{\rm IMB}$ enjoying the same properties as $\sim_{\rm IMB}$. Moreover, $\simeq_{\rm IMB}$ has the following additional characterizing axioms:

$$\begin{array}{|c|}\hline a.\tau.P = a.P \\ P + \tau.P = \tau.P \\ a.(P + \tau.Q) + \tau.Q = a.(P + \tau.Q) \\ (\lambda).\tau.P = (\lambda).P \\ \hline \end{array}$$

which are called τ-laws and witness the capability of $\simeq_{\rm IMB}$ of abstracting from τ-actions.

4.3 Extended Markovian Bisimilarity

Markovian bisimilarity is restricted to exponential distributions. Although their combinations can approximate most of the general distributions arbitrarily closely, some useful distributions are left out, specially the one representing a zero duration. The capability of expressing zero durations would also constitute a good performance abstraction mechanism, similarly to the functional abstraction mechanism given by the invisible action name τ.

In the modeling process it often happens to deal with choices among logical events (like the reception of a message vs. its loss) with which no timing can reasonably be associated, or to encounter activities that are several orders of magnitude faster than the activities that are important for the evaluation of certain performance measures. In all of these cases, using a zero duration would be an adequate solution from the modeling standpoint.

Zero durations can be introduced by admitting the so-called immediate actions. Each of them has a name and a zero duration (or, equivalently, an infinite rate), together with a priority level $l \in \mathbb{N}_{>0}$ and a weight $w \in \mathbb{R}_{>0}$. Priority levels and weights are used to choose among several immediate actions that are simultaneously enabled. According to the generative [26] preselection policy, each of the highest priority immediate actions that are enabled is given an execution probability proportional to its weight.

It is worth noting that this extended Markovian framework is complementary with respect to the interactive Markovian framework, because it is as if nondeterminism were ruled out by augmenting each interacting action with a priority level and a weight. Here maximal progress is subsumed by pre-emption: immediate τ-actions take precedence over all the lower priority actions.

We now derive from CMPC a new calculus, which we call concurrent extended Markovian process calculus (CEMPC):

– The syntax is extended as follows:

$$\begin{array}{|l|}\hline P ::= \underline{0} \\ \quad \mid \; <a, \lambda>.P \\ \quad \mid \; <a, \infty_{l,w}>.P \\ \quad \mid \; <a, *_w^{l'}>.P \\ \quad \mid \; P + P \\ \quad \mid \; P \parallel_S P \\ \quad \mid \; X \\ \quad \mid \; rec\, X : P \\ \hline \end{array}$$

where $a \in Name \cup \{\tau\}$, $l \in \mathbf{N}_{>0}$, and $l' \in \mathbf{N}$. We denote by $\mathcal{P}_{\mathrm{CE}}$ the set of the closed and guarded process terms of CEMPC. We point out that every passive action has been augmented with a priority constraint, which is useful to keep under control the process priority interrelation. Besides synchronizing with passive actions with the same priority constraint, a passive action with priority constraint 0 can only synchronize with an exponentially timed action, while a passive action with priority constraint $l' > 0$ can only synchronize with an immediate action with priority level $l = l'$.

– The additional semantic rules specific for immediate actions are the following:

$$\boxed{<a, \infty_{l,w}>.P \xrightarrow{a,\infty_{l,w}} P}$$

$$\boxed{\dfrac{P_1 \xrightarrow{a,\infty_{l,w}} P_1' \qquad P_2 \xrightarrow{a,*_v^l} P_2' \qquad a \in S}{P_1 \parallel_S P_2 \xrightarrow{a,\infty_{l,w} \cdot \frac{v}{weight(P_2,a,l)}} P_1' \parallel_S P_2'}}$$

$$\boxed{\dfrac{P_1 \xrightarrow{a,*_v^l} P_1' \qquad P_2 \xrightarrow{a,\infty_{l,w}} P_2' \qquad a \in S}{P_1 \parallel_S P_2 \xrightarrow{a,\infty_{l,w} \cdot \frac{v}{weight(P_1,a,l)}} P_1' \parallel_S P_2'}}$$

Note that, consistently with the asymmetric action synchronization discipline, immediate actions can synchronize only with passive actions.

Before defining the extended variant of Markovian bisimilarity, we extend to immediate actions the notion of exit rate. Below, the priority level of an action is encoded through a number in $\mathbf{Z}$, which is 0 if the action is exponentially timed, l if the action rate is $\infty_{l,w}$, $-l-1$ if the action rate is $*_w^l$.

Definition 26. *Let $P \in \mathcal{P}_{\mathrm{CE}}$, $a \in Name \cup \{\tau\}$, $l \in \mathbf{Z}$, and $C \subseteq \mathcal{P}_{\mathrm{CE}}$. The exit rate of P when executing actions of name a and priority level l that lead to C is defined through the following non-negative real function:*

$$\boxed{rate(P,a,l,C) = \begin{cases} \sum\{\!|\, \lambda \in \mathbf{R}_{>0} \mid \exists P' \in C.\, P \xrightarrow{a,\lambda} P' \,|\!\} & \text{if } l = 0 \\ \sum\{\!|\, w \in \mathbf{R}_{>0} \mid \exists P' \in C.\, P \xrightarrow{a,\infty_{l,w}} P' \,|\!\} & \text{if } l > 0 \\ \sum\{\!|\, w \in \mathbf{R}_{>0} \mid \exists P' \in C.\, P \xrightarrow{a,*_w^{-l-1}} P' \,|\!\} & \text{if } l < 0 \end{cases}}$$

where each summation is taken to be zero whenever its multiset is empty. ■

In the following we denote by $pri^{\tau}_{\infty}(P)$ the priority level of the highest priority immediate τ-action enabled by P, and we set $pri^{\tau}_{\infty}(P) = 0$ if P does not enable any immediate τ-action. Moreover, given $l \in \mathbb{Z}$, we use *no-pre*(l, P) to denote that no action whose priority level is l can be pre-empted in P. Formally, this is the case whenever $l \geq pri^{\tau}_{\infty}(P)$ or $-l - 1 \geq pri^{\tau}_{\infty}(P)$.

Definition 27. *An equivalence relation $\mathcal{B} \subseteq \mathcal{P}_{\mathrm{CE}} \times \mathcal{P}_{\mathrm{CE}}$ is an extended Markovian bisimulation iff, whenever $(P_1, P_2) \in \mathcal{B}$, then for all action names $a \in Name \cup \{\tau\}$, equivalence classes $C \in \mathcal{P}_{\mathrm{CE}}/\mathcal{B}$, and priority levels $l \in \mathbb{Z}$ such that no-pre(l, P_1) and no-pre(l, P_2):*

$$rate(P_1, a, l, C) = rate(P_2, a, l, C)$$

■

Definition 28. *Extended Markovian bisimilarity, denoted by $\sim_{\mathrm{EMB}}$, is the union of all the extended Markovian bisimulations.* ■

$\sim_{\mathrm{EMB}}$ enjoys the same properties as $\sim_{\mathrm{MB}}$. The characterizing axioms are:

$$\begin{array}{|rll|}
\hline
<a, \lambda_1>.P + <a, \lambda_2>.P &= <a, \lambda_1 + \lambda_2>.P & \\
<a, \infty_{l,w_1}>.P + <a, \infty_{l,w_2}>.P &= <a, \infty_{l,w_1+w_2}>.P & \\
<a, *^{l}_{w_1}>.P + <a, *^{l}_{w_2}>.P &= <a, *^{l}_{w_1+w_2}>.P & \\
<\tau, \infty_{l,w}>.P + <a, \lambda>.Q &= <\tau, \infty_{l,w}>.P & \\
<\tau, \infty_{l,w}>.P + <a, \infty_{l',w'}>.Q &= <\tau, \infty_{l,w}>.P & \text{if } l > l' \\
<\tau, \infty_{l,w}>.P + <a, *^{l'}_{w'}>.Q &= <\tau, \infty_{l,w}>.P & \text{if } l > l' \\
\hline
\end{array}$$

with the last three encoding pre-emption.

Similarly to the interactive framework, when comparing process terms the immediate τ-actions should be abstracted away, as they are unobservable both from the functional viewpoint and from the performance viewpoint. However, weakening $\sim_{\mathrm{EMB}}$ is harder than weakening $\sim_{\mathrm{IMB}}$, because it is necessary to keep track of the priority levels and of the weights associated with the actions to be abstracted away. Furthemore, it is also necessary to take into account the degree of observability of the states.

