

An Intermediate Language for the Stochastic Simulation of Biological Systems

Roberto Barbuti, Giulio Caravagna,
Andrea Maggiolo–Schettini, Paolo Milazzo

*Dipartimento di Informatica, Università di Pisa
Largo Bruno Pontecorvo 3, 56127 Pisa, Italy*

Abstract

We introduce Stochastic String MultiSet Rewriting (sSMSR), and propose this formalism as an intermediate language for the simulation of biomolecular systems. Higher level formalisms for biological systems description can be translated into sSMSR, and the features of sSMSR allow the development of efficient simulators. In this paper we show the encoding into sSMSR of two formalisms for the description of biological systems, namely Stochastic CLS and the Stochastic π -calculus. We prove soundness and completeness of both the encodings.

Key words: MultiSet Rewriting, Stochastic Simulation, Stochastic Calculus of Looping Sequences, Stochastic π -calculus.

1 Introduction

Stochastic simulation of biomolecular systems is usually based on Gillespie's framework [11] which describes a system as a multiset of elements representing molecules. A system transformation due to a chemical reaction among molecules is described as the replacement, in the multiset representing the system, of the elements representing reactants with those representing products of the reaction. These replacements are made with a frequency that depends on an exponentially distributed random variable. Multisets and their transformations, which can be formalized as Stochastic MultiSet Rewriting (sMSR) [12], can be easily implemented and many tools exist for the purpose.

Email addresses: barbuti@di.unipi.it (Roberto Barbuti),
caravagn@di.unipi.it (Giulio Caravagna), maggiolo@di.unipi.it (Andrea Maggiolo–Schettini), milazzo@di.unipi.it (Paolo Milazzo).

In the last years the need has arisen to describe biological phenomena at system level. This can be done by ignoring structural and behavioral details of individual system components and by taking into account organization of components in compartments and interaction capabilities of these components. The formalism sMSR does not allow descriptions at this high level and, consequently, many new formalisms, sometimes adaptations of existing ones, have been proposed. We mention, as examples, the κ -calculus [9], the Stochastic π -calculus [18], BioAmbients [19], Brane Calculi [6], P Systems [15] and Stochastic CLS [13]. For some of the mentioned formalisms specific simulators exist (e.g. SPiM [21] based on the Stochastic π -calculus, and CytoSim and PSym [16] based on P Systems and the CLSm [20] based on Stochastic CLS). However, in general, the development of simulators for formalisms which allow the description of complex biological structures and operations may require the use of complex data structures and algorithms. Moreover, the translation from a high level formalism into sMSR, which allows the use of existing simulators, may pose some difficulties or be impossible at all due to the non Turing-completeness of sMSR. Hence the idea arises of defining an intermediate language into which high level descriptions can be translated and for which an efficient simulator can be developed.

In this paper we propose an extension of sMSR, called *Stochastic String MultiSet Rewriting (sSMSR)*, in which multiset elements are strings and rewrite rules are extended with some features that ease the translation of higher level formalisms. Among these features we have variables, that can be used to match either individual symbols or portions of the strings which are involved in the application of a rule. Moreover, we have a unique matching operator and a maximal matching operator, which allow a rule to be applicable to a multiset of strings only if such a multiset contains a single string with a certain prefix (unique matching), or only if all the strings with the same given prefix are involved in the rule application (maximal matching). Finally, we have that fresh symbols, namely symbols that are present neither in the multiset of strings to which the rule is applied, nor in any other rewrite rule, can be generated when a rewrite rule is applied.

The features of sSMSR can ease the translation of high level languages. The idea is to compute from a tree representation of a term of a high level language a multiset of strings, each representing a path from the root of the tree to a leaf. Variables in sSMSR rewrite rules can be used to encode variables in the high level language. Unique and maximal matchings can be used to translate high level languages with a notion of membrane: the former operator can be used to encode operations which require that a membrane contains a precise number of elements, and the latter operator can be used to encode operations that apply on the whole content of a membrane.

The use of strings as multiset elements and of operations on strings in rewrite

rules allows the development of a simulator for sSMSR based on efficient data structures and pattern matching algorithms. Moreover, by developing analysis and verification techniques on this rather simple intermediate language, one could apply such techniques to study systems described in a higher level language via translation into sSMSR.

