Markovian Testing and Trace Equivalences Exactly Lump More Than Markovian Bisimilarity

Marco Bernardo

Università di Urbino "Carlo Bo"
Istituto di Scienze e Tecnologie dell'Informazione
Piazza della Repubblica 13, 61029 Urbino, Italy
Email: bernardo@sti.uniurb.it

Abstract

The notion of equivalence that is typically used to relate Markovian process terms and to reduce their underlying state spaces is Markovian bisimilarity. The reason is that, besides being a congruence, Markovian bisimilarity is consistent with ordinary lumping, an exact aggregation for Markov chains. In this paper we show that two non-bisimulation-based Markovian behavioral equivalences – Markovian testing equivalence and Markovian trace equivalence – induce at the Markov chain level an aggregation strictly coarser than ordinary lumping that is still exact.

1 Introduction

In order to account for performance aspects, in the last two decades algebraic process calculi have been extended so that stochastic processes can be associated with their terms. In this field, the focus has primarily been on equipping process terms with performance models in the form of continuous-time Markov chains (CTMCs). Several Markovian process calculi have been proposed in the literature (see, e.g., [12,10,3] and the references therein). Although they differ for the action representation – durational actions vs. instantaneous actions separated from time passing – as well as for the synchronization discipline – asymmetric vs. symmetric – such Markovian process calculi share a common feature: Markovian bisimulation equivalence.

Markovian bisimilarity [12] is a semantic theory building on [14,13] that has proven to be useful to relate Markovian process terms and to reduce their underlying state spaces. The basic idea is that two Markovian bisimilar process terms are able to mimic each other's behavior both from the functional and the performance viewpoint. The reason of the success of Markovian bisimilarity is that it enjoys several nice properties, both on the algebraic side and

This paper is electronically published in
Electronic Notes in Theoretical Computer Science
URL: www.elsevier.nl/locate/entcs

on the performance side. First, it is a congruence with respect to all the typical process algebraic operators [12], thus allowing for compositional reasoning and compositional state space reduction. Second, it has a sound and complete axiomatization [11], which elucidates the fundamental equational laws on which Markovian bisimilarity relies. Third, it is consistent with ordinary lumping [12], which implies the usefulness of Markovian bisimilarity from the performance viewpoint. Ordinary lumping [5] is a notion of aggregation for Markov chains that is exact, i.e. the stationary/transient probability of being in a macrostate of an ordinarily lumped Markov chain is the sum of the stationary/transient probabilities of being in one of the constituent microstates of the original Markov chain. Thus, whenever two process terms are Markovian bisimilar, they are guaranteed to possess the same performance characteristics.

In the continuous-time setting research has mainly focused on branching-time equivalences [1] due to their connection with ordinary lumping. Only recently linear-time equivalences and testing scenarios have been investigated as well in the continuous-time case. In [4] Markovian testing equivalence has been proposed on the basis of [8,6]. Unlike Markovian bisimilarity, in which the ability to mimic the functional and performance behavior is taken into account, Markovian testing equivalence relies on a generic notion of efficiency, which is based on the probability of performing a test-driven computation within a certain average amount of time. In [16] a behavioral equivalence for a process algebraic language based on probabilistic I/O automata has been considered, which is parameterized with respect to generic observables that associate real numbers with rated traces. In [17] the Markovian variants of several linear-time equivalences – trace equivalence, complete trace equivalence, failure trace equivalence, ready trace equivalence – have been investigated by means of push-button experiments conducted with appropriate Markovian trace machines.

All the Markovian behavioral equivalences defined in [4,16,17] are strictly coarser than Markovian bisimilarity, so at the CTMC level they result in aggregations that are strictly coarser than ordinary lumping. Although this can be helpful in practice to attack the state space explosion problem, we do not know whether such non-bisimulation-based Markovian behavioral equivalences are useful from the performance viewpoint. We are in fact facing the following open problem: are the CTMC-level aggregations induced by such equivalences exact? In other words, given two process terms that are related by one of these non-bisimulation-based Markovian behavioral equivalences, we do not know whether they possess the same performance characteristics or not.