Definition 29. *Let $P \in \mathcal{P}_{\mathrm{CE}}$ and $l \in \mathbb{N}_{>0}$. We say that P is l-unobservable iff $pri^{\tau}_{\infty}(P) = l$ and P does not enable any visible action with priority level $l' \in \mathbb{Z}$ such that $l' \geq l$ or $-l' - 1 \geq l$.* ■

Definition 30. *Let $n \in \mathbb{N}_{>0}$ and $P_1, P_2, \ldots, P_{n+1} \in \mathcal{P}_{\mathrm{CE}}$. A computation c of length n:*

$$P_1 \stackrel{\tau,\infty_{l_1,w_1}}{\longrightarrow} P_2 \stackrel{\tau,\infty_{l_2,w_2}}{\longrightarrow} \ldots \stackrel{\tau,\infty_{l_n,w_n}}{\longrightarrow} P_{n+1}$$

is unobservable iff for all $i = 1, \ldots, n$ process term P_i is l_i-unobservable. In that case, the probability of executing c is given by:

$$prob(c) = \prod_{i=1}^{n} \frac{w_i}{rate(P_i, \tau, l_i, \mathcal{P}_{\mathrm{CE}})}$$

■

Definition 31. *Let $P \in \mathcal{P}_{\mathrm{CE}}$, $a \in Name \cup \{\tau\}$, $l \in \mathbb{Z}$, and $C \subseteq \mathcal{P}_{\mathrm{CE}}$. The weak exit rate of P when executing actions with name a and priority level l that lead to C is defined through the following non-negative real function:*

$$\boxed{rate_{\mathrm{w}}(P, a, l, C) = \sum_{P' \in C_{\mathrm{w}}} rate(P, a, l, \{P'\}) \cdot prob_{\mathrm{w}}(P', C)}$$

where:

- *C_{w} is the weak backward closure of C:*
  $C_{\mathrm{w}} = C \cup \{Q \in \mathcal{P}_{\mathrm{CE}} - C \mid Q \text{ can reach } C \text{ via unobservable computations}\}$
- *$prob_{\mathrm{w}}$ is a $\mathbb{R}_{]0,1]}$-valued function representing the sum of the probabilities of all the unobservable computations from a term in C_{w} to C:*
  $$prob_{\mathrm{w}}(P', C) = \begin{cases} 1 & \text{if } P' \in C \\ \sum\{\!|\, prob(c) \mid c \text{ unobservable computation from } P' \text{ to } C \,|\!\} & \\ & \text{if } P' \in C_{\mathrm{w}} - C \end{cases}$$ ■

The definition of $\sim_{\mathrm{EMB}}$ can be weakened by using $rate_{\mathrm{w}}$ instead of $rate$ and by skipping the weak exit rate comparison for some equivalence classes that contain certain unobservable states:

- An observable state is a state that enables an observable action that cannot be pre-empted by any enabled unobservable action.
- An initially unobservable state is a state in which all the enabled observable actions are pre-empted by some enabled unobservable action, but at least one of the computations starting at this state with one of the higher priority enabled unobservable actions reaches an observable state.
- A fully unobservable state is a state in which all the enabled observable actions are pre-empted by some enabled unobservable action, and all the computations starting at this state with one of the higher priority enabled unobservable actions are unobservable (note that $\underline{0}$ is fully unobservable). We denote by $\mathcal{P}_{\mathrm{CE,fu}}$ the set of the fully unobservable process terms of $\mathcal{P}_{\mathrm{CE}}$.

Definition 32. *An equivalence relation $\mathcal{B} \subseteq \mathcal{P}_{\mathrm{CE}} \times \mathcal{P}_{\mathrm{CE}}$ is a weak extended Markovian bisimulation iff, whenever $(P_1, P_2) \in \mathcal{B}$, then for all action names $a \in Name \cup \{\tau\}$ and priority levels $l \in \mathbb{Z}$ such that no-pre(l, P_1) and no-pre(l, P_2):*

$$rate_{\mathrm{w}}(P_1, a, l, C) = rate_{\mathrm{w}}(P_2, a, l, C) \qquad \text{for all observable } C \in \mathcal{P}_{\mathrm{CE}}/\mathcal{B}$$
$$rate_{\mathrm{w}}(P_1, a, l, \mathcal{P}_{\mathrm{CE,fu}}) = rate_{\mathrm{w}}(P_2, a, l, \mathcal{P}_{\mathrm{CE,fu}})$$ ■

Definition 33. *Weak extended Markovian bisimilarity, denoted by $\approx_{\mathrm{EMB}}$, is the union of all the weak extended Markovian bisimulations.* ■

$\approx_{\mathrm{EMB}}$ enjoys the same properties as $\sim_{\mathrm{EMB}}$ except for congruence. In fact, to recover congruence with respect to parallel composition, we have to restrict ourselves to the set $\mathcal{P}_{\mathrm{CE,wp}}$ of the well-prioritized process terms of $\mathcal{P}_{\mathrm{CE}}$. This is the smallest subset of $\mathcal{P}_{\mathrm{CE}}$ closed with respect to null term, action prefix, alternative composition, recursion and closed with respect to parallel composition under the

following constraint: If $P_1, P_2 \in \mathcal{P}_{\mathrm{CE,wp}}$ and any immediate/passive transition of each of $[\![P_1]\!]$ and $[\![P_2]\!]$ has priority level/constraint less than the priority level of any unobservable transition departing from an unobservable state of the other one, then $P_1 \parallel_S P_2 \in \mathcal{P}_{\mathrm{CE,wp}}$.

The additional characterizing axioms of $\approx_{\mathrm{EMB}}$ over $\mathcal{P}_{\mathrm{CE,wp}}$ are the following:

$$\begin{array}{|c|}\hline <a,\lambda>.\sum_{i\in I} <\tau,\infty_{l,w_i}>.P_i = \sum_{i\in I} <a,\lambda\cdot w_i/\,\Sigma_{k\in I}\, w_k>.P_i \\ <a,\infty_{l',w'}>.\sum_{i\in I} <\tau,\infty_{l,w_i}>.P_i = \sum_{i\in I} <a,\infty_{l',w'\cdot w_i/\,\Sigma_{k\in I}\, w_k}>.P_i \\ <a,*^{l'}_{w'}>.\sum_{i\in I} <\tau,\infty_{l,w_i}>.P_i = \sum_{i\in I} <a,*^{l'}_{w'\cdot w_i/\,\Sigma_{k\in I}\, w_k}>.P_i \\ \hline\end{array}$$

which witness the capability of $\approx_{\mathrm{EMB}}$ of abstracting from immediate τ-actions in a way that correctly takes into account their weights.

5 Markovian Testing Equivalence

Markovian testing equivalence considers two process terms to be equivalent whenever an external observer, who can interact with them by means of tests, is not able to distinguish between them from the functional or performance viewpoint. In this section we provide the definition of Markovian testing equivalence over $\mathcal{P}_{\mathrm{C,pc}}$ and we recall its properties from [7, 9].

5.1 Test Formalization

The only way that the external observer has to infer information about the behavior of the process terms is to interact with them by means of tests and look at their reactions. Was the test passed? If so, with which probability? And how long did it take to pass the test?

In our Markovian framework with asymmetric action synchronization discipline, the most convenient way to represent a test is through another process term composed of passive actions only, which interacts with the terms to be tested by means of a parallel composition operator that enforces synchronization on any action name. In this way, the parallel composition of a performance closed term to be tested and a test will still be performance closed.

From the testing viewpoint, in any of its states a process term to be tested generates the proposal of an action to be executed by means of a race among the exponentially timed actions enabled in that state. Then the test either reacts by participating in the interaction with the process term through a passive action having the same name as the proposed exponentially timed action, or blocks the interaction if it has no passive actions with the proposed name.

Since it is necessary to measure the probability with which process terms pass tests within a finite amount of time, for the test formalization we can restrict ourselves to non-recursive terms (composed of passive actions only). In other words, the expressiveness provided by labeled multitransition systems with a finite dag-like structure will be enough for the tests.

In order to represent the fact that a test is passed or not, each of the terminal nodes of the dag-like semantic model underlying a test must be suitably labeled so as to establish whether it is a success or failure state. At the process calculus level, this amounts to replace $\underline{0}$ with two zeroary operators, which we denote by "s" (for success) and "f" (for failure).

Ambiguous terms like $\mathrm{s} + \mathrm{f}$ will be avoided in the test syntax by replacing the action prefix operator and the binary alternative composition operator with a set of n-ary guarded alternative composition operators, with n ranging over the whole $\mathbf{N}_{>0}$.

Definition 34. *The set $\mathcal{T}$ of the tests is generated by the following syntax:*

$$\begin{array}{l} T ::= \mathrm{f} \\ \quad \mid \mathrm{s} \\ \quad \mid \sum\limits_{i \in I} <a_i, *_{w_i}>.T_i \end{array}$$

where I is a non-empty finite index set. ■

5.2 Equivalence Definition

Markovian testing equivalence relies on comparing the process term probabilities of performing a successful test-driven computation within a given sequence of average amounts of time. A test-driven computation is a sequence of transitions in the labeled multitransition system underlying the parallel composition of a process term and a test. Due to the restrictions imposed on the test syntax, all the considered test-driven computations will turn out to have a finite length, hence the inductive definitions of Sect. 2.3 apply to them.