Most of high level description languages belong to two main classes of formalisms, namely process calculi and rewriting systems. In order to show that sSMSR can be suitably used as an intermediate language, we define the encoding into sSMSR of two formalisms which are representative of the two mentioned classes, namely Stochastic CLS and the Stochastic π -calculus. The former has been shown to be suitable for describing various kinds of biological phenomena and it has been chosen also because the rewriting mechanism on which it is based makes the translation into sSMSR easy to understand. The latter has been widely used for the description of biological systems and many other formalisms are defined as extensions of the Stochastic π -calculus. We prove soundness and completeness for both the encodings we give.

The paper is structured as follows. In Section 2 we recall the definition of sMSR. In Section 3 we define sSMSR as an extension of sMSR. In Section 4 we show the encoding of Stochastic CLS and in Section 5 the encoding of the Stochastic π -calculus. Finally, in Section 6 we mention some related works and conclude.

2 Stochastic MultiSet Rewriting

In this section we recall the Stochastic MultiSet Rewriting (sMSR) [12]. The syntax we give differs slightly from that given in [12].

Let us assume a countably infinite alphabet \mathcal{E} ranged over by a, b, c, \dots . Terms of sMSR are multisets defined as follows.

Definition 1 (Multisets) *Multisets M are given by the following grammar:*

$$M ::= \epsilon \quad | \quad a \quad | \quad M \mid M$$

where $a \in \mathcal{E}$ and ϵ is the empty multiset. We denote with \mathcal{M} the set of all multisets.

In the syntax of multisets an alphabet symbol a represents the singleton $\{a\}$ and $M_1 \mid M_2$ represents the union of multisets M_1 and M_2 . We assume the *structural congruence* \equiv to be the least congruence on multisets satisfying axioms $M_1 \mid M_2 \equiv M_2 \mid M_1$, $M_1 \mid (M_2 \mid M_3) \equiv (M_1 \mid M_2) \mid M_3$ and $M \mid \epsilon \equiv M$.

The structural congruence \equiv allows us to formally define the algebraic multiset operations \in , \subseteq , \subset , \cup , \cap and \setminus on sMSR terms. For example, $a \in M$ corresponds to $\exists M' \in \mathcal{M}. M \equiv a \mid M'$ and $M \subseteq M'$ corresponds to $\exists M'' \in \mathcal{M}. M' \equiv M \mid M''$. Furthermore, given a multiset $M \in \mathcal{M}$ we denote with \overline{M} the set of all the distinct objects that appear in M , namely $\overline{M} = \{a \in \mathcal{E} \mid a \in M\}$, and we assume a function $\mathbf{n} : \mathcal{M} \times \mathcal{E} \rightarrow \mathbb{N}$ such that $\mathbf{n}(M, a)$ gives the number of occurrences of object a in the multiset M . For example, $\mathbf{n}(a \mid a, a) = 2$ and $\mathbf{n}(a \mid a, b) = 0$.

Definition 2 (Stochastic Rewrite Rules) *A stochastic rewrite rule is a triple (M_1, M_2, k) , denoted as $M_1 \xrightarrow{k} M_2$, where $M_1, M_2 \in \mathcal{M}$, $M_1 \neq \epsilon$ and $k \in \mathbb{R}$. With \mathfrak{R} we denote the set of all stochastic rewrite rules.*

In the following we will often write $R : M_1 \xrightarrow{k} M_2$ to mean that R can be used as a shorter notation for the stochastic rewrite rule $M_1 \xrightarrow{k} M_2$. Stochastic rewrite rules can be used to describe possible evolutions of a multiset. A rule $M_1 \xrightarrow{k} M_2$ applied to a multiset M replaces one of the occurrences of M_1 in M with M_2 . In accordance with Gillespie's algorithm [11], the rate of application of the rule is given by k multiplied by the number of combinations of M_1 in M . Namely,

$$\text{Rate}(k, M_1, M) = k \cdot \prod_{a \in \overline{M_1}} \binom{\mathbf{n}(M, a)}{\mathbf{n}(M_1, a)}.$$

The function *Rate* is used in the definition of the semantics of sMSR.

