Reward Based Congruences: Can We Aggregate More?

Marco Bernardo[1] and Mario Bravetti[2]

[1] Università di Torino, Dipartimento di Informatica
Corso Svizzera 185, 10149 Torino, Italy
bernardo@di.unito.it
[2] Università di Bologna, Dipartimento di Scienze dell'Informazione
Mura Anteo Zamboni 7, 40127 Bologna, Italy
bravetti@cs.unibo.it

Abstract. In this paper we extend a performance measure sensitive Markovian bisimulation congruence based on yield and bonus rewards that has been previously defined in the literature, in order to aggregate more states and transitions while preserving compositionality and the values of the performance measures. The extension is twofold. First, we show how to define a performance measure sensitive Markovian bisimulation congruence that aggregates bonus rewards besides yield rewards. This is achieved by taking into account in the aggregation process the conditional execution probabilities of the transitions to which the bonus rewards are attached. Second, we show how to define a performance measure sensitive Markovian bisimulation congruence that allows yield rewards and bonus rewards to be used interchangeably up to suitable correcting factors, aiming at the introduction of a normal form for rewards. We demonstrate that this is possible in the continuous time case, while it is not possible in the discrete time case because compositionality is lost.

1 Introduction

In the past five years the problem of specifying performance measures has been addressed in the field of Markovian concurrent systems. Following [10], in [6, 7, 3] the performance measures are characterized through atomic rewards to be suitably attached to the states and the transitions of the Markov chains (MCs for short) associated with the Markovian process algebraic specifications of the systems. While in [6, 7] the states to which certain rewards have to be attached are singled out by means of temporal logic formulas to be model checked, in [3] rewards are directly specified within the actions occurring in the specifications and are then trasferred to the MCs during their construction. In [13], instead, temporal reward formulas are introduced, which are able to express accumulated atomic rewards over sequences of states and allow performance measures to be evaluated on MCs through techniques for computing long run averages. Finally, in [1] a temporal logic is used to directly specify performance measures for Markovian transition systems, where the evaluation of such performance measures is conducted via model checking.

Among the approaches mentioned above, in this paper we concentrate on that of [3]. Its distinguishing feature is that of being deeply rooted in the Markovian process algebraic formalism. This has allowed us to define a performance measure sensitive congruence in the bisimulation style inspired by [11], which permits to compositionally manipulate system specifications without altering the values of their performance measures. We recall that taking performance measures into account when e.g. compositionally minimizing the state space of a Markovian process algebraic specification is important. If for example the equivalence used in the minimization process gives rise to the merging of two states whose associated rewards are different, we come to the undesirable situation in which the original model and the minimized model result in two different values for the same performance measure.

Following [10], the reward based Markovian bisimulation equivalence of [3] considers two types of rewards: yield rewards, which are accumulated while staying in the states, and bonus rewards, which are instantaneously gained when executing the transitions. Such an equivalence essentially aggregates yield rewards whenever possible, but does not manipulate bonus rewards at all.

The contribution of this paper is to extend the equivalence of [3] in order to aggregate as much as possible while retaining the bisimulation style, the congruence property, and the value of the performance measures. The extension is twofold. First, we show how to define a Markovian bisimulation congruence that aggregates bonus rewards as well. This is achieved by taking into account in the aggregation process the conditional execution probabilities of the transitions to which the bonus rewards are attached. Second, we show how to define a Markovian bisimulation congruence that allows yield rewards and bonus rewards to be used interchangeably up to suitable correcting factors, aiming at the introduction of a normal form for rewards. More precisely, we demonstrate that this is possible in the continuous time case, while it is not possible in the discrete time case because compositionality is lost.

Since the way in which bonus rewards can be aggregated is easy to find and the way in which yield and bonus rewards can be interchanged is known in the literature, this paper is especially concerned with investigating whether they respect compositionality or not. We also observe that, although such an investigation is conducted for a particular language ($\mathrm{EMPA}_{\mathrm{gr}_n}$ [3]), its results can be applied to every Markovian process algebra.

This paper is organized as follows. In Sect. 2 we recall the basic notions about reward structures. In Sect. 3 we recall $\mathrm{EMPA}_{\mathrm{gr}_n}$, a process algebra for the specification of continuous time and discrete time Markovian systems, and the related reward based Markovian bisimulation congruence. In Sect. 4 we present an improved reward based Markovian bisimulation equivalence that aggregates bonus rewards as well, we prove that it is a congruence, and we show a sound and complete axiomatization. In Sect. 5 we present a further improved reward based Markovian bisimulation equivalence that allows yield rewards and bonus rewards to be used interchangeably in the continuous time case, we prove that it is a congruence, and we show a sound and complete axiomatization. In Sect. 6 we report some concluding remarks.

2 Reward Structures

In the performance evaluation area, the technique of rewards is frequently used to specify and derive measures for system models whose underlying stochastic process is a MC. According to [10], a reward structure for a MC is composed of: a yield function $y_{i,j}(t)$, expressing the rate at which reward is accumulated at state i t time units after i was entered when the successor state is j, and a bonus function $b_{i,j}(t)$, expressing the reward awarded upon exit from state i and subsequent entry into state j given that the holding time in state i was t time units. Since the generality of this structure is difficult to fully exploit due to the complexity of the resulting solution, the analysis is usually simplified by considering yield functions that do not depend on the time nor the successor state, as well as bonus functions that do not depend on the holding time of the previously occupied state: $y_{i,j}(t) = y_i$ and $b_{i,j}(t) = b_{i,j}$.

Several performance measures can be calculated by exploiting rewards. According to the classifications proposed in [12, 9], we have instant-of-time measures, expressing the gain received at a particular time instant, and interval-of-time (or cumulative) measures, expressing the overall gain received over some time interval. Both kinds of measures can refer to stationary or transient state. In the following, we concentrate on instant-of-time performance measures.

In the stationary case, instant-of-time performance measures quantify the long run gain received per unit of time. Given yield rewards y_i and bonus rewards $b_{i,j}$ for a certain MC, the corresponding stationary performance measure is computed as:

$$\sum_i y_i \cdot \pi_i + \sum_i \sum_j b_{i,j} \cdot \phi_{i,j} \tag{1}$$

where π_i is the stationary probability of state i and $\phi_{i,j}$ is the stationary frequency with which the transition from state i to state j is traversed. $\phi_{i,j}$ is given by the stationary frequency with which state i is entered (i.e. the ratio of its stationary probability to its average holding time) multiplied by the probability with which the transition from state i to state j is traversed given that the current state is i. In the case of a continuous time MC (CTMC) we have $\phi_{i,j} = \pi_i \cdot q_{i,j}$ with $q_{i,j}$ being the rate of the transition from i to j, while in the case of a discrete time MC (DTMC) we have $\phi_{i,j} = \pi_i \cdot p_{i,j}$ with $p_{i,j}$ being the probability of the transition from i to j.