Definition 35. *Let $P \in \mathcal{P}_{\mathrm{C,pc}}$ and $T \in \mathcal{T}$. The interaction system of P and T is process term $P \parallel_{Name} T \in \mathcal{P}_{\mathrm{C,pc}}$, where we say that:*

- *A configuration is a state of $[\![P \parallel_{Name} T]\!]$.*
- *A configuration is formed by a process part and a test part.*
- *A configuration is successful (resp. failed) iff its test part is "s" (resp. "f").*
- *A computation is successful (resp. failed) iff so is its last configuration.*
- *A computation that is neither successful nor failed is interrupted.*

We denote by $\mathcal{SC}(P,T)$ the multiset of the successful computations of $\mathcal{C}_{\mathrm{f}}(P \parallel_{Name} T)$. ■

Note that $\mathcal{SC}(P,T) \subseteq \mathcal{I}_{\mathrm{f}}(P \parallel_{Name} T)$, because of the maximality of the successful test-driven computations, and that $\mathcal{SC}(P,T)$ is finite, because of the finitely-branching structure of the considered terms.

Definition 36. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. We say that P_1 is Markovian testing equivalent to P_2, written $P_1 \sim_{\mathrm{MT}} P_2$, iff for all tests $T \in \mathcal{T}$ and sequences $\theta \in (\mathbb{R}_{>0})^*$ of average amounts of time:*

$$prob(\mathcal{SC}_{\leq\theta}(P_1,T)) = prob(\mathcal{SC}_{\leq\theta}(P_2,T))$$

■

Obviously, $\sim_{\rm MT}$ is strictly finer than classical testing equivalence [21] and probabilistic testing equivalence [17, 19]. On the other hand, it is strictly coarser than $\sim_{\rm MB}$ as it is less sensitive to branching points. A consequence of this fact is that the derivatives of two Markovian testing equivalent terms are not necessarily related by $\sim_{\rm MT}$. We conclude with a necessary condition.

Proposition 2. *Let $P_1, P_2 \in \mathcal{P}_{\rm C,pc}$ and $T \in \mathcal{T}$. Whenever $P_1 \sim_{\rm MT} P_2$, then for all $c_k \in \mathcal{SC}(P_k, T)$ with $k \in \{1,2\}$ there exists $c_h \in \mathcal{SC}(P_h, T)$ with $h \in \{1,2\} - \{k\}$ such that:*

$$trace(c_k) = trace(c_h)$$
$$time_{\rm a}(c_k) = time_{\rm a}(c_h)$$

and for all $a \in Name$:

$$rate(P_{k,{\rm last}}, a, 0, \mathcal{P}_{\rm C}) = rate(P_{h,{\rm last}}, a, 0, \mathcal{P}_{\rm C})$$

with $P_{k,{\rm last}}$ (resp. $P_{h,{\rm last}}$) being the process part of the last configuration of c_k (resp. c_h). ■

5.3 Alternative Characterizations

We now present two alternative characterizations of $\sim_{\rm MT}$. The first one is based on the probability distribution of passing a test within a certain sequence of amounts of time. A consequence of this characterization is that considering the (more accurate) stepwise durations of the test-driven computations leads to the same equivalence as considering the (easier to work with) stepwise average durations of the test-driven computations. This justifies the use of expected values instead of random variables in the definition of $\sim_{\rm MT}$.

Definition 37. *Let $P_1, P_2 \in \mathcal{P}_{\rm C,pc}$. We say that P_1 is Markovian distribution-testing equivalent to P_2, written $P_1 \sim_{\rm MT,d} P_2$, iff for all tests $T \in \mathcal{T}$ and sequences $\theta \in (\mathbf{R}_{>0})^*$ of amounts of time:*

$$prob_{\rm d}(\mathcal{SC}(P_1, T), \theta) = prob_{\rm d}(\mathcal{SC}(P_2, T), \theta)$$ ■

Theorem 6. *Let $P_1, P_2 \in \mathcal{P}_{\rm C,pc}$. Then:*

$$P_1 \sim_{\rm MT,d} P_2 \iff P_1 \sim_{\rm MT} P_2$$ ■

The second alternative characterization of $\sim_{\rm MT}$ is based on traces that are suitably extended with the sets of the action names permitted at each step by the environment. This means that it is possible to characterize $\sim_{\rm MT}$ in a way that fully abstracts from the tests. A consequence of the proof of this characterization is the identification of a set of canonical tests, i.e. a set of tests that are necessary and sufficient in order to establish whether two process terms are Markovian testing equivalent. Each such test admits a single computation leading to success, whose states can have additional computations each leading to failure in one step.

Definition 38. *An element σ of $(Name \times 2^{Name})^*$ is an extended trace iff either σ is the empty sequence or:*

$$\sigma \equiv (a_1, \mathcal{E}_1) \circ (a_2, \mathcal{E}_2) \circ \ldots \circ (a_n, \mathcal{E}_n)$$

for some $n \in \mathbf{N}_{>0}$ with $a_i \in \mathcal{E}_i$ for each $i = 1, \ldots, n$. We denote by $\mathcal{ET}$ the set of the extended traces. ■

Definition 39. *Let $\sigma \in \mathcal{ET}$. The trace associated with σ is defined by induction on the length of σ through the following $Name^*$-valued function:*

$$\boxed{trace(\sigma) = \begin{cases} \varepsilon & \text{if } length(\sigma) = 0 \\ a \circ trace(\sigma') & \text{if } \sigma \equiv (a, \mathcal{E}) \circ \sigma' \end{cases}}$$

where ε is the empty trace. ■

Definition 40. *Let $P \in \mathcal{P}_{\mathrm{C,pc}}$, $c \in \mathcal{C}_{\mathrm{f}}(P)$, and $\sigma \in \mathcal{ET}$. We say that c is compatible with σ iff:*

$$trace(c) = trace(\sigma)$$

We denote by $\mathcal{CC}(P, \sigma)$ the multiset of the computations of $\mathcal{C}_{\mathrm{f}}(P)$ that are compatible with σ. ■

Note that $\mathcal{CC}(P, \sigma) \subseteq \mathcal{I}_{\mathrm{f}}(P)$ because of the compatibility of the considered computations with the same extended trace σ.

Definition 41. *Let $P \in \mathcal{P}_{\mathrm{C,pc}}$, $\sigma \in \mathcal{ET}$, and $c \in \mathcal{CC}(P, \sigma)$. The probability of executing c with respect to σ is defined by induction on the length of c through the following $\mathbb{R}_{]0,1]}$-valued function:*

$$\boxed{prob^{\sigma}(c) = \begin{cases} 1 & \text{if } length(c) = 0 \\ \dfrac{\lambda}{\sum\limits_{b \in \mathcal{E}} rate(P, b, 0, \mathcal{P}_{\mathrm{C}})} \cdot prob^{\sigma'}(c') & \text{if } c \equiv P \xrightarrow{a,\lambda} c' \\ & \text{with } \sigma \equiv (a, \mathcal{E}) \circ \sigma' \end{cases}}$$

We also define the probability of executing a computation of C with respect to σ as:

$$\boxed{prob^{\sigma}(C) = \sum_{c \in C} prob^{\sigma}(c)}$$

for all $C \subseteq \mathcal{CC}(P, \sigma)$. ■

Definition 42. *Let $P \in \mathcal{P}_{\mathrm{C,pc}}$, $\sigma \in \mathcal{ET}$, and $c \in \mathcal{CC}(P, \sigma)$. The stepwise average duration of c with respect to σ is defined by induction on the length of c through the following $(\mathbb{R}_{>0})^*$-valued function:*

$$\boxed{time_{\mathrm{a}}^{\sigma}(c) = \begin{cases} \varepsilon & \text{if } length(c) = 0 \\ \dfrac{1}{\sum\limits_{b \in \mathcal{E}} rate(P, b, 0, \mathcal{P}_{\mathrm{C}})} \circ time_{\mathrm{a}}^{\sigma'}(c') & \text{if } c \equiv P \xrightarrow{a,\lambda} c' \\ & \text{with } \sigma \equiv (a, \mathcal{E}) \circ \sigma' \end{cases}}$$

where ε is the empty stepwise average duration. We also define the multiset of the computations of C whose stepwise average duration with respect to σ is not greater than θ as:

$$\boxed{C_{\leq \theta}^{\sigma} = \{\!|\, c \in C \mid length(c) \leq length(\theta) \wedge \forall i = 1, \ldots, length(c).\, time_{\mathrm{a}}^{\sigma}(c)[i] \leq \theta[i] \,|\!\}}$$

for all $C \subseteq \mathcal{CC}(P, \sigma)$ and $\theta \in (\mathbb{R}_{>0})^*$. ■

Definition 43. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. We say that P_1 is Markovian extended-trace equivalent to P_2, written $P_1 \sim_{\mathrm{MTr,e}} P_2$, iff for all extended traces $\sigma \in \mathcal{ET}$ and sequences $\theta \in (\mathbf{R}_{>0})^*$ of average amounts of time:*

$$prob^{\sigma}(\mathcal{CC}^{\sigma}_{\leq\theta}(P_1,\sigma)) = prob^{\sigma}(\mathcal{CC}^{\sigma}_{\leq\theta}(P_2,\sigma))$$