Definition 3 (Semantics) *Given a finite set of stochastic rewrite rules $\mathcal{R} \subset \mathfrak{R}$, the semantics of sMSR is the least transition relation $\xrightarrow{R, r}$, with $R \in \mathcal{R}$ and $r \in \mathbb{R}$, closed with respect to \equiv and satisfying the following inference rule:*

$$\frac{R : M_1 \xrightarrow{k} M_2 \in \mathcal{R}}{M_1 \mid M_3 \xrightarrow{R, \text{Rate}(k, M_1, M_1 \mid M_3)} M_2 \mid M_3}$$

The semantics is a labeled transition system in which each transition corresponds to the application of a stochastic rewrite rule. The label of a transition contains the rule that has been applied and the application rate of such a rule. The label contains the applied rule in order to distinguish two transitions between the same states and with the same rate, but caused by the application of different rules. The following example shows such a situation: given $M \equiv a \mid a \mid a \mid b$, $R_1 : a \xrightarrow{2} c$ and $R_2 : a \mid b \xrightarrow{2} c \mid b$ we have that $\text{Rate}(2, a, M) = \text{Rate}(2, a \mid b, M) = 6$. By applying R_1 and R_2 to M we obtain these two transitions $M \xrightarrow{R_1, 6} a \mid a \mid b \mid c$ and $M \xrightarrow{R_2, 6} a \mid a \mid b \mid c$, respectively, that are distinguished only by the rule in the label.

An sMSR model is a pair $\langle M, \mathcal{R} \rangle$ where M is a multiset modeling the initial state of the described systems and \mathcal{R} is a set of stochastic rewrite rules modeling the events that may occur in the system.

3 Stochastic String MultiSet Rewriting

In this section we introduce the *Stochastic String MultiSet Rewriting* (sSMSR) as an extension of sMSR. The formalism sSMSR extends sMSR with strings, rather than individual alphabet symbols, as multiset elements, and with richer rewrite rules. The formalism sSMSR is also the stochastic extension of the String MultiSet Rewriting formalism presented in [2].

As in sMSR we assume a countably infinite alphabet \mathcal{E} ranged over by a, b, c, \dots . We assume also a total order \prec on alphabet symbols. Given a finite subset A of \mathcal{E} , we denote with $\max(A)$ the greatest symbol in A with respect to \prec , and with $\text{next}(A)$ the least symbol a such that $\max(A) \prec a$. Similarly, we denote with $\max_n(A)$ the set of the n least symbols that are greater than $\max(A)$.

Definition 4 (Terms) String multisets M_S and strings S are given by the following grammar:

$$\begin{array}{l} M_S ::= S \quad | \quad M_S | M_S \\ S ::= \epsilon \quad | \quad a \quad | \quad S \cdot S \end{array}$$

where $a \in \mathcal{E}$ and ϵ is the empty string. We denote with \mathcal{M}_S the set of all string multisets and with \mathcal{S} the set of all strings.

Strings over \mathcal{E} can be constructed by means of the concatenation operator \cdot , with ϵ representing the concatenation of zero elements. Multisets of strings can be constructed by means of the union operator $|$. Note that any multiset of sMSR is also a string multiset of sSMSR, namely $\mathcal{M} \subset \mathcal{M}_S$. As for sMSR, we assume a structural congruence relation \equiv on string multisets. In this case we need also a similar relation \equiv_S on strings defined as the least congruence satisfying axioms $S_1 \cdot (S_2 \cdot S_3) \equiv_S (S_1 \cdot S_2) \cdot S_3$ and $\epsilon \cdot S \equiv_S S \cdot \epsilon \equiv_S S$. Hence, the structural congruence \equiv on string multisets is the least congruence including \equiv_S and satisfying axioms $M_1 | M_2 \equiv M_2 | M_1$, $M_1 | (M_2 | M_3) \equiv (M_1 | M_2) | M_3$ and $M | \epsilon \equiv M$. The definition of algebraic operations on multisets of sMSR can be trivially extended to string multisets of sSMSR.

Now we introduce sSMSR patterns, that are terms enriched with variables and with two different matching operators. We assume a countably infinite set of variables $\mathcal{V} = \mathcal{V}_\mathcal{E} \cup \mathcal{V}_S \cup \mathcal{V}_M$ where $\mathcal{V}_\mathcal{E}$ is a countably infinite set of *element variables*, ranged over by x, y, z, \dots , \mathcal{V}_S is a countably infinite set of