In the transient case, instant-of-time performance measures quantify the gain received at a specific time instant. Given yield rewards y_i and bonus rewards $b_{i,j}$ for a certain MC, the corresponding transient state performance measure is computed as:

$$\sum_i y_i \cdot \pi_i(t) + \sum_i \sum_j b_{i,j} \cdot \phi_{i,j}(t) \tag{2}$$

where $\pi_i(t)$ is the transient probability of being in state i at time t and $\phi_{i,j}(t)$ is the transient frequency with which the transition from state i to state j is traversed at time t, which is computed in the same way as $\phi_{i,j}$ with $\pi_i(t)$ in place of π_i.

3 An Overview of $\mathrm{EMPA}_{\mathrm{gr}_n}$

$\mathrm{EMPA}_{\mathrm{gr}_n}$ [3] is a family of extended Markovian process algebras with generative-reactive synchronizations [5] whose actions are enriched in order to accommodate the specification of $n \in \mathbb{N}$ different performance measures simultaneously. This is achieved by associating with every action a pair composed of a yield reward and a bonus reward for each performance measure. The number n is called the order. For the sake of simplicity, without loss of generality in this paper we deal with order 1 only.

3.1 Syntax and Informal Semantics

The main ingredients of our calculus are the actions, each composed of a type, a rate, and a sequence of n pairs of yield and bonus rewards, and the algebraic operators. As far as actions are concerned, based on their rates they are classified into exponentially timed (rate $\lambda \in \mathbb{R}_+$ representing the parameter of the corresponding exponentially distributed duration), immediate (rate $\infty_{l,w}$ to denote a zero duration, with $l \in \mathbb{N}_+$ being a priority level and $w \in \mathbb{R}_+$ being a weight associated with the action choice), and passive (rate $*_{l,w}$ to denote an unspecified duration with priority level l and weight w associated with the action choice). Moreover, based on their types, actions are classified into observable and invisible depending on whether they are different or equal to τ, as usual.

Definition 1. *Let AType be the set of action types including the invisible type* τ, $ARate = \mathbb{R}_+ \cup \{\infty_{l,w} \mid l \in \mathbb{N}_+ \wedge w \in \mathbb{R}_+\} \cup \{*_{l,w} \mid l \in \mathbb{N}_+ \wedge w \in \mathbb{R}_+\}$ *be the set of action rates,* $ARew = \mathbb{R} \cup \{*\}$ *be the set of action rewards. We use* a *to range over AType,* $\tilde{\lambda}$ *to range over ARate,* λ *to range over exponentially timed rates,* $\bar{\lambda}$ *to range over active (i.e. nonpassive) rates,* $\tilde{y}$ *to range over yield rewards (*y *if not* $*$*), and* $\tilde{b}$ *to range over bonus rewards (*b *if not* $*$*). The set of actions of order 1 is defined by*

$$\begin{aligned} Act_1 = \{<a, \tilde{\lambda}, (\tilde{y}, \tilde{b})> \in AType \times ARate \times (ARew \times ARew) \mid \\ (\tilde{\lambda} \in \{*_{l,w} \mid l \in \mathbb{N}_+ \wedge w \in \mathbb{R}_+\} \wedge \tilde{y} = \tilde{b} = *) \vee \\ (\tilde{\lambda} \in \mathbb{R}_+ \cup \{\infty_{l,w} \mid l \in \mathbb{N}_+ \wedge w \in \mathbb{R}_+\} \wedge \tilde{y}, \tilde{b} \in \mathbb{R})\} \end{aligned} \quad \blacksquare$$

Definition 2. *Let Const be a set of constants ranged over by* A *and let ATRFun* $= \{\varphi : AType \longrightarrow AType \mid \varphi^{-1}(\tau) = \{\tau\}\}$ *be a set of action type relabeling functions ranged over by* φ*. The set* $\mathcal{L}_1$ *of process terms of* $\mathrm{EMPA}_{\mathrm{gr}_1}$ *is generated by the following syntax*

$$E ::= \underline{0} \mid <a, \tilde{\lambda}, (\tilde{y}_1, \tilde{b}_1)>.E \mid E/L \mid E[\varphi] \mid E + E \mid E \parallel_S E \mid A$$

where $L, S \subseteq AType - \{\tau\}$. $\blacksquare$

The null term "$\underline{0}$" is the term that cannot execute any action.

The action prefix operator "$<a, \tilde{\lambda}, (\tilde{y}, \tilde{b})>.\_$" denotes the sequential composition of an action and a term. Term $<a, \tilde{\lambda}, (\tilde{y}, \tilde{b})>.E$ can execute an action with type a and rate $\tilde{\lambda}$, thus making the corresponding state earn additional yield reward $\tilde{y}$ and the related transition gain bonus reward $\tilde{b}$, and then behaves as term E.

The functional abstraction operator "$\_/L$" abstracts from the type of the actions. Term E/L behaves as term E except that the type a of each executed action is turned into τ whenever $a \in L$.

The functional relabeling operator "$\_[\varphi]$" changes the type of the actions. Term $E[\varphi]$ behaves as term E except that the type a of each executed action becomes $\varphi(a)$.

The alternative composition operator "$\_ + \_$" expresses a choice between two terms. Term $E_1 + E_2$ behaves as either term E_1 or term E_2 depending on whether an action of E_1 or an action of E_2 is executed. The choice among several enabled exponentially timed actions is solved according to the race policy, i.e. the fastest action is executed (this implies that each action has an execution probability proportional to its rate). If immediate actions are enabled as well, they take precedence over exponentially timed ones and the choice among them is solved according to the preselection policy: the immediate actions having the highest priority level are singled out, then each of them is given an execution probability proportional to its weight. The choice among several enabled passive actions is instead solved according to the reactive preselection policy: for each action type, the passive actions having the highest priority level are singled out, then each of them is given an execution probability proportional to its weight. Therefore, the choice among passive actions having different types and the choice between passive and active actions are nondeterministic.

The parallel composition operator "$\_ \parallel_S \_$" expresses the concurrent execution of two terms. Term $E_1 \parallel_S E_2$ asynchronously executes actions of E_1 or E_2 not belonging to S and synchronously executes actions of E_1 and E_2 belonging to S according to the two following synchronization disciplines. The synchronization discipline on action types establishes that two actions can synchronize if and only if they have the same observable type in S, which becomes the resulting type. Following the terminology of [8], the synchronization discipline on action rates is the generative-reactive mechanism, which establishes that two actions can synchronize if and only if at least one of them is passive (behaves reactively). In case of synchronization of an active action a having rate $\tilde{\lambda}$ executed by E_1 (E_2) with a passive action a having rate $*_{l,w}$ executed by E_2 (E_1), the resulting active action a has rate/weight given by the original rate/weight multiplied by the probability that E_2 (E_1) chooses the passive action at hand among its passive actions of type a. Instead, in case of synchronization of two passive actions a having rate $*_{l_1,w_1}$ and $*_{l_2,w_2}$ executed by E_1 and E_2, respectively, the resulting passive action of type a has priority level given by the maximum l_{max} between l_1 and l_2 and weight given by the probability that E_1 and E_2 independently choose the two actions, multiplied by a normalization factor given by the overall weight of the passive actions of type a executable by E_1 and E_2 at the priority level l_{max}. As far as rewards are concerned, since only the rewards of active actions are specified, in case of synchronization they are handled as follows. The yield rewards of an active action are treated exactly as the rate of that action, i.e. they are multiplied by the execution probabilities of the passive actions involved in the synchronization. Instead, the bonus rewards of an active action are just inherited, as multiplying them by the execution probabilities of the aforementioned passive actions would lead to an underestimation of the performance measures. The reason is that, in the calculation of the performance measures according to formulas (1) and (2), each bonus reward of a transition is multiplied by a factor that is proportional to the rate of the transition itself, hence multiplying the rates by the execution probabilities of passive actions is all we have to do. In the case of synchronization between two passive actions, the rewards of the resulting passive actions are still unspecified.