The contribution of this paper is to solve the above open problem by showing that both Markovian testing equivalence and Markovian trace equivalence induce at the CTMC level an aggregation strictly coarser than ordinary lumping that is still exact. This result ensures that any two process terms that are Markovian testing or trace equivalent possess the same performance characteristics. A further consequence is that Markovian testing and trace equivalences turn out to aggregate more than Markovian bisimilarity while preserving the exactness of the aggregation.

The strategy adopted in this paper to prove the exact aggregation property is to demonstrate first that Markovian testing and trace equivalences have sound and complete axiomatizations, which in turn requires to prove first that Markovian testing and trace equivalences are congruences. These two side results are provided for a basic Markovian process calculus with durational actions, which generates all the CTMCs with as few operators as possible: the null term, the action prefix operator, the alternative composition operator, and the process invocation mechanism. This ensures the general validity of the exact aggregation property without complicating the proof of the two side results. Once the axiomatizations of Markovian testing and trace equivalences will have been obtained, we shall observe that they differ from the axiomatization of Markovian bisimilarity just for a new axiom schema subsuming one of the axioms of Markovian bisimilarity. As a consequence, in the proof of the exact aggregation property it will be necessary to concentrate only on the aggregations resulting from the application of this new axiom schema.

This paper is organized as follows. In Sect. 2 we introduce a basic Markovian process calculus and we recall Markovian bisimilarity. In Sect. 3 we present the definition of Markovian testing equivalence in a way that is slightly different from [4], then we prove that it is a congruence, has a sound and complete axiomatization, and induces a CTMC-level aggregation strictly coarser than ordinary lumping that is still exact. In Sect. 4 we present the definition of Markovian trace equivalence in a way that is slightly different from [17], then we prove the same congruence, axiomatization and exact aggregation results as for Markovian testing equivalence. Finally, in Sect. 5 we report some concluding remarks. The proofs of the results can be found in [2].

2 Markovian Process Calculi and Bisimilarity

In this section we introduce a Markovian process calculus with durational actions, which generates all the CTMCs with as few operators as possible: the null term, the action prefix operator, the alternative composition operator, and the process invocation mechanism. After defining the syntax and the semantics for the calculus, which we call MPC, we recall Markovian bisimilarity and show its properties on the calculus.

2.1 Syntax

In MPC every action is durational, hence it is represented as a pair $<a, \lambda>$, where $a \in \mathit{AType}$ is the type of the action while $\lambda \in \mathbb{R}_{>0}$ is the rate of the exponential distribution characterizing the duration of the action. We denote by $\mathit{Act} = \mathit{AType} \times \mathbb{R}_{>0}$ the set of the actions of MPC. Unlike standard process theory, here we assume that all the actions are observable.

Definition 2.1 The set of the process terms of MPC is generated by the following syntax:

$$P ::= \underline{0} \mid {<}a, \lambda{>}.P \mid P + P \mid A$$

where A is a process constant defined through the (possibly recursive) equation $A \stackrel{\Delta}{=} P$. We denote by $\mathcal{P}$ the set of the closed and guarded process terms of MPC. ■

2.2 Semantics

The semantics for MPC can be defined in the usual operational style. As a consequence, the behavior of each process term is given by a multitransition system, whose states correspond to process terms and whose transitions – each of which has a multiplicity – are labeled with actions. From such a multitransition system the CTMC underlying the process term can easily be retrieved by (i) discarding the action types from the transition labels and (ii) collapsing all the transitions between the same two states into a single transition whose rate is the sum of the rates of the original transitions.

The null term $\underline{0}$ cannot execute any action, hence the corresponding labeled multitransition system is just a state with no transitions. Term ${<}a, \lambda{>}.P$ can execute an action of type a and average duration $1/\lambda$ and then behaves as P:

$$ {<}a, \lambda{>}.P \xrightarrow{a,\lambda} P $$

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

$$ \frac{P_1 \xrightarrow{a,\lambda} P'}{P_1 + P_2 \xrightarrow{a,\lambda} P'} \qquad \frac{P_2 \xrightarrow{a,\lambda} P'}{P_1 + P_2 \xrightarrow{a,\lambda} P'} $$

where the actions executable by P_1 and those executable by P_2 are considered to be in a race, hence each of them has an execution probability proportional to its rate. Finally, process constant A behaves as the right-hand side process term in the defining equation for A:

$$ \frac{P \xrightarrow{a,\lambda} P'}{A \xrightarrow{a,\lambda} P'} \text{ if } A \stackrel{\Delta}{=} P $$

2.3 Markovian Bisimilarity

The notion of equivalence that is typically used to reason on the process terms of a calculus like MPC is Markovian bisimilarity.