■

Theorem 7. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. Then:*

$$P_1 \sim_{\mathrm{MTr,e}} P_2 \iff P_1 \sim_{\mathrm{MT}} P_2$$

■

Definition 44. *The set $\mathcal{T}_{\mathrm{c}}$ of the canonical tests is generated by the following syntax:*

$$\boxed{\begin{array}{l} T ::= \mathrm{s} \\ \quad \mid \; <a, *_1>.T + \displaystyle\sum_{b \in \mathcal{E}-\{a\}} <b, *_1>.\mathrm{f} \end{array}}$$

where the summation is absent whenever $\mathcal{E} - \{a\} = \emptyset$. ■

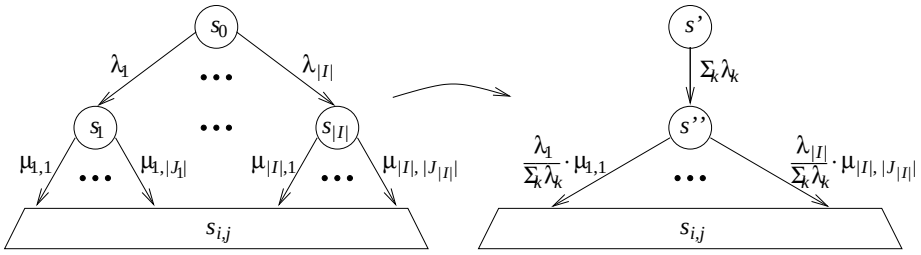
Corollary 2. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. Then $P_1 \sim_{\mathrm{MT}} P_2$ iff for all $T \in \mathcal{T}_{\mathrm{c}}$ and $\theta \in (\mathbf{R}_{>0})^*$:*

$$prob(\mathcal{SC}_{\leq\theta}(P_1,T)) = prob(\mathcal{SC}_{\leq\theta}(P_2,T))$$

■

5.4 Exactness

Markovian testing equivalence induces a CTMC-level aggregation that is strictly coarser than ordinary lumping and was not known in the CTMC field. This aggregation can be depicted through the following rewriting rule:



where:

- I is a finite index set with $|I| \geq 2$.
- k ranges over I.
- J_i is a non-empty finite index set for all $i \in I$.
- For all $i_1, i_2 \in I$:

$$\boxed{\sum_{j \in J_{i_1}} \mu_{i_1,j} = \sum_{j \in J_{i_2}} \mu_{i_2,j}}$$

  with each summation being zero whenever its index set is empty.

Theorem 8. *The CTMC-level aggregation induced by $\sim_{\mathrm{MT}}$ is exact.* ■

5.5 Congruence

Markovian testing equivalence is a congruence with respect to all the operators of CMPC. In particular, we have what follows.

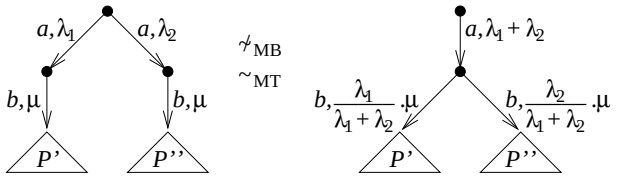
Theorem 9. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. Whenever $P_1 \sim_{\mathrm{MT}} P_2$, then:*

1. $<a,\lambda>.P_1 \sim_{\mathrm{MT}} <a,\lambda>.P_2$ *for all* $<a,\lambda> \in Act_{\mathrm{S}}$.
2. $P_1 + P \sim_{\mathrm{MT}} P_2 + P$ *and* $P + P_1 \sim_{\mathrm{MT}} P + P_2$ *for all* $P \in \mathcal{P}_{\mathrm{C,pc}}$.
3. $P_1 \parallel_S P \sim_{\mathrm{MT}} P_2 \parallel_S P$ *and* $P \parallel_S P_1 \sim_{\mathrm{MT}} P \parallel_S P_2$ *for all* $S \subseteq Name$ *and* $P \in \mathcal{P}_{\mathrm{C}}$ *containing only passive actions such that* $P_1 \parallel_S P, P_2 \parallel_S P \in \mathcal{P}_{\mathrm{C,pc}}$.

■

5.6 Axiomatization

Markovian testing equivalence is strictly coarser than Markovian bisimilarity, hence the axioms of Table 1 are still valid for $\sim_{\mathrm{MT}}$ over $\mathcal{P}_{\mathrm{C,pc}}$, but not complete. In fact, the two process terms depicted below (with $P' \not\sim_{\mathrm{MB}} P''$):



show that $\sim_{\mathrm{MB}}$ is highly sensitive to branching points. By contrast, $\sim_{\mathrm{MT}}$ allows choices to be deferred as long as they are related to branches starting with actions having the same name that are immediately followed by actions having the same names and the same total rates in all the branches.

The two terms above constitute the simplest instance of an axiom schema subsuming $\mathcal{A}_4^{\mathrm{MB}}$ that characterizes $\sim_{\mathrm{MT}}$. The axiom schema is the following:

$$\sum_{i\in I} <a,\lambda_i>.\sum_{j\in J_i} <b_{i,j},\mu_{i,j}>.P_{i,j} = <a,\sum_{k\in I}\lambda_k>.\sum_{i\in I}\sum_{j\in J_i} <b_{i,j},\frac{\lambda_i}{\Sigma_{k\in I}\lambda_k}\cdot\mu_{i,j}>.P_{i,j}$$

where:

- I is a finite index set with $|I| \geq 2$.
- J_i is a finite index set for all $i \in I$, with the related summation being $\underline{0}$ whenever $J_i = \emptyset$.
- For all $i_1, i_2 \in I$ and $b \in Name$:

$$\sum_{j\in J_{i_1}} \{\!|\, \mu_{i_1,j} \mid b_{i_1,j} = b \,|\!\} = \sum_{j\in J_{i_2}} \{\!|\, \mu_{i_2,j} \mid b_{i_2,j} = b \,|\!\}$$

  with each summation being zero whenever its index set is empty.

5.7 Logical Characterization

Markovian testing equivalence has a modal logic characterization over $\mathcal{P}_{\mathrm{C,pc}}$ based on a Markovian variant of a restriction of the Hennessy-Milner logic, in which both negation and logical conjunction are ruled out, while the diamond operator is made dependent from the environment.

Unlike $\mathrm{HML}_{\mathrm{MB}}$, where the syntax is decorated with rate lower bounds and the satisfaction relation is qualitative, here there is no extra information in the syntax and a quantitative interpretation inspired by [39] is adopted. This establishes the probability with which a process term satisfies a formula quickly enough on average, i.e. within a given sequence of average amounts of time.

Definition 45. *The set of the formulas of* $\mathrm{HML}_{\mathrm{MT}}$ *is generated by the following syntax:*

$$\boxed{\phi ::= \mathrm{true} \mid \langle a|\mathcal{E}\rangle\phi}$$

where $a \in Name$ *and* $\mathcal{E} \subseteq Name$ *such that* $a \in \mathcal{E}$. ■

Definition 46. *The interpretation function* $[\![.]\!]_{\mathrm{MT}}$ *of* $\mathrm{HML}_{\mathrm{MT}}$ *over* $\mathcal{P}_{\mathrm{C,pc}} \times (\mathbf{R}_{>0})^*$ *is defined by structural induction as follows:*

$$\boxed{\begin{array}{l} [\![\mathrm{true}]\!]_{\mathrm{MT}}(P,\theta) = 1 \\ [\![\langle a|\mathcal{E}\rangle\phi]\!]_{\mathrm{MT}}(P,\theta) = \begin{cases} 0 & \text{if } \theta = \varepsilon \vee \frac{1}{\sum_{b\in\mathcal{E}} rate(P,b,0,\mathcal{P}_{\mathrm{C}})} > \theta[1] \\ \sum\limits_{P \xrightarrow{a,\lambda} P'} \frac{\lambda}{\sum_{b\in\mathcal{E}} rate(P,b,0,\mathcal{P}_{\mathrm{C}})} \cdot [\![\phi]\!]_{\mathrm{MT}}(P',\theta') & \\ & \text{if } \theta = t \circ \theta' \wedge \frac{1}{\sum_{b\in\mathcal{E}} rate(P,b,0,\mathcal{P}_{\mathrm{C}})} \leq t \end{cases} \end{array}}$$

where the summation is taken to be zero whenever there are no a*-transitions departing from* P. ■

Theorem 10. *Let* $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. *Then:*

$P_1 \sim_{\mathrm{MT}} P_2 \iff \forall \phi \in \mathrm{HML}_{\mathrm{MT}}. \forall \theta \in (\mathbf{R}_{>0})^*. [\![\phi]\!]_{\mathrm{MT}}(P_1,\theta) = [\![\phi]\!]_{\mathrm{MT}}(P_2,\theta)$ ■

5.8 Verification Complexity

Markovian testing equivalence can be decided in polynomial time because two action-labeled CTMCs are Markovian testing equivalent iff their corresponding embedded action-labeled DTMCs (with suitably enriched labels) are probabilistic testing equivalent, with the latter coinciding with probabilistic ready equivalence and hence being decidable in polynomial time [35].