Finally, we assume the existence of a set of constant defining equations of the form $A \stackrel{\Delta}{=} E$. In order to guarantee the correctness of recursive definitions, as usual we restrict ourselves to the set $\mathcal{G}_1$ of terms that are closed and guarded w.r.t. Def_1.

3.2 Reward Master-Slaves Transition Systems

The semantic model of $\mathrm{EMPA}_{\mathrm{gr}_1}$ is a special kind of LTS called master-slaves transition system of order 1 (RMSTS_1 for short), whose transitions are labeled with elements of Act_1. Recalling that active actions play the role of the masters (they behave generatively) while passive actions play the role of the slaves (they behave reactively), each state of a RMSTS_1 has a single master bundle composed of all the transitions labeled with an active action and, for each action type a, a single slave bundle of type a composed of all the transitions labeled with a passive action of type a. Since the operational semantics for $\mathrm{EMPA}_{\mathrm{gr}_1}$ will be defined in such a way that lower priority active transitions are not pruned (in order to get a congruence) while lower priority passive transitions of a given type are, all the passive transitions belonging to the same slave bundle of a generated RMSTS_1 have the same priority level.

Definition 3. *A reward master-slaves transition system of order 1 ($RMSTS_1$) is a triple*

$$(S, AType, \longrightarrow)$$

where S is a set of states, $AType$ is a set of action types, and $\longrightarrow\, \in \mathcal{M}(S \times Act_1 \times S)$ is a multiset [1] of transitions such that for all $s \in S$ and $a \in AType$:

$$(s \xrightarrow{a, *_{l', w'}, (*,*)} s' \wedge s \xrightarrow{a, *_{l'', w''}, (*,*)} s'') \implies l' = l''$$

A rooted reward master-slaves transition system of order 1 ($RRMSTS_1$) is a quadruple

$$(S, AType, \longrightarrow, s_0)$$

where $(S, AType, \longrightarrow)$ is a $RMSTS_1$ and $s_0 \in S$ is the initial state. ■

[1] We use "$\{\!|$" and "$|\!\}$" as brackets for multisets and $\mathcal{M}(S)$ ($\mathcal{P}(S)$) to denote the collection of multisets over (subsets of) S.

We point out that the transition relation is a multiset, not a set. This allows the multiplicity of identically labeled transitions to be taken into account, which is necessary from the stochastic point of view. As an example, if a state has two transitions both labeled with $<a, \lambda, (y, b)>$, using sets instead of multisets would reduce the two transitions into a single one with rate λ, thus erroneously altering the average sojourn time in the state.

Given a state, the choice among the bundles of transitions enabled in that state is nondeterministic. The choice of a transition within the master bundle is governed by the race policy if there are only exponentially timed transitions, the preselection policy if there are immediate transitions (which take precedence over exponentially timed transitions). The choice of a transition within a slave bundle of type a is governed by the preselection policy.

We observe that the passive actions are seen as incomplete actions that must synchronize with active actions of the same type of another system component in order to form a complete system. Therefore, a fully specified system is performance closed, in the sense that it gives rise to a fully probabilistic transition system that does not include slave bundles. If in such a transition system we keep for each state only the highest priority transitions, then we can easily derive a performance model in the form of a reward DTMC or CTMC, depending on whether only immediate transitions occur or not. We point out that, if only immediate transitions occur, each of them is assumed to take one time step, hence the underlying stochastic model naturally turns out to be a DTMC. Should exponentially timed and immediate transitions coexist (in different states), a CTMC is derived by eliminating the immediate transitions and the related source states and by suitably splitting the exponentially timed transitions entering the removed source states, in such a way that they are caused to reach the target states of the removed immediate transitions.

As far as the yield rewards are concerned, when constructing a reward MC from a RRMSTS_1 we proceed as follows. Whenever a state has several actions, be it due to an alternative composition operator or a parallel composition operator, we make the additivity assumption, i.e. we assume that the yield reward earned by the state is the sum of the yield rewards of its transitions. This assumption is consistent with the race inherent in the parallel composition operator and with the adoption of the race policy for the alternative composition operator, i.e. with viewing alternative actions as being in parallel execution, hence all contributing to the reward accumulation in the state.

3.3 Operational Semantics

The formal semantics for $\mathrm{EMPA}_{\mathrm{gr}_1}$ maps terms onto RRMSTS_1. We preliminarily provide the following shorthands to make the definition of the operational semantic rules easier.

Definition 4. *Given a $RMSTS_1$ $M = (S, AType, \longrightarrow)$, $s \in S$, and $a \in AType$, we denote by $L_a(s)$ the priority level of the slave transitions of type a executable at s ($L_a(s) = 0$ if the slave bundle a of s is empty) and we denote* *by* $W_a(s)$ *the overall weight of the slave transitions of type a executable at s:*

$$W_a(s) = \sum \{\!| \, w \mid \exists s' \in S.\, s \xrightarrow{a,*_{L_a(s),w},(*,*)} s' \, |\!\}$$

Furthermore, we extend the real number multiplication to immediate rates as follows:

$$\infty_{l,w} \cdot p = \infty_{l,w \cdot p}$$ ■

The operational semantics for $\mathrm{EMPA}_{\mathrm{gr}_1}$ is the least RMSTS_1 $(\mathcal{G}_1, \mathit{AType}, \longrightarrow_1)$ satisfying the inference rules of Table 1, where in addition to the rules $(Ch1_l)$, $(Ch2_l)$, $(Pa1_l)$, $(Pa2_l)$, $(Sy1_l)$ referring to a move of the lefthand process E_1, we consider also the symmetrical rules $(Ch1_r)$, $(Ch2_r)$, $(Pa1_r)$, $(Pa2_r)$, $(Sy1_r)$ taking into account the moves of the righthand process E_2, obtained by exchanging the roles of terms E_1 and E_2. We consider the operational rules as generating a multiset of transitions (consistently with the definition of RMSTS_1), where a transition has arity m if and only if it can be derived in m possible ways from the operational rules.