Definition 2.2 An equivalence relation $\mathcal{B} \subseteq \mathcal{P} \times \mathcal{P}$ is a Markovian bisimulation iff, whenever $(P_1, P_2) \in \mathcal{B}$, then for all action types $a \in \mathit{AType}$ and equivalence classes $C \in \mathcal{P}/\mathcal{B}$:

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

where for each $i = 1, 2$:

$$rate(P_i, a, C) = \sum \{\!| \lambda \mid \exists P' \in C.\, P_i \xrightarrow{a,\lambda} P' |\!\}$$

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

Markovian bisimilarity enjoys the following properties. First, it is a congruence with respect to all the operators of MPC. Second, it has a sound and complete axiomatization over MPC, which includes the following four axioms:

$$(\mathcal{A}_1) \qquad P_1 + P_2 = P_2 + P_1$$

$$(\mathcal{A}_2) \qquad (P_1 + P_2) + P_3 = P_1 + (P_2 + P_3)$$

$$(\mathcal{A}_3) \qquad P + \underline{0} = P$$

$$(\mathcal{A}_4) \qquad <a, \lambda_1>.P + <a, \lambda_2>.P = <a, \lambda_1 + \lambda_2>.P$$

Third, it is consistent with ordinary lumping. Whenever $P_1 \sim_{\mathrm{MB}} P_2$, then the two CTMCs underlying P_1 and P_2 are ordinarily lumping equivalent. Since ordinary lumping is an exact aggregation, P_1 and P_2 are guaranteed to possess the same performance characteristics. To be more precise, this is the case unless we consider performance measures that distinguish between ordinarily lumpable states by assigning them different rewards. The interested reader is referred to [3] for a complete treatment of this issue.

3 Markovian Testing Equivalence

In this section we start by revisiting the definition of Markovian testing equivalence given in [4]. Before doing that, we need to provide the syntax for the tests as well as a parallel composition operator. We then proceed by proving that Markovian testing equivalence is a congruence with respect to the operators of MPC, has a sound and complete axiomatization for the non-recursive terms of MPC, and induces a CTMC-level aggregation strictly coarser than ordinary lumping that is still exact.

3.1 Test Formalization and Parallel Composition

A test is an entity that interacts with a process term in order to expose a part of the behavior of the latter. The most convenient way to represent a test is through another process term, which interacts with the first one by means of a parallel composition operator that enforces synchronization on any observable action type. As a consequence, the semantic model of the interaction of a process term and a test will still be a multitransition system labeled with actions.

Since a test should be conducted in a finite number of steps, for the test formalization we restrict ourselves to process terms that are finite state and acyclic, hence no recursion is admitted within the tests. In other words, the labeled multitransition systems underlying the tests must have a finite dag-like structure.

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 a failure state. At the process calculus level, this amounts to replace $\underline{0}$ with the two zeroary operators "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 $\mathbb{N}_{>0}$.

In our Markovian framework, the interaction of a process term and a test should be closed with respect to the class of exponential distributions, i.e. it should not give rise to transitions whose rate cannot be expressed through a positive real number representing an exponential distribution. This strictly depends on the synchronization discipline that is adopted. Based on the terminology of [9], the simplest way to achieve exponential closure is to enforce the Markovian generative-reactive form of communication [3]. Therefore, only the so-called passive actions can occur within the tests. Passive actions have no duration associated with them. Instead, they are given positive real numbers interpreted as weights, which are used to make a probabilistic selection among a set of passive actions of the same type.

From the testing viewpoint, the idea is that 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 reacts by probabilistically selecting a passive action (if any) of the same type as the proposed exponentially timed one in order to participate in the interaction.