The algorithm to check whether $P_1 \sim_{\mathrm{MT}} P_2$ thus proceeds as follows:

1. Transform $[\![P_1]\!]$ and $[\![P_2]\!]$ into their corresponding embedded discrete-time versions:
   (a) Divide the rate of each transition by the total exit rate of its source state.

(b) Augment the name of each transition with the total exit rate of its source state.

2. Compute the equivalence $\mathcal{R}$ that relates any two states of the disjoint union of $[\![P_1]\!]$ and $[\![P_2]\!]$ such that their two sets of (original) action names labeling their outgoing transitions coincide.
3. For each equivalence class R induced by $\mathcal{R}$, apply the algorithm of [47] to check the embedded discrete-time versions of $[\![P_1]\!]$ and $[\![P_2]\!]$ for probabilistic language equivalence by considering R as the set of accepting states.

The time complexity is $O(n^5)$, where n is the number of states of the disjoint union of $[\![P_1]\!]$ and $[\![P_2]\!]$.

6 Markovian Trace Equivalence

Markovian trace equivalence considers two process terms to be equivalent whenever they are able to perform computations with the same functional and performance characteristics. In this section we provide the definition of Markovian trace equivalence over $\mathcal{P}_{\mathrm{C,pc}}$ and we recall its properties from [49, 7, 9].

6.1 Equivalence Definition

Markovian trace equivalence relies on comparing the process term probabilities of performing a computation within a given sequence of average amounts of time. We emphasize that here, given a process term $P \in \mathcal{P}_{\mathrm{C,pc}}$, we no longer have tests that interact with P. Instead, we directly consider the finite-length computations of P, to which the inductive definitions of Sect. 2.3 apply.

Definition 47. *Let $P \in \mathcal{P}_{\mathrm{C,pc}}$, $c \in \mathcal{C}_{\mathrm{f}}(P)$, and $\alpha \in Name^*$. We say that c is compatible with α iff:*

$$trace(c) = \alpha$$

We denote by $\mathcal{CC}(P, \alpha)$ the multiset of the finite-length computations of P that are compatible with α. ■

Note that $\mathcal{CC}(P, \alpha) \subseteq \mathcal{I}_{\mathrm{f}}(P)$, because of the compatibility of the computations with the same trace α, and that $\mathcal{CC}(P, \alpha)$ is finite, because of the finitely-branching structure of the considered terms.

Definition 48. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. We say that P_1 is Markovian trace equivalent to P_2, written $P_1 \sim_{\mathrm{MTr}} P_2$, iff for all traces $\alpha \in Name^*$ and sequences $\theta \in (\mathbb{R}_{>0})^*$ of average amounts of time:*

$$prob(\mathcal{CC}_{\leq \theta}(P_1, \alpha)) = prob(\mathcal{CC}_{\leq \theta}(P_2, \alpha))$$ ■

Obviously, $\sim_{\mathrm{MTr}}$ is strictly finer than classical trace equivalence [33] and probabilistic trace equivalence [36]. On the other hand, it is strictly coarser than $\sim_{\mathrm{MT}}$ as it completely overrides branching points. Thus, similarly to $\sim_{\mathrm{MT}}$, the derivatives of two Markovian trace equivalent terms are not necessarily related by $\sim_{\mathrm{MTr}}$. We conclude with a necessary condition.

Proposition 3. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$ and $\alpha \in Name^*$. Whenever $P_1 \sim_{\mathrm{MTr}} P_2$, then for all $c_k \in \mathcal{CC}(P_k, \alpha)$ with $k \in \{1,2\}$ there exists $c_h \in \mathcal{CC}(P_h, \alpha)$ with $h \in \{1,2\} - \{k\}$ such that:*

$$trace(c_k) = trace(c_h)$$
$$time_{\mathrm{a}}(c_k) = time_{\mathrm{a}}(c_h)$$

and:

$$rate_{\mathrm{t}}(P_{k,\mathrm{last}}, 0) = rate_{\mathrm{t}}(P_{h,\mathrm{last}}, 0)$$

with $P_{k,\mathrm{last}}$ (resp. $P_{h,\mathrm{last}}$) being the last configuration of c_k (resp. c_h). ■

6.2 Alternative Characterizations

Similarly to $\sim_{\mathrm{MT}}$, it turns out that $\sim_{\mathrm{MTr}}$ has an alternative characterization based on the probability distribution of executing a trace within a certain sequence of amounts of time. A consequence of this characterization is that considering the (more accurate) stepwise durations of the computations leads to the same equivalence as considering the (easier to work with) stepwise average durations of the computations. This justifies the use of expected values instead of random variables in the definition of $\sim_{\mathrm{MTr}}$.

Definition 49. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. We say that P_1 is Markovian distribution-trace equivalent to P_2, written $P_1 \sim_{\mathrm{MTr,d}} P_2$, iff for all traces $\alpha \in Name^*$ and sequences $\theta \in (\mathbb{R}_{>0})^*$ of amounts of time:*

$$prob_{\mathrm{d}}(\mathcal{CC}(P_1, \alpha), \theta) = prob_{\mathrm{d}}(\mathcal{CC}(P_2, \alpha), \theta)$$ ■

Theorem 11. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. Then:*

$$P_1 \sim_{\mathrm{MTr,d}} P_2 \iff P_1 \sim_{\mathrm{MTr}} P_2$$ ■

6.3 Other Markovian Trace-Based Equivalences

Like for classical trace equivalence, it is possible to define some variants of $\sim_{\mathrm{MTr}}$, which are based on the notions of completed trace, failure set, ready set, failure trace, and ready trace, respectively.

These variants were originally conceived to overcome some drawbacks of classical trace equivalence. Completed traces are traces ending up in deadlock states, so considering them apart is useful to make classical trace equivalence sensitive to deadlock. A failure set is a set of names of actions that cannot be executed in a certain state, and a failure trace is a trace extended at each step with a failure set. Considering failures makes classical trace equivalence sensitive to safety properties. Likewise, a ready set is the set of the names of all the actions that must be executable in a certain state, and a ready trace is a trace extended at each step with a ready set. Considering readies makes classical trace equivalence sensitive to liveness properties.

Definition 50. *Let $P \in \mathcal{P}_{\mathrm{C,pc}}$, $c \in \mathcal{C}_{\mathrm{f}}(P)$, and $\alpha \in Name^*$. We say that c is a maximal computation compatible with α iff $c \in \mathcal{CC}(P, \alpha)$ and the last configuration of c is deadlocked. We denote by $\mathcal{MCC}(P, \alpha)$ the multiset of the finite-length maximal computations of P that are compatible with α.* ■

Definition 51. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. We say that P_1 is Markovian completed-trace equivalent to P_2, written $P_1 \sim_{\mathrm{MTr,c}} P_2$, iff for all traces $\alpha \in Name^*$ and sequences $\theta \in (\mathbf{R}_{>0})^*$ of average amounts of time:*

$$prob(\mathcal{CC}_{\leq\theta}(P_1, \alpha)) = prob(\mathcal{CC}_{\leq\theta}(P_2, \alpha))$$
$$prob(\mathcal{MCC}_{\leq\theta}(P_1, \alpha)) = prob(\mathcal{MCC}_{\leq\theta}(P_2, \alpha))$$

■

Definition 52. *Let $P \in \mathcal{P}_{\mathrm{C,pc}}$, $c \in \mathcal{C}_{\mathrm{f}}(P)$, and $\varphi \equiv (\alpha, \mathcal{F}) \in Name^* \times 2^{Name}$. We say that c is a failure computation compatible with φ iff $c \in \mathcal{CC}(P, \alpha)$ and the last configuration of c cannot execute any action whose name belongs to the failure set $\mathcal{F}$. We denote by $\mathcal{FCC}(P, \varphi)$ the multiset of the finite-length failure computations of P that are compatible with φ.* ■

Definition 53. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. We say that P_1 is Markovian failure equivalent to P_2, written $P_1 \sim_{\mathrm{MF}} P_2$, iff for all traces with final failure set $\varphi \in Name^* \times 2^{Name}$ and sequences $\theta \in (\mathbf{R}_{>0})^*$ of average amounts of time:*

$$prob(\mathcal{FCC}_{\leq\theta}(P_1, \varphi)) = prob(\mathcal{FCC}_{\leq\theta}(P_2, \varphi))$$

■

Definition 54. *Let $P \in \mathcal{P}_{\mathrm{C,pc}}$, $c \in \mathcal{C}_{\mathrm{f}}(P)$, and $\rho \equiv (\alpha, \mathcal{R}) \in Name^* \times 2^{Name}$. We say that c is a ready computation compatible with ρ iff $c \in \mathcal{CC}(P, \alpha)$ and the set of the names of all the actions executable by the last configuration of c coincides with the ready set $\mathcal{R}$. We denote by $\mathcal{RCC}(P, \rho)$ the multiset of the finite-length ready computations of P that are compatible with ρ.* ■