Some explanations are now in order. First of all, the operational rules give rise to an interleaving semantics, which is made possible by the memoryless property of exponential distributions. The removal of lower priority passive transitions of the same type is carried out in rules $(Ch2_l)$ and $(Ch2_r)$ for the alternative composition operator and rules $(Pa1_l)$ and $(Pa1_r)$ for the parallel composition operator by using $L_a(E)$.

In the case of a synchronization, the evaluation of the rate of the resulting action is carried out by rules $(Sy1_l)$, $(Sy1_r)$, and $(Sy2)$ as follows. Whenever an active action synchronizes with a passive action of the same type, the rate of the resulting active action is evaluated in rules $(Sy1_l)$ and $(Sy1_r)$ by multiplying the rate of the active action by the probability of choosing the passive action. The yield reward of the active action undergoes the same treatment, while the bonus reward is just inherited. Whenever two passive actions of type a synchronize, instead, the priority level and the weight of the resulting passive action are computed as described by rule $(Sy2)$. In particular, the weight is computed by multiplying the probability p of independently choosing the two original actions by the normalization factor N, which is given by the overall weight of the passive transitions of type a with maximum priority level executable by E_1 and E_2, computed by using $W_a(E)$.

Definition 5. *The integrated semantics of* $E \in \mathcal{G}_1$ *is the* RRMSTS_1

$$\mathcal{I}_1[\![E]\!] = (\mathcal{G}_{1,E}, \mathit{AType}, \longrightarrow_{1,E}, E)$$

where $\mathcal{G}_{1,E}$ *is the set of terms reachable from* E *according to the* RMSTS_1 $(\mathcal{G}_1, \mathit{AType}, \longrightarrow_1)$ *and* $\longrightarrow_{1,E}$ *is the restriction of* $\longrightarrow_1$ *to transitions between terms in* $\mathcal{G}_{1,E}$. *We say that* $E \in \mathcal{G}_1$ *is performance closed if and only if* $\mathcal{I}_1[\![E]\!]$ *does not contain passive transitions. We denote by* $\mathcal{E}_1$ *the set of performance closed terms of* $\mathcal{G}_1$. ■

We conclude by recalling that from $\mathcal{I}_1[\![E]\!]$ two projected semantic models can be obtained by essentially dropping action rates or action types, respectively. Before applying such a transformation to $\mathcal{I}_1[\![E]\!]$, lower priority active transitions

$$(\boldsymbol{Pr})\quad <a,\tilde{\lambda},(\tilde{y},\tilde{b})>.E \xrightarrow{a,\tilde{\lambda},(\tilde{y},\tilde{b})}_1 E$$

$$(\boldsymbol{Hi1})\quad \frac{E \xrightarrow{a,\tilde{\lambda},(\tilde{y},\tilde{b})}_1 E'}{E/L \xrightarrow{a,\tilde{\lambda},(\tilde{y},\tilde{b})}_1 E'/L}\ a \notin L \qquad (\boldsymbol{Hi2})\quad \frac{E \xrightarrow{a,\tilde{\lambda},(\tilde{y},\tilde{b})}_1 E'}{E/L \xrightarrow{\tau,\tilde{\lambda},(\tilde{y},\tilde{b})}_1 E'/L}\ a \in L$$

$$(\boldsymbol{Re})\quad \frac{E \xrightarrow{a,\tilde{\lambda},(\tilde{y},\tilde{b})}_1 E'}{E[\varphi] \xrightarrow{\varphi(a),\tilde{\lambda},(\tilde{y},\tilde{b})}_1 E'[\varphi]}$$

$$(\boldsymbol{Ch1_l})\quad \frac{E_1 \xrightarrow{a,\bar{\lambda},(y,b)}_1 E_1'}{E_1 + E_2 \xrightarrow{a,\bar{\lambda},(y,b)}_1 E_1'} \qquad (\boldsymbol{Ch2_l})\quad \frac{E_1 \xrightarrow{a,*_{l,w},(*,*)}_1 E_1' \qquad l \geq L_a(E_2)}{E_1 + E_2 \xrightarrow{a,*_{l,w},(*,*)}_1 E_1'}$$

$$(\boldsymbol{Pa1_l})\quad \frac{E_1 \xrightarrow{a,\bar{\lambda},(y,b)}_1 E_1'}{E_1 \parallel_S E_2 \xrightarrow{a,\bar{\lambda},(y,b)}_1 E_1' \parallel_S E_2}\ a \notin S$$

$$(\boldsymbol{Pa2_l})\quad \frac{E_1 \xrightarrow{a,*_{l,w},(*,*)}_1 E_1' \qquad l \geq L_a(E_2)}{E_1 \parallel_S E_2 \xrightarrow{a,*_{l,w},(*,*)}_1 E_1' \parallel_S E_2}\ a \notin S$$

$$(\boldsymbol{Sy1_l})\quad \frac{E_1 \xrightarrow{a,\bar{\lambda},(y,b)}_1 E_1' \qquad E_2 \xrightarrow{a,*_{l,w},(*,*)}_1 E_2'}{E_1 \parallel_S E_2 \xrightarrow{a,\bar{\lambda}\cdot\frac{w}{W_a(E_2)},(y\cdot\frac{w}{W_a(E_2)},b)}_1 E_1' \parallel_S E_2'}\ a \in S$$

$$(\boldsymbol{Sy2})\quad \frac{E_1 \xrightarrow{a,*_{l_1,w_1},(*,*)}_1 E_1' \qquad E_2 \xrightarrow{a,*_{l_2,w_2},(*,*)}_1 E_2'}{E_1 \parallel_S E_2 \xrightarrow{a,*_{\max(l_1,l_2),p\cdot N},(*,*)}_1 E_1' \parallel_S E_2'}\ a \in S$$

where: $p = \frac{w_1}{W_a(E_1)} \cdot \frac{w_2}{W_a(E_2)}$ $\qquad N = \begin{cases} W_a(E_1) + W_a(E_2) & \text{if } l_1 = l_2 \\ W_a(E_1) & \text{if } l_1 > l_2 \\ W_a(E_2) & \text{if } l_2 > l_1 \end{cases}$

$$(\boldsymbol{Co})\quad \frac{E \xrightarrow{a,\tilde{\lambda},(\tilde{y},\tilde{b})}_1 E'}{A \xrightarrow{a,\tilde{\lambda},(\tilde{y},\tilde{b})}_1 E'}\ A \stackrel{\Delta}{=} E$$

Table 1. $\mathrm{EMPA}_{\mathrm{gr}_1}$ operational semantics

are pruned because E is no longer to be composed with other terms as it describes the whole system we are interested in. The functional semantics $\mathcal{F}_1[\![E]\!]$ is a standard LTS whose transitions are decorated with action types only. The Markovian semantics $\mathcal{M}_1[\![E]\!]$ is instead a reward CTMC or DTMC, as seen in Sect. 3.2, which is well defined only if E is performance closed.