Definition 3.1 The set $\mathcal{T}$ of the tests is generated by the following syntax:

$$T ::= \mathrm{f} \mid \mathrm{s} \mid \sum_{i \in I} {<} a_i, *_{w_i} {>} . T_i$$

where I is a non-empty, finite index set, $a_i \in \mathit{AType} - \{\tau\}$, and $w_i \in \mathbb{R}_{>0}$. ■

The following operational rule defines the generative-reactive interaction of $P \in \mathcal{P}$ and $T \in \mathcal{T}$:

$$\frac{P \xrightarrow{a,\lambda} P' \qquad T \xrightarrow{a,*_w} T'}{P \parallel T \xrightarrow{a,\lambda \cdot w / \mathit{weight}(T,a)} P' \parallel T'}$$

where:

$$\mathit{weight}(T, a) = \sum \{\!| \, w \mid \exists T'. \, T \xrightarrow{a,*_w} T' \, |\!\}$$

3.2 Equivalence Definition

We now have all the ingredients to build the definition of Markovian testing equivalence. The first concept that we introduce is that of computation of the interaction of a process term and a test, which is a maximal sequence of transitions in the labeled multitransition system underlying the parallel composition of the process term and the test. Due to the restrictions previously imposed on the tests, all the considered computations will turn out to have a finite length.

Definition 3.2 The interaction system of $P \in \mathcal{P}$ and $T \in \mathcal{T}$ is process term $P \parallel T$, where we say that:

- A configuration is a state of the labeled multitransition system underlying $P \parallel T$.
- A configuration is successful (resp. failed) iff its test component is "s" (resp. "f").
- A computation is a maximal sequence of transitions:
$$P \parallel T \xrightarrow{a_1,\lambda_1} P_1 \parallel T_1 \xrightarrow{a_2,\lambda_2} \dots \xrightarrow{a_n,\lambda_n} P_n \parallel T_n$$
such that configuration $P_i \parallel T_i$ is neither successful nor failed for all $0 \leq i \leq n-1$.
- A computation is successful (resp. failed) iff so is its last configuration. A computation that is neither successful nor failed is said to be interrupted.

We denote by $\mathcal{C}(P,T)$ and $\mathcal{S}(P,T)$ the multisets of the computations and of the successful computations, respectively, of the interaction system of P and T. ■

The second concept that we introduce is a generic notion of efficiency, which is based on the probability of performing a test-driven computation within a given average amount of time. To this purpose, it is useful to recall that the average time taken by a state s of a CTMC to perform a transition – called the average sojourn time of s – is the inverse of the sum of the rates of all the transitions departing from s.

Definition 3.3 Let $P \in \mathcal{P}$, $T \in \mathcal{T}$, and $c \in \mathcal{C}(P,T)$. The execution probability and the average duration of c are defined by induction on the length of c as follows:

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

$$time(c) = \begin{cases} 0 & \text{if } length(c) = 0 \\ \frac{1}{rate_{\rm t}(P \parallel T)} + time(c') & \text{if } c \equiv P \parallel T \xrightarrow{a,\lambda} c' \end{cases}$$

where:

$$rate_{\rm t}(P \parallel T) = \sum \{\!| \lambda \mid \exists a, P', T'. \, P \parallel T \xrightarrow{a,\lambda} P' \parallel T' |\!\}$$

We also pose:

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

for all $C \subseteq \mathcal{C}(P,T)$ and:

$$C_{\leq t} = \{c \in C \mid time(c) \leq t\}$$

for all $C \subseteq \mathcal{C}(P, T)$ and $t \in \mathbb{R}_{\geq 0}$. ■

Definition 3.4 Let $P_1, P_2 \in \mathcal{P}$. We say that P_1 is Markovian testing equivalent to P_2, written $P_1 \sim_{\text{MT}} P_2$, iff for all tests $T \in \mathcal{T}$ and average amounts of time $t \in \mathbb{R}_{\geq 0}$:

$$prob(\mathcal{S}_{\leq t}(P_1, T)) = prob(\mathcal{S}_{\leq t}(P_2, T))$$ ■

3.3 Congruence Property

$\sim_{\text{MT}}$ turns out to be a congruence with respect to action prefix and alternative composition.

Theorem 3.5 *Let $P_1, P_2 \in \mathcal{P}$. Whenever $P_1 \sim_{\text{MT}} P_2$, then:*

(i) $<a, \lambda>.P_1 \sim_{\text{MT}} <a, \lambda>.P_2$ *for all* $<a, \lambda> \in Act$.