Definition 55. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. We say that P_1 is Markovian ready equivalent to P_2, written $P_1 \sim_{\mathrm{MR}} P_2$, iff for all traces with final ready set $\rho \in Name^* \times 2^{Name}$ and sequences $\theta \in (\mathbf{R}_{>0})^*$ of average amounts of time:*

$$prob(\mathcal{RCC}_{\leq\theta}(P_1, \rho)) = prob(\mathcal{RCC}_{\leq\theta}(P_2, \rho))$$

■

Definition 56. *Let $P \in \mathcal{P}_{\mathrm{C,pc}}$, $c \in \mathcal{C}_{\mathrm{f}}(P)$, and $\phi \in (Name \times 2^{Name})^*$. We say that c is a failure-trace computation compatible with ϕ iff c is compatible with the trace component of ϕ and each configuration of c cannot execute any action whose name belongs to the corresponding failure set in the failure component of ϕ. We denote by $\mathcal{FTCC}(P, \phi)$ the multiset of the finite-length failure-trace computations of P that are compatible with ϕ.* ■

Definition 57. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. We say that P_1 is Markovian failure-trace equivalent to P_2, written $P_1 \sim_{\mathrm{MFTr}} P_2$, iff for all failure traces $\phi \in (Name \times 2^{Name})^*$ and sequences $\theta \in (\mathbf{R}_{>0})^*$ of average amounts of time:*

$$prob(\mathcal{FTCC}_{\leq\theta}(P_1, \phi)) = prob(\mathcal{FTCC}_{\leq\theta}(P_2, \phi))$$

■

Definition 58. *Let $P \in \mathcal{P}_{\mathrm{C,pc}}$, $c \in \mathcal{C}_{\mathrm{f}}(P)$, and $\varrho \in (Name \times 2^{Name})^*$. We say that c is a ready-trace computation compatible with ϱ iff c is compatible with the trace component of ϱ and the sets of the names of all the actions executable by the configurations of c coincide with the corresponding ready sets in the ready component of ϱ. We denote by $\mathcal{RTCC}(P, \varrho)$ the multiset of the finite-length ready-trace computations of P that are compatible with ϱ.* ■

Definition 59. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. We say that P_1 is Markovian ready-trace equivalent to P_2, written $P_1 \sim_{\mathrm{MRTr}} P_2$, iff for all ready traces $\varrho \in (Name \times 2^{Name})^*$ and sequences $\theta \in (\mathbf{R}_{>0})^*$ of average amounts of time:*

$$prob(\mathcal{RTCC}_{\leq\theta}(P_1, \varrho)) = prob(\mathcal{RTCC}_{\leq\theta}(P_2, \varrho))$$

■

Similarly to the probabilistic case [36, 35], in the Markovian framework trace equivalence becomes deadlock sensitive, hence $\sim_{\mathrm{MTr}}$ coincides with Markovian completed-trace equivalence. Moreover, Markovian failure equivalence coincides with Markovian ready equivalence – with both coinciding with $\sim_{\mathrm{MT}}$ – and Markovian failure-trace equivalence coincides with Markovian ready-trace equivalence. As a consequence, the Markovian linear-time/branching-time spectrum turns out to be more condensed than the nondeterministic one [25].

Theorem 12. *The Markovian linear-time/branching-time spectrum is:*

$\sim_{\mathrm{MB}} \subset \sim_{\mathrm{MRTr}} = \sim_{\mathrm{MFTr}} \subset \sim_{\mathrm{MR}} = \sim_{\mathrm{MT}} = \sim_{\mathrm{MTr,e}} = \sim_{\mathrm{MF}} \subset \sim_{\mathrm{MTr,c}} = \sim_{\mathrm{MTr}}$ ■

6.4 Exactness

From the point of view of the induced CTMC-level aggregation, nothing changes when moving from $\sim_{\mathrm{MT}}$ to $\sim_{\mathrm{MTr}}$.

Theorem 13. *$\sim_{\mathrm{MTr}}$ induces the same CTMC-level aggregation as $\sim_{\mathrm{MT}}$.* ■

Corollary 3. *The CTMC-level aggregation induced by $\sim_{\mathrm{MTr}}$ is exact.* ■

6.5 Congruence

Markovian trace equivalence is a congruence with respect to all the operators of SMPC. In particular, we have what follows.

Theorem 14. *Let $P_1, P_2 \in \mathcal{P}_{\mathrm{S}}$. Whenever $P_1 \sim_{\mathrm{MTr}} P_2$, then:*

1. *$<a, \lambda>.P_1 \sim_{\mathrm{MTr}} <a, \lambda>.P_2$ for all $<a, \lambda> \in Act_{\mathrm{S}}$.*
2. *$P_1 + P \sim_{\mathrm{MTr}} P_2 + P$ and $P + P_1 \sim_{\mathrm{MTr}} P + P_2$ for all $P \in \mathcal{P}_{\mathrm{S}}$.* ■

Unfortunately, similarly to the probabilistic case [36], $\sim_{\mathrm{MTr}}$ is not a congruence with respect to parallel composition. Consider for instance the following two Markovian trace equivalent process terms:

$$P_1 \equiv <a, \lambda_1>.<b, \mu>.P' + <a, \lambda_2>.<c, \mu>.P''$$
$$P_2 \equiv <a, \lambda_1 + \lambda_2>.(<b, \tfrac{\lambda_1}{\lambda_1+\lambda_2} \cdot \mu>.P' + <c, \tfrac{\lambda_2}{\lambda_1+\lambda_2} \cdot \mu>.P'')$$

where $b \neq c$. If we place each of them in the following context:

$$\_ \|_{\{a,b,c\}} <a, *_1>.<b, *_1>.\underline{0}$$

we obtain two performance-closed process terms – which we call Q_1 and Q_2 – that are no longer Markovian trace equivalent.

In fact, the following trace:

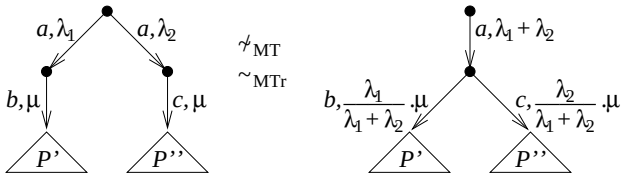
$$\alpha \equiv a \circ b$$

can distinguish between Q_1 and Q_2. The reason is that the only computation of

Q_1 compatible with α is formed by a transition labeled with $<a, \lambda_1>$ followed by a transition labeled with $<b, \mu>$, which has execution probability $\frac{\lambda_1}{\lambda_1+\lambda_2}$ and stepwise average duration $\frac{1}{\lambda_1+\lambda_2} \circ \frac{1}{\mu}$. By contrast, the only computation of Q_2 compatible with α is formed by a transition labeled with $<a, \lambda_1 + \lambda_2>$ followed by a transition labeled with $<b, \frac{\lambda_1}{\lambda_1+\lambda_2} \cdot \mu>$, which has execution probability 1 and stepwise average duration $\frac{1}{\lambda_1+\lambda_2} \circ \frac{\lambda_1+\lambda_2}{\lambda_1 \cdot \mu}$.

6.6 Axiomatization

Markovian trace equivalence is strictly coarser than Markovian testing equivalence, hence the axioms of $\sim_{\mathrm{MT}}$ are still valid for $\sim_{\mathrm{MTr}}$ over $\mathcal{P}_{\mathrm{S}}$, but not complete. In fact, the two process terms depicted below (with $b \neq c$):



show that, when moving from $\sim_{\mathrm{MT}}$ to $\sim_{\mathrm{MTr}}$, the action prefix operator tends to become left-distributive with respect to the alternative composition operator. More precisely, choices can be deferred as long as they are related to branches starting with actions having the same name that are followed by terms having the same total exit rate. Note that the names and the total rates of the initial actions of such derivative terms can be different in the various branches.

The two terms above constitute the simplest instance of an axiom schema that characterizes $\sim_{\mathrm{MTr}}$, which is more liberal than the one characterizing $\sim_{\mathrm{MT}}$. The axiom schema is the following:

$$\boxed{\sum_{i \in I} <a, \lambda_i>. \sum_{j \in J_i} <b_{i,j}, \mu_{i,j}>.P_{i,j} = <a, \sum_{k \in I} \lambda_k>. \sum_{i \in I} \sum_{j \in J_i} <b_{i,j}, \frac{\lambda_i}{\Sigma_{k \in I} \lambda_k} \cdot \mu_{i,j}>.P_{i,j}}$$

where:

- I is a finite index set with $|I| \geq 2$.
- J_i is a finite index set for all $i \in I$, with the related summation being $\underline{0}$ whenever $J_i = \emptyset$.
- For all $i_1, i_2 \in I$:

$$\boxed{\sum_{j \in J_{i_1}} \mu_{i_1,j} = \sum_{j \in J_{i_2}} \mu_{i_2,j}}$$

  with each summation being zero whenever its index set is empty.