3.4 Reward Based Markovian Bisimulation Equivalence

$\mathrm{EMPA}_{\mathrm{gr}_1}$ is equipped with a reward based Markovian bisimulation equivalence, which relates two systems having the same functional, probabilistic, prioritized and exponentially timed behavior, as well as the same performance measure values, by considering their ability to simulate each other behavior. To achieve this, the rates/weights of the transitions of the same type and priority level that leave the same state and reach states belonging to the same equivalence class are summed up, like in the exact aggregation for MCs known as ordinary lumping. The yield rewards labeling the transitions above are handled in the same way as the corresponding rates, because of the additivity assumption. The bonus rewards of the transitions above, instead, are not summed up, as this would result in an overestimation of the specified performance measures. The reason is that, in the calculation of the performance measures according to formulas (1) and (2), the bonus reward of a transition is multiplied by a factor that is proportional to the rate of the transition itself, hence summing rates up is all we have to do.

Definition 6. *We define function priority level $PL : ARate \longrightarrow \mathbb{Z}$ by:*

$$\begin{aligned} PL(*_{l,w}) &= -l \\ PL(\lambda) &= 0 \\ PL(\infty_{l,w}) &= l \end{aligned}$$

and we extend the real number summation to rates of the same priority level and to unspecified rewards as follows:

$$\begin{aligned} *_{l,w_1} + *_{l,w_2} &= *_{l,w_1+w_2} \\ \infty_{l,w_1} + \infty_{l,w_2} &= \infty_{l,w_1+w_2} \\ * + * &= * \end{aligned}$$

We define partial function aggregated rate-yield $RY_1 : \mathcal{G}_1 \times AType \times \mathbb{Z} \times ARew \times \mathcal{P}(\mathcal{G}_1) \multimap ARate \times ARew$ by:

$$RY_1(E, a, l, \tilde{b}, C) = (Rate_1(E, a, l, \tilde{b}, C),\ Yield_1(E, a, l, \tilde{b}, C))$$

where:

$$\begin{aligned} Rate_1(E, a, l, \tilde{b}, C) &= \sum\{\!|\ \tilde{\lambda} \mid \exists \tilde{y}.\, \exists E' \in C.\, E \xrightarrow{a,\tilde{\lambda},(\tilde{y},\tilde{b})}_1 E' \wedge PL(\tilde{\lambda}) = l\ |\!\} \\ Yield_1(E, a, l, \tilde{b}, C) &= \sum\{\!|\ \tilde{y} \mid \exists \tilde{\lambda}.\, \exists E' \in C.\, E \xrightarrow{a,\tilde{\lambda},(\tilde{y},\tilde{b})}_1 E' \wedge PL(\tilde{\lambda}) = l\ |\!\} \end{aligned}$$

with $RY_1(E, a, l, \tilde{b}, C) = \bot$ whenever the multisets above are empty. ■

Definition 7. *An equivalence relation $\mathcal{B} \subseteq \mathcal{G}_1 \times \mathcal{G}_1$ is a RY-Markovian bisimulation of order 1 if and only if, whenever $(E_1, E_2) \in \mathcal{B}$, then for all $a \in AType$, $l \in \mathbb{Z}$, $\tilde{b} \in ARew$, and equivalence classes $C \in \mathcal{G}_1/\mathcal{B}$*

$$RY_1(E_1, a, l, \tilde{b}, C) = RY_1(E_2, a, l, \tilde{b}, C)$$ ■

$$
\begin{array}{lrcl}
(\mathcal{A}_1)_1^{\mathrm{RY}} & (E_1 + E_2) + E_3 & = & E_1 + (E_2 + E_3) \\
(\mathcal{A}_2)_1^{\mathrm{RY}} & E_1 + E_2 & = & E_2 + E_1 \\
(\mathcal{A}_3)_1^{\mathrm{RY}} & E + \underline{0} & = & E \\
(\mathcal{A}_4)_1^{\mathrm{RY}} & <a, \tilde{\lambda}_1, (\tilde{y}_1, \tilde{b})>.E + <a, \tilde{\lambda}_2, (\tilde{y}_2, \tilde{b})>.E & = & <a, \tilde{\lambda}_1 + \tilde{\lambda}_2, (\tilde{y}_1 + \tilde{y}_2, \tilde{b})>.E \\
 & & & \text{if } PL(\tilde{\lambda}_1) = PL(\tilde{\lambda}_2) \\
(\mathcal{A}_5)_1^{\mathrm{RY}} & <a, *_{l_1, w_1}, (*, *)>.E_1 + <a, *_{l_2, w_2}, (*, *)>.E_2 & = & <a, *_{l_1, w_1}, (*, *)>.E_1 \\
 & & & \text{if } l_1 > l_2 \\
 & & & \\
(\mathcal{A}_6)_1^{\mathrm{RY}} & \underline{0}/L & = & \underline{0} \\
(\mathcal{A}_7)_1^{\mathrm{RY}} & (<a, \tilde{\lambda}, (\tilde{y}, \tilde{b})>.E)/L & = & <a, \tilde{\lambda}, (\tilde{y}, \tilde{b})>.(E/L) \\
 & & & \text{if } a \notin L \\
(\mathcal{A}_8)_1^{\mathrm{RY}} & (<a, \tilde{\lambda}, (\tilde{y}, \tilde{b})>.E)/L & = & <\tau, \tilde{\lambda}, (\tilde{y}, \tilde{b})>.(E/L) \\
 & & & \text{if } a \in L \\
(\mathcal{A}_9)_1^{\mathrm{RY}} & (E_1 + E_2)/L & = & E_1/L + E_2/L \\
 & & & \\
(\mathcal{A}_{10})_1^{\mathrm{RY}} & \underline{0}[\varphi] & = & \underline{0} \\
(\mathcal{A}_{11})_1^{\mathrm{RY}} & (<a, \tilde{\lambda}, (\tilde{y}, \tilde{b})>.E)[\varphi] & = & <\varphi(a), \tilde{\lambda}, (\tilde{y}, \tilde{b})>.(E[\varphi]) \\
(\mathcal{A}_{12})_1^{\mathrm{RY}} & (E_1 + E_2)[\varphi] & = & E_1[\varphi] + E_2[\varphi]
\end{array}
$$