(ii) $P_1 + P \sim_{\text{MT}} P_2 + P$ *and* $P + P_1 \sim_{\text{MT}} P + P_2$ *for all* $P \in \mathcal{P}$. ■

3.4 Sound and Complete Axiomatization

As shown in [4], $\sim_{\text{MB}}$ is strictly contained in $\sim_{\text{MT}}$, hence the axioms $\mathcal{A}_1$-$\mathcal{A}_4$ of Sect. 2.3 are still valid for $\sim_{\text{MT}}$, but not complete. An example of process terms that are Markovian testing equivalent but not Markovian bisimilar is given by:

$$<a, \lambda_1>.<b, \mu>.P_1 + <a, \lambda_2>.<b, \mu>.P_2$$

and:

$$<a, \lambda_1 + \lambda_2>.\left(<b, \frac{\lambda_1}{\lambda_1 + \lambda_2} \cdot \mu>.P_1 + <b, \frac{\lambda_2}{\lambda_1 + \lambda_2} \cdot \mu>.P_2\right)$$

In fact, no test starting with a passive a-action possibly followed by a passive b-action can distinguish between the two terms, because in both terms the average time to perform an a-action followed by a b-action is $1/(\lambda_1 + \lambda_2) + 1/\mu$ and the probability of reaching P_1 (resp. P_2) is $\lambda_1/(\lambda_1 + \lambda_2)$ (resp. $\lambda_2/(\lambda_1 + \lambda_2)$). By contrast, there is no way to relate $<b, \mu>.P_1$ and $<b, \mu>.P_2$ with $<b, \lambda_1/(\lambda_1 + \lambda_2) \cdot \mu>.P_1 + <b, \lambda_2/(\lambda_1 + \lambda_2) \cdot \mu>.P_2$ through $\sim_{\text{MB}}$ if $P_1 \not\sim_{\text{MB}} P_2$.

It turns out that the two terms above constitute the simplest instance of an axiom schema $\mathcal{A}'_4$ subsuming $\mathcal{A}_4$ that we have to add to $\mathcal{A}_1$-$\mathcal{A}_3$ in order to obtain a sound and complete axiomatization of $\sim_{\text{MT}}$ over the set $\mathcal{P}_{\text{nr}}$ of the non-recursive process terms of $\mathcal{P}$. Such an axiomatization is given by the set $\mathcal{A}'$ of axioms shown in Table 1, where I and J_i are finite index sets with $|I| \geq 2$ (if $J_i = \emptyset$, the related summations are taken to be $\underline{0}$).

Theorem 3.6 *The deduction system $DED(\mathcal{A}')$ is sound and complete for $\sim_{\text{MT}}$ over $\mathcal{P}_{\text{nr}}$, i.e. for all $P_1, P_2 \in \mathcal{P}_{\text{nr}}$:*

$$\mathcal{A}' \vdash P_1 = P_2 \iff P_1 \sim_{\text{MT}} P_2$$ ■

$$(\mathcal{A}_1) \quad P_1 + P_2 = P_2 + P_1$$

$$(\mathcal{A}_2) \quad (P_1 + P_2) + P_3 = P_1 + (P_2 + P_3)$$

$$(\mathcal{A}_3) \quad P + \underline{0} = P$$

$$(\mathcal{A}'_4) \quad \sum_{i \in I} <a, \lambda_i>. \sum_{j \in J_i} <b_{i,j}, \mu_{i,j}>.P_{i,j} =$$

$$<a, \textstyle\sum_{k \in I} \lambda_k>. \displaystyle\sum_{i \in I} \sum_{j \in J_i} <b_{i,j}, \tfrac{\lambda_i}{\sum_{k \in I} \lambda_k} \cdot \mu_{i,j}>.P_{i,j}$$

if for all $i_1, i_2 \in I$:

$$\{b_{i_1,j} \mid j \in J_{i_1}\} = \{b_{i_2,j} \mid j \in J_{i_2}\} \equiv \{b_1, b_2, \dots, b_n\}$$