6.7 Logical Characterization

Markovian trace equivalence has a modal logic characterization over $\mathcal{P}_{\mathrm{C,pc}}$ similar to the one for Markovian testing equivalence, in which the diamond operator is no longer dependent from the environment.

Definition 60. *The set of the formulas of* $\mathrm{HML}_{\mathrm{MTr}}$ *is generated by the following syntax:*

$$\boxed{\phi ::= \mathrm{true} \mid \langle a \rangle \phi}$$

where $a \in Name$. ■

Definition 61. *The interpretation function* $[\![.]\!]_{\mathrm{MTr}}$ *of* $\mathrm{HML}_{\mathrm{MTr}}$ *over* $\mathcal{P}_{\mathrm{C,pc}} \times (\mathbf{R}_{>0})^*$ *is defined by structural induction as follows:*

$$\boxed{\begin{array}{l} [\![\mathrm{true}]\!]_{\mathrm{MTr}}(P,\theta) = 1 \\ [\![\langle a \rangle \phi]\!]_{\mathrm{MTr}}(P,\theta) = \begin{cases} 0 & \text{if } \theta = \varepsilon \vee \frac{1}{rate_{\mathrm{t}}(P,0)} > \theta[1] \\ \displaystyle\sum_{P \xrightarrow{a,\lambda} P'} \frac{\lambda}{rate_{\mathrm{t}}(P,0)} \cdot [\![\phi]\!]_{\mathrm{MTr}}(P',\theta') & \\ & \text{if } \theta = t \circ \theta' \wedge \frac{1}{rate_{\mathrm{t}}(P,0)} \leq t \end{cases} \end{array}}$$

where the summation is taken to be zero whenever there are no a*-transitions departing from* P. ■

Theorem 15. *Let* $P_1, P_2 \in \mathcal{P}_{\mathrm{C,pc}}$. *Then:*

$$P_1 \sim_{\mathrm{MTr}} P_2 \iff \forall \phi \in \mathrm{HML}_{\mathrm{MTr}}.\, \forall \theta \in (\mathbf{R}_{>0})^*.\, [\![\phi]\!]_{\mathrm{MTr}}(P_1,\theta) = [\![\phi]\!]_{\mathrm{MTr}}(P_2,\theta)$$ ■

6.8 Verification Complexity

Markovian trace equivalence can be decided in polynomial time because two action-labeled CTMCs are Markovian trace equivalent iff their corresponding embedded action-labeled DTMCs (with suitably enriched labels) are probabilistic trace equivalent, with the latter being decidable in polynomial time [35].

Similarly to the verification of $\sim_{\mathrm{MT}}$, the algorithm to check whether $P_1 \sim_{\mathrm{MTr}} P_2$ thus proceeds as follows:

1. Transform $[\![P_1]\!]$ and $[\![P_2]\!]$ into their corresponding embedded discrete-time versions:
   (a) Divide the rate of each transition by the total exit rate of its source state.
   (b) Augment the name of each transition with the total exit rate of its source state.
2. Apply the algorithm of [47] to check the embedded discrete-time versions of $[\![P_1]\!]$ and $[\![P_2]\!]$ for probabilistic language equivalence by considering each of their states as being an accepting state.

The time complexity is $O(n^4)$, where n is the number of states of the disjoint union of $[\![P_1]\!]$ and $[\![P_2]\!]$.

7 Conclusion

In this survey we have recalled the definitions and the properties of Markovian behavioral equivalences. Besides providing information about the Markovian linear-time/branching-time spectrum, we have compared Markovian bisimilarity, Markovian testing equivalence, and Markovian trace equivalence with respect to a number of criteria, as summarized below:

	exact aggregation	*congruence property*	*sound & complete axiomatization*	*logical characteriz.*	*verification complexity*
$\sim_{\mathrm{MB}}$	*OK*	*OK*	*OK*	*OK*	$O(m \cdot \log n)$
$\sim_{\mathrm{MT}}$	*OK*	*OK*	*OK*	*OK*	$O(n^5)$
$\sim_{\mathrm{MTr}}$	*OK*	OK_{SMPC}	OK_{SMPC}	*OK*	$O(n^4)$

As can be noted, $\sim_{\mathrm{MB}}$ is satisfactory with respect to all the criteria, although it is often too discriminating. A good alternative may be $\sim_{\mathrm{MT}}$, as it encodes the viewpoint of an external observer and is more flexible with respect to branching points. By contrast, $\sim_{\mathrm{MTr}}$ cannot be considered as a valid alternative, as it fails to be a congruence with respect to parallel composition.

It is also worth emphasizing the exactness of the CTMC-level aggregations induced by each of the three considered Markovian behavioral equivalences. This means that $\sim_{\mathrm{MB}}$, $\sim_{\mathrm{MT}}$, and $\sim_{\mathrm{MTr}}$ are all meaningful for performance evaluation purposes. Besides justifying the investigation of the other properties, this can be exploited in practice. For instance, such Markovian behavioral equivalences can be used to aggregate the state space of a model by taking advantage of symmetries within the model [24], or to reduce the state space of a model before applying analysis techniques like model checking [45], without altering the performance properties to be assessed.

We conclude by mentioning some open problems in the field of Markovian behavioral equivalences:

- In our framework based on an asymmetric action synchronization discipline, while $\sim_{\mathrm{MB}}$ is defined over non-performance-closed terms too, this is not the case for $\sim_{\mathrm{MT}}$ and $\sim_{\mathrm{MTr}}$. Finding a way to extend their definitions so that it is still possible to determine the execution probability and the stepwise average duration of the computations in the presence of passive transitions is highly desirable.
- The exactness of the CTMC-level aggregations induced by $\sim_{\mathrm{MB}}$, $\sim_{\mathrm{MT}}$, and $\sim_{\mathrm{MTr}}$ establishes a strong connection between concurrency theory and Markov chain theory, which in particular gives rise to a process algebraic characterization of the aggregations themselves. The question here is whether the aggregation induced by $\sim_{\mathrm{MT}}$ and $\sim_{\mathrm{MTr}}$ is the coarsest exact non-trivial one that can be obtained, or whether it can be further extended.
- The set of logical operators necessary to characterize Markovian behavioral equivalences decreases as the discriminating power of the equivalences

decreases. However, the logical characterization of $\sim_{\mathrm{MT}}$ relies on a non-standard variant of the diamond operator. What if we replace the non-standard operator with the standard one and we reintroduce logical conjunction? We claim that this may result in an alternative logical characterization of $\sim_{\mathrm{MT}}$ if the diamonds occurring in a conjunction are independent of each other, i.e. if the names of the related actions are all different [39].
- The verification algorithm for $\sim_{\mathrm{MB}}$ can also be used as a minimization algorithm (with respect to $\sim_{\mathrm{MB}}$), whereas this is not the case for the verification algorithms for $\sim_{\mathrm{MT}}$ and $\sim_{\mathrm{MTr}}$. As far as testing is concerned, it should be investigated whether the algorithm for verifying classical testing equivalence [20] can be adapted to the Markovian framework. The reason is that this algorithm reduces the verification of classical testing equivalence over standard state-transition models to the verification of (a generalization of) classical bisimilarity over transformed models inspired by acceptance trees [28], hence it can be exploited for minimization purposes as well.
- We would like to assess whether $\sim_{\mathrm{MT}}$ can be used also for quantitative analysis. So far, it supports a merely qualitative analysis, in the sense that it only allows one to establish whether two models pass an arbitrary test in the same way. What we envision is the possibility of identifying classes of tests that are related to specific performability measures, which may thus be used to evaluate models with respect to certain indices of interest.
- Finally, it would be interesting to develop weaker versions of $\sim_{\mathrm{MB}}$, $\sim_{\mathrm{MT}}$, and $\sim_{\mathrm{MTr}}$. In this survey we have seen $\approx_{\mathrm{IMB}}$ and $\approx_{\mathrm{EMB}}$, which abstract from invisible immediate actions. However, one could also consider the possibility of abstracting from invisible actions that are exponentially timed, which amounts to understand to what extent exponential delays can be neglected. This issue has been tackled in [32, 11, 13, 4], but none of the proposals seems to induce an exact aggregation at the CTMC level.

References

1. M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis, *"Modelling with Generalized Stochastic Petri Nets"*, John Wiley & Sons, 1995.
2. J.C.M. Baeten and W.P. Weijland, *"Process Algebra"*, Cambridge University Press, 1990.
3. C. Baier, J.-P. Katoen, and H. Hermanns, *"Approximate Symbolic Model Checking of Continuous Time Markov Chains"*, in Proc. of the *10th Int. Conf. on Concurrency Theory (CONCUR 1999)*, LNCS 1664:146-162, Eindhoven (The Netherlands), 1999.
4. C. Baier, J.-P. Katoen, H. Hermanns, and V. Wolf, *"Comparative Branching-Time Semantics for Markov Chains"*, in Information and Computation 200:149-214, 2005.
5. S. Balsamo, M. Bernardo, and M. Simeoni, *"Performance Evaluation at the Software Architecture Level"*, in *Formal Methods for Software Architectures*, LNCS 2804:207-258, 2003.
6. J.A. Bergstra, A. Ponse, and S.A. Smolka (eds.), *"Handbook of Process Algebra"*, Elsevier, 2001.