$$
\begin{aligned}
(\mathcal{A}_{13})_1^{\mathrm{RY}}\ & \sum_{i \in I_0} <a_i, \tilde{\lambda}_i, (\tilde{y}_i, \tilde{b}_i)>.E_i \parallel_S \sum_{i \in I_1} <a_i, \tilde{\lambda}_i, (\tilde{y}_i, \tilde{b}_i)>.E_i \ = \\
& \sum_{j \in I_0, a_j \notin S} <a_j, \tilde{\lambda}_j, (\tilde{y}_j, \tilde{b}_j)>.(E_j \parallel_S \sum_{i \in I_1} <a_i, \tilde{\lambda}_i, (\tilde{y}_i, \tilde{b}_i)>.E_i) + \\
& \sum_{j \in I_1, a_j \notin S} <a_j, \tilde{\lambda}_j, (\tilde{y}_j, \tilde{b}_j)>.(\sum_{i \in I_0} <a_i, \tilde{\lambda}_i, (\tilde{y}_i, \tilde{b}_i)>.E_i \parallel_S E_j) + \\
& \sum_{k \in K_0} \sum_{h \in P_{1, a_k}} <a_k, \tilde{\lambda}_k \cdot (w_h / W_{1, a_k}), (\tilde{y}_k \cdot (w_h / W_{1, a_k}), \tilde{b}_k)>.(E_k \parallel_S E_h) + \\
& \sum_{k \in K_1} \sum_{h \in P_{0, a_k}} <a_k, \tilde{\lambda}_k \cdot (w_h / W_{0, a_k}), (\tilde{y}_k \cdot (w_h / W_{0, a_k}), \tilde{b}_k)>.(E_h \parallel_S E_k) + \\
& \sum_{k \in P'_0} \sum_{h \in P_{1, a_k}} <a_k, *_{\max(l_k, l_h), (w_k / W_{0, a_k}) \cdot (w_h / W_{0, a_k}) \cdot N_{a_k}}, (*, *)>.(E_k \parallel_S E_h)
\end{aligned}
$$

where $I_0 \cap I_1 = \emptyset,\ \tilde{\lambda}_i = *_{l_i, w_i}$ for $i \in I_0 \cup I_1.\ PL(\tilde{\lambda}_i) < 0,$ and for $j \in \{0, 1\}$

$$
\begin{aligned}
L_{j,a} &= \max\{l_k \mid k \in I_j \wedge a_k = a \wedge \tilde{\lambda}_k = *_{l_k, w_k}\} \\
P_{j,a} &= \{k \in I_j \mid a_k = a \wedge \tilde{\lambda}_k = *_{l_k, w_k} \wedge l_k = L_{j,a}\} \\
K_j &= \{k \in I_j \mid a_k \in S \wedge PL(\tilde{\lambda}_k) \geq 0 \wedge P_{1-j, a_k} \neq \emptyset\} \\
P'_0 &= \{k \in I_0 \mid \exists a \in S.\, k \in P_{0,a} \wedge P_{1,a} \neq \emptyset\} \\
W_{j,a} &= \sum \{\!| \, w_k \mid k \in P_{j,a} \wedge \tilde{\lambda}_k = *_{l_k, w_k} \, |\!\} \\
N_a &= \begin{cases} W_{0,a} + W_{1,a} & \text{if } L_{0,a} = L_{1,a} \\ W_{0,a} & \text{if } L_{0,a} > L_{1,a} \\ W_{1,a} & \text{if } L_{1,a} > L_{0,a} \end{cases}
\end{aligned}
$$

Table 2. Axiomatization of $\sim_{\mathrm{MB}_1^{\mathrm{RY}}}$

It is easy to see that the union of all the RY-Markovian bisimulations of order 1 is a RY-Markovian bisimulation of order 1. Such a union, denoted $\sim_{\mathrm{MB}_1^{\mathrm{RY}}}$, is called the RY-Markovian bisimulation equivalence of order 1. $\sim_{\mathrm{MB}_1^{\mathrm{RY}}}$ is a congruence w.r.t. all the algebraic operators as well as recursive constant definitions. In Table 2 we report from [3] the sound and complete axiomatization of nonrecursive $\mathrm{EMPA}_{\mathrm{gr}_1}$ terms w.r.t. $\sim_{\mathrm{MB}_1^{\mathrm{RY}}}$.

4 Aggregating Bonus Rewards

As witnessed by axiom $(\mathcal{A}_4)_1^{\mathrm{RY}}$, $\sim_{\mathrm{MB}_1^{\mathrm{RY}}}$ aggregates rates and yield rewards without manipulating bonus rewards at all. However, if we look at the way performance measures are computed according to formulas (1) and (2), we note that whenever two actions are merged into a single one, then their bonus rewards can be aggregated as well. Unlike the rates and the yield rewards of those two actions, the bonus rewards are not just summed up as each of them needs to be preliminarily multiplied by the execution probability of the corresponding action. As an example, $<a, \lambda_1, (y_1, b_1)>.E + <a, \lambda_2, (y_2, b_2)>.E$ can be equated to $<a, \lambda_1 + \lambda_2, (y_1 + y_2, \frac{\lambda_1}{\lambda_1+\lambda_2} \cdot b_1 + \frac{\lambda_2}{\lambda_1+\lambda_2} \cdot b_2)>.E$; similarly $<a, \infty_{l,w_1}, (y_1, b_1)>.E + <a, \infty_{l,w_2}, (y_2, b_2)>.E$ can be equated to $<a, \infty_{l,w_1+w_2}, (y_1 + y_2, \frac{w_1}{w_1+w_2} \cdot b_1 + \frac{w_2}{w_1+w_2} \cdot b_2)>.E$. To be more precise, the probability by which each bonus reward involved in the aggregation must be multiplied is the probability of executing the corresponding action conditioned on the fact that one of the actions involved in the aggregation is executed. Considering such a conditional execution probability instead of just the execution probability is not only necessary to preserve the value of the performance measures according to formulas (1) and (2), but is also crucial to get the congruence property. We introduce below an improved reward based Markovian bisimulation congruence that aggregates bonus rewards as well.

Definition 8. *We extend the real number division to rates of the same priority level as follows:*

$$\begin{aligned} *_{l,w_1} \;/\; *_{l,w_2} &= *_{l,w_1/w_2} \\ \infty_{l,w_1} \;/\; \infty_{l,w_2} &= w_1/w_2 \end{aligned}$$

and we extend the real number multiplication to passive rates and unspecified rewards as follows:

$$*_{l,w} \cdot * = *$$

We define partial function aggregated rate-yield-bonus $RYB_1 : \mathcal{G}_1 \times AType \times \mathbb{Z} \times \mathcal{P}(\mathcal{G}_1) \nrightarrow ARate \times ARew \times ARew$ *by:*

$$RYB_1(E, a, l, C) = (Rate_1(E, a, l, C), Yield_1(E, a, l, C), Bonus_1(E, a, l, C))$$

where:

$$Rate_1(E, a, l, C) = \sum\{\!|\, \tilde{\lambda} \mid \exists \tilde{y}, \tilde{b}.\, \exists E' \in C.\, E \xrightarrow{a, \tilde{\lambda}, (\tilde{y}, \tilde{b})}_1 E' \wedge PL(\tilde{\lambda}) = l \,|\!\}$$

$$Yield_1(E, a, l, C) = \sum\{\!|\, \tilde{y} \mid \exists \tilde{\lambda}, \tilde{b}.\, \exists E' \in C.\, E \xrightarrow{a, \tilde{\lambda}, (\tilde{y}, \tilde{b})}_1 E' \wedge PL(\tilde{\lambda}) = l \,|\!\}$$