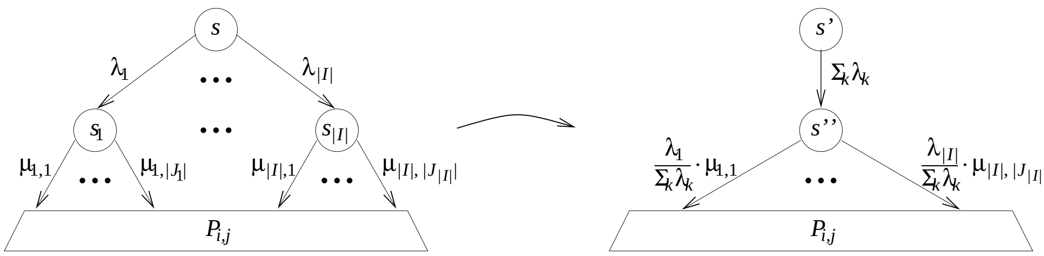
and for all $h = 1, \dots, n$:

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

Table 1
Axiomatization of $\sim_{\rm MT}$ over $\mathcal{P}_{\rm nr}$

3.5 Exact Aggregation Property

The axiomatization of $\sim_{\rm MT}$ over $\mathcal{P}_{\rm nr}$ differs from the one of $\sim_{\rm MB}$ only for the last axiom, thus we can concentrate on $\mathcal{A}'_4$ to study the aggregation induced by $\sim_{\rm MT}$ at the CTMC level. If we view $\mathcal{A}'_4$ as the following rewriting rule:



where for all $i_1, i_2 \in I$:

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

it turns out that $\mathcal{A}'_4$ aggregates $|I| \geq 2$ states into a single one. Now the question arises as to whether this kind of aggregation is exact, i.e. whether the stationary/transient probability of being in a macrostate of the aggregated CTMC on the right is the sum of the stationary/transient probabilities of being in one of the constituent microstates of the original CTMC on the left. A positive answer would entail the usefulness of $\sim_{\rm MT}$ for performance evaluation purposes, i.e. the preservation of the value of the performance measures across process terms that are Markovian testing equivalent.

Theorem 3.7 *The CTMC-level aggregation induced by* $\sim_{\rm MT}$ *is exact.* ■

4 Markovian Trace Equivalence

In this section we start by revisiting the definition of Markovian trace equivalence given in [17], then we prove that Markovian trace equivalence is a congruence with respect to the operators of MPC, has a sound and complete axiomatization for the non-recursive terms of MPC, and induces the same exact CTMC-level aggregation as Markovian testing equivalence.

4.1 Equivalence Definition

Unlike Markovian testing equivalence, given a process term $P \in \mathcal{P}$ in the case of Markovian trace equivalence we no longer have tests that interact with P. Instead, we directly consider the multiset $\mathcal{C}_{\rm f}(P)$ of the finite-length computations of P taken in isolation.

Definition 4.1 Let $P \in \mathcal{P}$ and $c \in \mathcal{C}_{\rm f}(P)$. The trace associated with the execution of c is defined by induction on the length of c as follows:

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

where ε is the empty trace. ■

Definition 4.2 Let $P \in \mathcal{P}$, $c \in \mathcal{C}_{\rm f}(P)$, and $\alpha \in AType^*$. 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. ■

Definition 4.3 Let $P \in \mathcal{P}$, $\alpha \in AType^*$, and $c \in \mathcal{CC}(P, \alpha)$. The execution probability and the average duration of c are defined by induction on the length of c as follows:

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

$$time(c) = \begin{cases} 0 & \text{if } length(c) = 0 \\ \frac{1}{rate_{\rm t}(P)} + time(c') & \text{if } c \equiv P \xrightarrow{a,\lambda} c' \end{cases}$$

We also pose:

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

for all $C \subseteq \mathcal{CC}(P, \alpha)$ and:

$$C_{\le t} = \{c \in C \mid time(c) \le t\}$$

for all $C \subseteq \mathcal{CC}(P, \alpha)$ and $t \in \mathbb{R}_{\ge 0}$. ■

Definition 4.4 Let $P_1, P_2 \in \mathcal{P}$. We say that P_1 is Markovian trace equivalent to P_2, written $P_1 \sim_{\rm MTr} P_2$, iff for all traces $\alpha \in AType^*$ and average amounts of time $t \in \mathbb{R}_{\ge 0}$:

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

4.2 Congruence Property

$\sim_{\rm MTr}$ turns out to be a congruence with respect to action prefix and alternative composition.