7. M. Bernardo, *"Non-Bisimulation-Based Markovian Behavioral Equivalences"*, to appear in Journal of Logic and Algebraic Programming.
8. M. Bernardo and A. Aldini, *"Weak Markovian Bisimilarity: Abstracting from Prioritized/Weighted Internal Immediate Actions"*, submitted for publication.
9. M. Bernardo and S. Botta, *"A Survey of Modal Logics Characterizing Behavioral Equivalences for Nondeterministic and Stochastic Systems"*, to appear in Mathematical Structures in Computer Science.
10. M. Bernardo and M. Bravetti, *"Performance Measure Sensitive Congruences for Markovian Process Algebras"*, in Theoretical Computer Science 290:117-160, 2003.
11. M. Bernardo and R. Cleaveland, *"A Theory of Testing for Markovian Processes"*, in Proc. of the *11th Int. Conf. on Concurrency Theory (CONCUR 2000)*, LNCS 1877:305-319, State College (PA), 2000.
12. H. Bohnenkamp, P.R. D'Argenio, H. Hermanns, and J.-P. Katoen, "MODEST*: A Compositional Modeling Formalism for Hard and Softly Timed Systems"*, in IEEE Trans. on Software Engineering 32:812-830, 2006.
13. M. Bravetti, *"Revisiting Interactive Markov Chains"*, in Proc. of the *3rd Int. Workshop on Models for Time-Critical Systems (MTCS 2002)*, ENTCS 68(5):1-20, Brno (Czech Republic), 2002.
14. M. Bravetti, M. Bernardo, and R. Gorrieri, *"A Note on the Congruence Proof for Recursion in Markovian Bisimulation Equivalence"*, in Proc. of the *6th Int. Workshop on Process Algebra and Performance Modelling (PAPM 1998)*, pp. 153-164, Nice (France), 1998.
15. P. Buchholz, *"Exact and Ordinary Lumpability in Finite Markov Chains"*, in Journal of Applied Probability 31:59-75, 1994.
16. P. Buchholz, *"Markovian Process Algebra: Composition and Equivalence"*, in Proc. of the *2nd Int. Workshop on Process Algebra and Performance Modelling (PAPM 1994)*, Technical Report 27-4, pp. 11-30, Erlangen (Germany), 1994.
17. I. Christoff, *"Testing Equivalences and Fully Abstract Models for Probabilistic Processes"*, in Proc. of the *1st Int. Conf. on Concurrency Theory (CONCUR 1990)*, LNCS 458:126-140, Amsterdam (The Netherlands), 1990.
18. G. Clark, S. Gilmore, and J. Hillston, *"Specifying Performance Measures for PEPA"*, in Proc. of the *5th AMAST Int. Workshop on Formal Methods for Real Time and Probabilistic Systems (ARTS 1999)*, LNCS 1601:211-227, Bamberg (Germany), 1999.
19. R. Cleaveland, Z. Dayar, S.A. Smolka, and S. Yuen, *"Testing Preorders for Probabilistic Processes"*, in Information and Computation 154:93-148, 1999.
20. R. Cleaveland and M. Hennessy, *"Testing Equivalence as a Bisimulation Equivalence"*, in Formal Aspects of Computing 5:1-20, 1993.
21. R. De Nicola and M. Hennessy, *"Testing Equivalences for Processes"*, in Theoretical Computer Science 34:83-133, 1983.
22. R. De Nicola, D. Latella, and M. Massink, *"Formal Modeling and Quantitative Analysis of* KLAIM*-Based Mobile Systems"*, in Proc. of the *20th ACM Symp. on Applied Computing (SAC 2005)*, ACM Press, pp. 428-435, Santa Fe (NM), 2005.
23. S. Derisavi, H. Hermanns, and W.H. Sanders, *"Optimal State-Space Lumping in Markov Chains"*, in Information Processing Letters 87:309-315, 2003.
24. S. Gilmore, J. Hillston, and M. Ribaudo, *"An Efficient Algorithm for Aggregating PEPA Models"*, in IEEE Trans. on Software Engineering 27:449-464, 2001.
25. R.J. van Glabbeek, *"The Linear Time - Branching Time Spectrum I"*, in [6], pp. 3-99, 2001.

26. R.J. van Glabbeek, S.A. Smolka, and B. Steffen, *"Reactive, Generative and Stratified Models of Probabilistic Processes"*, in Information and Computation 121:59-80, 1995.
27. N. Götz, U. Herzog, and M. Rettelbach, *"Multiprocessor and Distributed System Design: The Integration of Functional Specification and Performance Analysis Using Stochastic Process Algebras"*, in Proc. of the *16th Int. Symp. on Computer Performance Modelling, Measurement and Evaluation (PERFORMANCE 1993)*, LNCS 729:121-146, Roma (Italy), 1993.
28. M. Hennessy, *"Acceptance Trees"*, in Journal of the ACM 32:896-928, 1985.
29. M. Hennessy and R. Milner, *"Algebraic Laws for Nondeterminism and Concurrency"*, in Journal of the ACM 32:137-162, 1985.
30. H. Hermanns, *"Interactive Markov Chains"*, LNCS 2428, 2002.
31. H. Hermanns and M. Rettelbach, *"Syntax, Semantics, Equivalences, and Axioms for MTIPP"*, in Proc. of the *2nd Int. Workshop on Process Algebra and Performance Modelling (PAPM 1994)*, Technical Report 27-4, pp. 71-87, Erlangen (Germany), 1994.
32. J. Hillston, *"A Compositional Approach to Performance Modelling"*, Cambridge University Press, 1996.
33. C.A.R. Hoare, *"Communicating Sequential Processes"*, Prentice Hall, 1985.
34. R.A. Howard, *"Dynamic Probabilistic Systems"*, John Wiley & Sons, 1971.
35. D.T. Huynh and L. Tian, *"On Some Equivalence Relations for Probabilistic Processes"*, in Fundamenta Informaticae 17:211-234, 1992.
36. C.-C. Jou and S.A. Smolka, *"Equivalences, Congruences, and Complete Axiomatizations for Probabilistic Processes"*, in Proc. of the *1st Int. Conf. on Concurrency Theory (CONCUR 1990)*, LNCS 458:367-383, Amsterdam (The Netherlands), 1990.
37. P.C. Kanellakis and S.A. Smolka, *"CCS Expressions, Finite State Processes, and Three Problems of Equivalence"*, in Information and Computation 86:43-68, 1990.
38. L. Kleinrock, *"Queueing Systems"*, John Wiley & Sons, 1975.
39. M.Z. Kwiatkowska and G.J. Norman, *"A Testing Equivalence for Reactive Probabilistic Processes"*, in Proc. of the *2nd Int. Workshop on Expressiveness in Concurrency (EXPRESS 1998)*, ENTCS 16(2):114-132, Nice (France), 1998.
40. K.G. Larsen and A. Skou, *"Bisimulation through Probabilistic Testing"*, in Information and Computation 94:1-28, 1991.
41. R. Milner, *"Communication and Concurrency"*, Prentice Hall, 1989.
42. R. Paige and R.E. Tarjan, *"Three Partition Refinement Algorithms"*, in SIAM Journal on Computing 16:973-989, 1987.
43. D. Park, *"Concurrency and Automata on Infinite Sequences"*, in Proc. of the *5th GI Conf. on Theoretical Computer Science*, LNCS 104:167-183, 1981.
44. C. Priami, *"Stochastic π-Calculus"*, in Computer Journal 38:578-589, 1995.
45. J. Sproston and S. Donatelli, *"Backward Bisimulation in Markov Chain Model Checking"*, in IEEE Trans. on Software Engineering 32:531-546, 2006.
46. E.W. Stark, R. Cleaveland, and S.A. Smolka, *"A Process-Algebraic Language for Probabilistic I/O Automata"*, in Proc. of the *14th Int. Conf. on Concurrency Theory (CONCUR 2003)*, LNCS 2761:189-203, Marseille (France), 2003.
47. W.-G. Tzeng, *"A Polynomial-Time Algorithm for the Equivalence of Probabilistic Automata"*, in SIAM Journal on Computing 21:216-227, 1992.
48. M. Woodside, *"From Annotated Software Designs (UML SPT/MARTE) to Model Formalisms"*, to appear in *Formal Methods for Performance Evaluation*, LNCS 4486, 2007.

49. V. Wolf, C. Baier, and M. Majster-Cederbaum, *"Trace Machines for Observing Continuous-Time Markov Chains"*, in Proc. of the *3rd Int. Workshop on Quantitative Aspects of Programming Languages (QAPL 2005)*, ENTCS 153(2):259-277, Edinburgh (UK), 2005.