string variables, ranged over by $\tilde{x}, \tilde{y}, \tilde{z}, \dots$, and \mathcal{V}_M is a countably infinite set of *multiset variables*, ranged over X, Y, Z, \dots . We assume $\mathcal{V}_E, \mathcal{V}_S$ and \mathcal{V}_M to be pairwise disjoint and that, as for alphabet symbols, a total order \prec exists on element variables. We assume also that for each multiset variable X there exists a countably infinite subset of \mathcal{V}_S , denoted $\mathcal{V}_S(X)$, for which an ordering is defined. We denote the elements of $\mathcal{V}_S(X)$ as \tilde{x}_i , where i is the position of the element in the ordering. Moreover, we assume that for any $X, Y \in \mathcal{V}_M$ such that $X \neq Y$, it holds $\mathcal{V}_S(X) \cap \mathcal{V}_S(Y) = \emptyset$.

Definition 5 (Patterns) Multiset patterns MP and string patterns SP are given by the following grammar:

$$\begin{aligned} MP & ::= SP \mid MP \mid MP \mid \{\!\{SP\}\!\}_X \mid \{SP\} \\ SP & ::= \epsilon \mid a \mid SP \cdot SP \mid \tilde{x} \mid x \end{aligned}$$

where ϵ is the empty string, $a \in \mathcal{E}$, $x \in \mathcal{V}_E$, $\tilde{x} \in \mathcal{V}_S$ and $X \in \mathcal{V}_M$. We denote with \mathcal{MP} and \mathcal{SP} the sets of all multiset and string patterns, respectively.

A string pattern is a concatenation of alphabet symbols, element variables and string variables, with ϵ representing the concatenation of zero elements. A multiset pattern is either a string pattern, or a union of multiset patterns, or a *maximal matching* $\{\!\{SP\}\!\}_X$, or a *unique matching* $\{SP\}$. We assume the structural congruence relation to be trivially extended to multiset patterns.

Multiset patterns are used to define stochastic rewrite rules of sSMSR. A stochastic rewrite rule is composed by a pair of multiset patterns and a rate constant. The first multiset pattern of the pair describes the term that is modified by an application of the rule and the second describes how the term changes after the application. Variables in patterns allow a rewrite rule to be applicable to any term that can be obtained by properly instantiating them. The maximal matching operator $\{\!\{SP\}\!\}_X$ represents a multiset of strings which have as prefix the same instantiation of the string pattern SP . The unique matching operator $\{SP\}$ represents the multiset containing a single string obtained as instantiation of the string pattern SP ; the union of n copies of an instance of this operator represents the multiset containing exactly n (identical) strings obtained as instantiations of the string pattern SP . Roughly speaking, the maximal matching operator allows a rewrite rule to rewrite all the occurrences of strings with the same prefix (the instantiation of SP) that are contained in a string multiset. On the contrary, a rewrite rule containing in its left pattern n copies of the unique matching operator $\{SP\}$ is applicable only to string multisets containing exactly n copies of a string obtained by instantiating SP . For example, given the string multiset $M = \{a \cdot b \cdot c \mid a \cdot b \cdot c \mid a \cdot b \cdot d \mid c \cdot d \mid c \cdot d \cdot e\}$, the maximal matching $\{\!\{a \cdot b\}\!\}_X$ represents the string multiset $\{a \cdot b \cdot c \mid a \cdot b \cdot c \mid a \cdot b \cdot d\}$, that is the multiset of all the strings in M prefixed by $a \cdot b$. As regards the unique matching we have that a rule

containing a single occurrence of $\{c \cdot d\}$ in its left pattern could be applied to M , while a rule containing a single occurrence of $\{a \cdot b \cdot c\}$ in its left pattern could not, because M contains more than one copy of $a \cdot b \cdot c$.

An *instantiation* is a function $\sigma : \mathcal{V}_{\mathcal{E}} \cup \mathcal{V}_{\mathcal{S}} \rightarrow \mathcal{E} \cup \mathcal{S}$ such that $\sigma(x) \in \mathcal{E}$ and $\sigma(\tilde{x}) \in \mathcal{S}$ for $x \in \mathcal{V}_{\mathcal{E}}$ and $\tilde{x} \in \mathcal{V}_{\mathcal{S}}$, respectively. We denote with Σ the set of all instantiations. Given $MP \in \mathcal{MP}$, with $MP\sigma$ we denote the multiset obtained by replacing each occurrence of an element or string variable v appearing in MP with the corresponding instantiation $\sigma(v)$. Given a set of variables $V \subseteq \mathcal{V}_{\mathcal{E}} \cup \mathcal{V}_{\mathcal{S}}$, we denote with $\sigma(V)$ the set $\{\sigma(v) \mid v \in V\}$. Note that instantiations are not defined for multiset variables. In the semantics of sSMSR such variables, which appear only as subscripts of maximal matching operators, will be replaced by a set of string variables before pattern instantiation. This replacement will be performed by applying a *pattern expansion* function.