Theorem 4.5 *Let $P_1, P_2 \in \mathcal{P}$. Whenever $P_1 \sim_{\rm MTr} P_2$, then:*

(i) $<a, \lambda>.P_1 \sim_{\rm MTr} <a, \lambda>.P_2$ *for all* $<a, \lambda> \in Act$.

(ii) $P_1 + P \sim_{\rm MTr} P_2 + P$ *and* $P + P_1 \sim_{\rm MTr} P + P_2$ *for all* $P \in \mathcal{P}$. ■

4.3 Sound and Complete Axiomatization

It is easy to see that $\sim_{\rm MTr}$ is strictly contained in $\sim_{\rm MT}$, hence the axioms $\mathcal{A}_1$-$\mathcal{A}_4'$ of Table 1 are still valid for $\sim_{\rm MTr}$, but not complete. An example of process terms that are Markovian trace equivalent but not Markovian testing equivalent is given by:

$$<a, \lambda_1>.<b, \mu>.P_1 + <a, \lambda_2>.<c, \mu>.P_2$$

and:

$$<a, \lambda_1 + \lambda_2>.\left(<b, \frac{\lambda_1}{\lambda_1 + \lambda_2} \cdot \mu>.P_1 + <c, \frac{\lambda_2}{\lambda_1 + \lambda_2} \cdot \mu>.P_2\right)$$

In fact, no trace starting with a possibly followed by b or c can distinguish between the two terms, because in both terms the average time to perform an a-action followed by a b-action or a c-action is $1/(\lambda_1 + \lambda_2) + 1/\mu$ and the probability of reaching P_1 (resp. P_2) is $\lambda_1/(\lambda_1 + \lambda_2)$ (resp. $\lambda_2/(\lambda_1 + \lambda_2)$). By contrast, if $b \neq c$ then any test starting with a passive a-action followed by a passive b-action or a passive c-action can distinguish between the two terms as it increases the average sojourn time of the configuration involving $<b, \lambda_1/(\lambda_1+\lambda_2)\cdot\mu>.P_1 + <c, \lambda_2/(\lambda_1+\lambda_2)\cdot\mu>.P_2$ from $1/\mu$ to $(\lambda_1+\lambda_2)/\lambda_1 \cdot 1/\mu$ or $(\lambda_1 + \lambda_2)/\lambda_2 \cdot 1/\mu$, respectively.

$(\mathcal{A}_1)$	$P_1 + P_2 = P_2 + P_1$
$(\mathcal{A}_2)$	$(P_1 + P_2) + P_3 = P_1 + (P_2 + P_3)$
$(\mathcal{A}_3)$	$P + \underline{0} = P$
$(\mathcal{A}_4'')$	$\sum\limits_{i \in I} <a, \lambda_i>.\sum\limits_{j \in J_i} <b_{i,j}, \mu_{i,j}>.P_{i,j} = \; <a, \sum_{k \in I} \lambda_k>.\sum\limits_{i \in I}\sum\limits_{j \in J_i} <b_{i,j}, \frac{\lambda_i}{\sum_{k \in I} \lambda_k} \cdot \mu_{i,j}>.P_{i,j}$ if for all $i_1, i_2 \in I$: $\sum\limits_{j \in J_{i_1}} \mu_{i_1,j} = \sum\limits_{j \in J_{i_2}} \mu_{i_2,j} \equiv \mu$

Table 2
Axiomatization of $\sim_{\rm MTr}$ over $\mathcal{P}_{\rm nr}$

It turns out that the two terms above constitute the simplest instance of a more liberal axiom schema $\mathcal{A}_4''$ that we have to substitute for the more restrictive axiom schema $\mathcal{A}_4'$ in order to obtain a sound and complete axiomatization of $\sim_{\rm MTr}$ over $\mathcal{P}_{\rm nr}$. Such an axiomatization is given by the set $\mathcal{A}''$ of axioms shown in Table 2, where I and J_i are finite index sets with $|I| \geq 2$ (if $J_i = \emptyset$, the related summations are taken to be $\underline{0}$).