$$Bonus_1(E, a, l, C) = \sum\{\!|\, \frac{\tilde{\lambda}}{Rate_1(E,a,l,C)} \cdot \tilde{b} \mid \exists \tilde{y}.\, \exists E' \in C.\, E \xrightarrow{a, \tilde{\lambda}, (\tilde{y}, \tilde{b})}_1 E' \wedge PL(\tilde{\lambda}) = l \,|\!\}$$

with $RYB_1(E,a,l,C) = \bot$ *whenever the multisets above are empty.* ■

Definition 9. *An equivalence relation* $\mathcal{B} \subseteq \mathcal{G}_1 \times \mathcal{G}_1$ *is a RYB-Markovian bisimulation of order 1 if and only if, whenever* $(E_1, E_2) \in \mathcal{B}$*, then for all* $a \in AType$*,* $l \in \mathbb{Z}$ *, and equivalence classes* $C \in \mathcal{G}_1/\mathcal{B}$

$$RYB_1(E_1,a,l,C) = RYB_1(E_2,a,l,C)$$ ■

It is easy to see that the union of all the RYB-Markovian bisimulations of order 1 is a RYB-Markovian bisimulation of order 1. Such a union, denoted $\sim_{\mathrm{MB}_1^{\mathrm{RYB}}}$, is called the RYB-Markovian bisimulation equivalence of order 1.

Theorem 1. $\sim_{\mathrm{MB}_1^{\mathrm{RYB}}}$ *is a congruence w.r.t. all the algebraic operators as well as recursive constant definitions.*

Proof. See [4]. ■

Theorem 2. *Let* $\mathcal{A}_1^{\mathrm{RYB}}$ *be the set of axioms obtained from those in Table 2 by replacing* $(\mathcal{A}_4)_1^{\mathrm{RY}}$ *with*

$$(\mathcal{A}_4)_1^{\mathrm{RYB}} \; <a, \tilde{\lambda}_1, (\tilde{y}_1, \tilde{b}_1)>.E + <a, \tilde{\lambda}_2, (\tilde{y}_2, \tilde{b}_2)>.E \; = \; <a, \tilde{\lambda}_1 + \tilde{\lambda}_2, (\tilde{y}_1 + \tilde{y}_2, \frac{\tilde{\lambda}_1}{\tilde{\lambda}_1+\tilde{\lambda}_2} \cdot \tilde{b}_1 + \frac{\tilde{\lambda}_2}{\tilde{\lambda}_1+\tilde{\lambda}_2} \cdot \tilde{b}_2)>.E$$

if $PL(\tilde{\lambda}_1) = PL(\tilde{\lambda}_2)$*. The deductive system* $Ded(\mathcal{A}_1^{\mathrm{RYB}})$ *is sound and complete for* $\sim_{\mathrm{MB}_1^{\mathrm{RYB}}}$ *over the set of nonrecursive terms of* $\mathcal{G}_1$*.*

Proof. See [4]. ■

Theorem 3. *Let* $E_1, E_2 \in \mathcal{E}_1$*. If* $E_1 \sim_{\mathrm{MB}_1^{\mathrm{RYB}}} E_2$ *then the value of the reward based performance measure is the same for* E_1 *and* E_2*.*

Proof. See [4]. ■

5 Mixing Yield and Bonus Rewards

Having the objective of defining a reward based Markovian bisimulation congruence that aggregates as much as possible, the question arises as to whether it is possible to consider just one type of reward instead of two. From the point of view of an equivalence, this can be rephrased in terms of being able to jointly consider yield and bonus rewards. By looking at formulas (1) and (2) and the way transition frequencies are computed, we note that in the continuous time case $<a, \lambda_1, (y_1, b_1)>.E + <a, \lambda_2, (y_2, b_2)>.E$ can be equated to $<a, \lambda_1 + \lambda_2, (y_1 + y_2 + \lambda_1 \cdot b_1 + \lambda_2 \cdot b_2, 0)>.E$, while in the discrete time case $<a, \infty_{l,w_1}, (y_1, b_1)>.E + <a, \infty_{l,w_2}, (y_2, b_2)>.E$ can be equated to $<a, \infty_{l,w_1+w_2}, (y_1 + y_2 + \frac{w_1}{w_1+w_2} \cdot b_1 + \frac{w_2}{w_1+w_2} \cdot b_2, 0)>.E$. This gives rise to a normal form where only yield rewards are actually present, with bonus rewards being zero (or unspecified in the case of passive actions). However, in the discrete time case, we observe that the factor by which each bonus reward must be multiplied is equal to the execution probability of the transition to which it is attached. Since such a probability varies depending on the context in which the term is placed, compositionality is lost. As an example, if we call E_1 and E_2 the two equivalent terms above, respectively, and we take E_3 defined by $<a, \infty_{l,w_3}, (y_3, b_3)>.E$, we have that the aggregated yield reward for $E_1 + E_3$ is $y_1 + y_2 + y_3 + \frac{w_1}{w_1+w_2+w_3} \cdot b_1 + \frac{w_2}{w_1+w_2+w_3} \cdot b_2 + \frac{w_3}{w_1+w_2+w_3} \cdot b_3$, while the aggregated yield reward for $E_2 + E_3$ is $y_1 + y_2 + \frac{w_1}{w_1+w_2} \cdot b_1 + \frac{w_2}{w_1+w_2} \cdot b_2 + y_3 + \frac{w_1+w_2}{w_1+w_2+w_3} \cdot 0 + \frac{w_3}{w_1+w_2+w_3} \cdot b_3$. We introduce below a further improved reward based Markovian bisimulation congruence that mixes yield and bonus rewards in the continuous time case.

Definition 10. *We define partial function aggregated rate-reward* $RR_1 : \mathcal{G}_1 \times AType \times \mathbb{Z} \times \mathcal{P}(\mathcal{G}_1) \multimap (ARate \times ARew) \cup (ARate \times ARew \times ARew)$ *by:*

$$RR_1(E, a, l, C) = \begin{cases} (Rate_1(E, a, l, C), Reward_1(E, a, C)) & \text{if } l = 0 \\ RYB_1(E, a, l, C) & \text{if } l \neq 0 \end{cases}$$

where:

$$Reward_1(E, a, C) = \sum \{\!|\, \tilde{y} + \tilde{\lambda} \cdot \tilde{b} \mid \exists E' \in C.\, E \xrightarrow{a, \tilde{\lambda}, (\tilde{y}, \tilde{b})}_1 E' \wedge PL(\tilde{\lambda}) = 0 \,|\!\}$$

with $RR_1(E, a, l, C) = \perp$ *whenever the multisets above are empty.* ■

Definition 11. *An equivalence relation* $\mathcal{B} \subseteq \mathcal{G}_1 \times \mathcal{G}_1$ *is a RR-Markovian bisimulation of order 1 if and only if, whenever* $(E_1, E_2) \in \mathcal{B}$*, then for all* $a \in AType$*,* $l \in \mathbb{Z}$ *, and equivalence classes* $C \in \mathcal{G}_1/\mathcal{B}$

$$RR_1(E_1, a, l, C) = RR_1(E_2, a, l, C)$$ ■

It is easy to see that the union of all the RR-Markovian bisimulations of order 1 is a RR-Markovian bisimulation of order 1. Such a union, denoted $\sim_{\mathrm{MB}_1^{\mathrm{RR}}}$, is called the RR-Markovian bisimulation equivalence of order 1.