Definition 6 (Pattern Expansion) A pattern expansion is a function $\langle _ \rangle_- : \mathcal{MP} \times (\mathcal{V}_{\mathcal{M}} \rightarrow \mathbb{N}) \rightarrow \mathcal{M}$ recursively defined as follows:

$$\begin{aligned} \langle SP \rangle_{\rho} &= \langle \{SP\} \rangle_{\rho} = SP \\ \langle \{SP\}_X \rangle_{\rho} &= SP \cdot \tilde{x}_1 \mid \dots \mid SP \cdot \tilde{x}_{\rho(X)} \\ \langle MP_1 \mid MP_2 \rangle_{\rho} &= \langle MP_1 \rangle_{\rho} \mid \langle MP_2 \rangle_{\rho} \end{aligned}$$

where \tilde{x}_i is the i -th element of $\mathcal{V}_{\mathcal{S}}(X)$.

A pattern expansion transforms each maximal matching operator $\{SP\}_X$ into a union of sequence patterns, all with the same prefix SP and each followed by a different sequence variable. The number of sequence patterns to be created by the expansion of a maximal matching operator is given by an auxiliary function $\rho : \mathcal{V}_{\mathcal{M}} \rightarrow \mathbb{N}$ which is a parameter of the pattern expansion function. The result of the expansion of the unique matching operator containing a sequence pattern is the sequence pattern itself and, analogously, the expansion of a sequence pattern is the sequence pattern itself.

Given a multiset pattern MP , we denote with $Var(MP)$ the set containing all element and string variables occurring in MP and all sets $\mathcal{V}_{\mathcal{S}}(X)$ for each multiset variable X occurring in MP . For example, $Var(a \cdot \tilde{x} \mid a \cdot x \mid \{d\}_Y) = \{\tilde{x}, x\} \cup \{\tilde{y}_i \mid i \in \mathbb{N}\}$. Moreover, we denote with $Symbols(MP)$ the set of all alphabet symbols occurring in MP . For example, $Symbols(a \cdot \tilde{x} \mid a \cdot x \mid \{d \cdot e\}) = \{a, d, e\}$. We assume Var and $Symbols$ to be trivially extended to sets of sSMSR patterns. Given a multiset pattern MP , we say that an alphabet symbol a is *fresh* in MP if $a \notin Symbols(MP)$.

Now we can define stochastic rewrite rules of sSMSR.

Definition 7 (Stochastic Rewrite Rules) A stochastic rewrite rule is a

triple (MP_1, MP_2, k) , denoted as $MP_1 \xrightarrow{k} MP_2$, where $MP_1, MP_2 \in \mathcal{MP}$, $MP_1 \neq \epsilon$, $MP_1 \neq MP_2$, $(Var(MP_2) \setminus Var(MP_1)) \subset \mathcal{V}_\mathcal{E}$ and $k \in \mathbb{R}$. With \mathfrak{R} we denote the set of all possible stochastic rewrite rules.

As in sMSR we will often write $R : MP_1 \xrightarrow{k} MP_2$ to mean that R can be used as a shorter notation for the stochastic rewrite rule $MP_1 \xrightarrow{k} MP_2$. Given a stochastic rewrite rule $R : MP_1 \xrightarrow{k} MP_2$, we write $Var(R)$ for $Var(MP_1) \cup Var(MP_2)$ and $Symbols(R)$ for $Symbols(MP_1) \cup Symbols(MP_2)$. Moreover, given a set of rules \mathcal{R} , we write $Symbols(\mathcal{R})$ for $\bigcup_{R \in \mathcal{R}} Symbols(R)$.

In a stochastic rewrite rule $MP_1 \xrightarrow{k} MP_2$ some element variables may appear in MP_2 but not in MP_1 . We call these variables *free* and we denote them with $FV(MP_1 \xrightarrow{k} MP_2)$, namely $FV(MP_1 \xrightarrow{k} MP_2) = \{v \mid v \in Var(MP_2) \wedge v \notin Var(MP_1)\}$. We call variables appearing in MP_1 *bound*, namely $BV(MP_1 \xrightarrow{k} MP_2) = Var(MP_1)$.