Theorem 4.6 *The deduction system DED($\mathcal{A}''$) is sound and complete for $\sim_{\rm MTr}$ over $\mathcal{P}_{\rm nr}$, i.e. for all $P_1, P_2 \in \mathcal{P}_{\rm nr}$:*

$$\mathcal{A}'' \vdash P_1 = P_2 \iff P_1 \sim_{\rm MTr} P_2$$

■

4.4 Exact Aggregation Property

By looking at the structure and at the rate constraints of the axiom schemata $\mathcal{A}_4'$ and $\mathcal{A}_4''$, it is straightforward to conclude that both axiom schemata result in the same CTMC-level aggregation, which is the one depicted in Sect. 3.5.

Theorem 4.7 $\sim_{\rm MTr}$ *induces the same CTMC-level aggregation as* $\sim_{\rm MT}$. ■

Corollary 4.8 *The CTMC-level aggregation induced by* $\sim_{\rm MTr}$ *is exact.* ■

5 Conclusion

In this paper we have shown that Markovian testing and trace equivalences induce at the CTMC level the same aggregation, which is strictly coarser than ordinary lumping and exact. This ensures that, whenever two process terms are Markovian testing or trace equivalent, they possess the same performance characteristics. Another consequence is that Markovian testing and trace equivalences improve on Markovian bisimilarity in terms of state space reduction, while preserving the exact aggregation property.

Viewed from a different angle, the main result of this paper is – to the best of our knowledge – the discovery of a new exact aggregation in the Markov chain theory, which is strictly coarser than ordinary lumping and entirely characterized in a process algebraic setting like ordinary lumping.

After establishing the fundamental property of exact aggregation, it becomes meaningful to investigate further properties of Markovian testing and trace equivalences. Apart from congruence and sound and complete axiomatization for dynamic operators, which we have already addressed in this paper, on the theoretical side we would like to investigate the congruence property with respect to parallel composition and we would like to derive logical characterizations of Markovian testing and trace equivalences.

On the verification side, a good starting point to devise an algorithm to check two process terms for Markovian testing equivalence may be the algorithm for classical testing equivalence proposed in [7]. This requires a more denotational characterization of Markovian testing equivalence, which may be inspired by the probabilistic variant of the acceptance tree model proposed in [15]. The issue of checking two process terms for Markovian trace equivalence has already been addressed in [17], where a polynomial-time algorithm has been devised.

As a final remark, in the light of the exact aggregation property proved in this paper for Markovian testing and trace equivalences, which in a sense extends ordinary lumping, it becomes interesting to understand whether the CTMC-level aggregation induced by Markovian testing and trace equivalences is the coarsest exact one that can be obtained or it can be further extended.

Acknowledgement

The author wishes to mention that a conjecture of the exactness of the CTMC-level aggregation induced by Markovian testing equivalence was made in 2000, at the end of a discussion with Mario Bravetti.

References

[1] 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.

[2] M. Bernardo, *"Non-Bisimulation-Based Markovian Behavioral Equivalences"*, submitted for publication.

[3] M. Bernardo and M. Bravetti, *"Performance Measure Sensitive Congruences for Markovian Process Algebras"*, in Theoretical Computer Science 290:117-160, 2003.

[4] M. Bernardo and W.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.

[5] P. Buchholz, *"Exact and Ordinary Lumpability in Finite Markov Chains"*, in Journal of Applied Probability 31:59-75, 1994.

[6] W.R. Cleaveland, Z. Dayar, S.A. Smolka, and S. Yuen, *"Testing Preorders for Probabilistic Processes"*, in Information and Computation 154:93-148, 1999.

[7] W.R. Cleaveland and M.C.B. Hennessy, *"Testing Equivalence as a Bisimulation Equivalence"*, in Formal Aspects of Computing 5:1-20, 1993.

[8] R. De Nicola and M.C.B. Hennessy, *"Testing Equivalences for Processes"*, in Theoretical Computer Science 34:83-133, 1983.

[9] 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.

[10] H. Hermanns, *"Interactive Markov Chains"*, LNCS 2428, 2002.

[11] 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.

[12] J. Hillston, *"A Compositional Approach to Performance Modelling"*, Cambridge University Press, 1996.

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

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

[15] M. Nuñez, *"Algebraic Theory of Probabilistic Processes"*, in Journal of Logic and Algebraic Programming 56:117-177, 2003.

[16] E.W. Stark, W.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.

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