Theorem 4. $\sim_{\mathrm{MB}_1^{\mathrm{RR}}}$ *is a congruence w.r.t. all the algebraic operators as well as recursive constant definitions.*

Proof. See [4]. ■

Theorem 5. *Let* $\mathcal{A}_1^{\mathrm{RR}}$ *be the set of axioms obtained from* $\mathcal{A}_1^{\mathrm{RYB}}$ *by adding*

$$(\mathcal{A}_{4'})_1^{\mathrm{RR}} \quad <a, \lambda, (y, b)>.E = <a, \lambda, (y + \lambda \cdot b, 0)>.E$$

The deductive system $Ded(\mathcal{A}_1^{\mathrm{RR}})$ *is sound and complete for* $\sim_{\mathrm{MB}_1^{\mathrm{RR}}}$ *over the set of nonrecursive terms of* $\mathcal{G}_1$.

Proof. See [4]. ■

Theorem 6. *Let* $E_1, E_2 \in \mathcal{E}_1$. *If* $E_1 \sim_{\mathrm{MB}_1^{\mathrm{RR}}} E_2$ *then the value of the reward based performance measure is the same for* E_1 *and* E_2.

Proof. See [4]. ■

6 Conclusion

In this paper we have improved the performance measure sensitive Markovian bisimulation congruence of [3] in order to aggregate more states and transitions while preserving compositionality and the values of the performance measures. While the congruence of [3] aggregates yield rewards without manipulating bonus rewards at all, the congruence of Def. 11 aggregates also the bonus rewards, provided that they are multiplied by the conditional probabilities of executing the actions to which they are attached, and allows yield rewards and bonus rewards to be used interchangeably in the continuous time case, provided that they are divided/multiplied by the rates of the actions to which they are attached.

The impossibility result of this paper, i.e. the fact that it is not possible to define a performance measure sensitive Markovian bisimulation congruence that allows yield and bonus rewards to be used interchangeably in the discrete time case, emphasizes the necessity of the bonus rewards. In the literature of Markov reward processes it is well known that yield and bonus rewards can be used interchangeably in the continuous time case, and in this paper we have verified that such a property does not violate compositionality. In the continuous time case the yield rewards work well because of the race policy. In particular, the additivity assumption is sound because in every state all the transitions are viewed as being in parallel execution, hence each of them contributes with its yield reward to the accumulation of reward at the state. In the discrete time case, instead, the preselection policy applies, hence the bonus rewards are more natural to express performance measures. Besides being more convenient from the modeling viewpoint, in the discrete time case the bonus rewards are also necessary from the compositionality viewpoint, i.e. they cannot be transformed into yield rewards if we want to get a congruence. In fact, if we transform them into yield rewards, we have that the contribution of the transitions to the accumulation of reward at the state is given by their average bonus reward, i.e. the weighted sum of their bonus rewards with each of them multiplied by the execution probability of the corresponding transition. Since the above mentioned execution probabilities (unlike the rates in the continuous time case) vary depending on the environment in which the state is placed, compositionality is lost.

The performance measure sensitive Markovian bisimulation congruence of Def. 11 aggregates more than that of [3]. The reason is that the new congruence can merge also those transitions that the old congruence cannot merge only because of their different bonus rewards. A quantification of the achieved improvement is left for future research.

We conclude with a practice related observation. Describing the rewards directly within the Markovian process algebra specifications of the systems has the advantage of allowing the specifications to be compositionally manipulated while preserving the values of the performance measures. This advantage on the analysis side is unfortunately diminished by a drawback on the modeling side: the system specifications are obfuscated with performance measure related details. However, the Markovian process algebra specification of a system and the specification of its reward based performance measures of interest can be easily decoupled by separately describing the rewards to be attached to the actions occurring in the system specification. A syntactical preprocessing step, like the one performed by the $\mathrm{EMPA}_{\mathrm{gr}_n}$ based software tool TwoTowers [2], then permits to automatically insert the rewards into the system specification. This avoids burdening the system specifications with rewards at modeling time, eases the specification of additional performance measures for the same system specification, and allows every performance measure specification to be reused for different system specifications.

References

1. C. Baier, J.-P. Katoen, H. Hermanns, *"Approximate Symbolic Model Checking of Continuous Time Markov Chains"*, in Proc. of CONCUR '99, LNCS 1664:146-162
2. M. Bernardo, *"Theory and Application of Extended Markovian Process Algebra"*, Ph.D. Thesis, University of Bologna (Italy), 1999 (`http://www.di.unito.it/~bernardo/`)
3. M. Bernardo, M. Bravetti, *"Performance Measure Sensitive Congruences for Markovian Process Algebras"*, to appear in Theoretical Computer Science, 2001
4. M. Bernardo, M. Bravetti, *"Formal Specification of Performance Measures for Process Algebra Models of Concurrent Systems"*, Tech. Rep. UBLCS-1998-08, University of Bologna (Italy), 1998 (revised 2001)
5. M. Bravetti, M. Bernardo, *"Compositional Asymmetric Cooperations for Process Algebras with Probabilities, Priorities, and Time"*, Tech. Rep. UBLCS-2000-01, University of Bologna (Italy), 2000 (extended abstract in MTCS '00, Electronic Notes in Theoretical Computer Science 39(3))
6. G. Clark, *"Formalising the Specification of Rewards with PEPA"*, in Proc. of PAPM '96, CLUT, pp. 139-160
7. G. Clark, S. Gilmore, J. Hillston, *"Specifying Performance Measures for PEPA"*, in Proc. of ARTS '99, LNCS 1601:211-227
8. R.J. van Glabbeek, S.A. Smolka, B. Steffen, *"Reactive, Generative and Stratified Models of Probabilistic Processes"*, in Information and Computation 121:59-80, 1995
9. B.R. Haverkort, K.S. Trivedi, *"Specification Techniques for Markov Reward Models"*, in Discrete Event Dynamic Systems: Theory and Applications 3:219-247, 1993
10. R.A. Howard, *"Dynamic Probabilistic Systems"*, John Wiley & Sons, 1971
11. V.F. Nicola, *"Lumping in Markov Reward Processes"*, Tech. Rep. RC-14719, IBM T.J. Watson Research Center, Yorktown Heights (NY), 1990
12. W.H. Sanders, J.F. Meyer, *"A Unified Approach for Specifying Measures of Performance, Dependability, and Performability"*, in Dependable Computing and Fault Tolerant Systems 4:215-237, 1991
13. J.E. Voeten, *"Temporal Rewards for Performance Evaluation"*, in Proc. of PAPM '00, Carleton Scientific, pp. 511-522