We permit free variables to be used in stochastic rewrite rules to allow generation of fresh symbols during rewrite rule applications. In particular, in the semantics of rule application we will require that all the free variables are instantiated to symbols that are fresh with respect to the string multiset to which the rule is applied and with respect to all the stochastic rewrite rules. This means that free variables have a meaning similar to the existentially quantified variables in first-order multiset rewriting [7].

Now we define the semantics of sSMSR. In the definition we assume the function *Rate* introduced in Section 2 to be extended to string multisets as follows

$$Rate(k, M_1, M) = k \cdot \prod_{S \in \overline{M_1}} \binom{\mathbf{n}(M, S)}{\mathbf{n}(M_1, S)}$$

where $\mathbf{n}(M, S)$ and \overline{M} are the extensions of the corresponding functions defined in Section 2. The former gives the number of occurrences of string S as complete strings (not as portions of longer strings) in the string multiset M and the latter the set of strings occurring as complete strings in the string multiset M . For example, given $M \equiv a \cdot b \mid a \cdot b \cdot c \mid a \cdot b \mid a \cdot c$, we have $\mathbf{n}(M, a \cdot b) = 2$ and $\overline{M} = \{a \cdot b, a \cdot b \cdot c, a \cdot c\}$.

Definition 8 (Semantics) *Given a finite set of stochastic rewrite rules $\mathcal{R} \subset \mathfrak{R}$, the semantics of sSMSR is the least labeled transition relation $\xrightarrow{R, M, r}$, with $R \in \mathcal{R}, M \in \mathcal{M}$ and $r \in \mathbb{R}$, closed with respect to \equiv and satisfying the*

following inference rule:

$$\begin{array}{c}
R : MP_1 \xrightarrow{k} MP_2 \in \mathcal{R} \quad \sigma \in \Sigma \quad \exists \rho. (\langle MP_1 \rangle_\rho) \sigma \equiv M_1 \wedge (\langle MP_2 \rangle_\rho) \sigma \equiv M_2 \\
\forall S \in \{SP\sigma \mid \{SP\}_X \in MP_1 \vee \{SP\} \in MP_1\}. \#S' \in \mathcal{S}. (S \cdot S') \in M_3 \\
Symbols(\sigma(FV(R))) = next_{|FV(R)|}(Symbols(M_1 \mid M_3) \cup Symbols(\mathcal{R})) \\
\forall x, y \in FV(R). x \prec y \implies \sigma(x) \prec \sigma(y) \\
\hline
M_1 \mid M_3 \xrightarrow{R, \diamond(MP_1)\sigma, Rate(k, \diamond(MP_1)\sigma, M_1 \mid M_3)} M_2 \mid M_3
\end{array}$$

where $\diamond(MP_1)$ denotes the multiset pattern obtained by replacing all occurrences of maximal and unique matchings in MP_1 with ϵ .

The semantics of sSMSR is based on the same idea of that of sMSR: transitions represent rewrite rule applications and are enriched with labels containing the rule that has been applied and the application rate computed by the function *Rate*. The four main differences with respect to the semantics of sMSR are the following: (i) an instantiation σ and an expansion ρ must exist such that the left hand side of the applied rule can match a portion of the considered string multiset; (ii) some constraints are included in the premise of the inference rule in accordance with the meaning of the maximal and unique matching operators and to ensure the correct instantiation of free variables; (iii) the strings represented by the maximal and unique matching operators are not considered as reactants in the computation of the application rate; (iv) transitions labels are enriched with a string multiset.

The motivation for (i) is obvious. As regards (ii) the constraint in the second line of the premise of the inference rule ensures that all the strings having a prefix obtained by the instantiation of some string pattern appearing in a matching operator, are contained in the string multiset corresponding to the instantiation of the left hand side of the applied rewrite rule. This constraint and the definition of pattern expansion ensure maximality and uniqueness of the maximal and unique matching operators, respectively. Moreover, the constraints in the third and fourth lines of the premise of the inference rule ensure that the free variables of the applied rewrite rule are instantiated with symbols and strings that are different from each other and fresh with respect to the current string multiset and all the rewrite rules. More precisely, free variables are instantiated with fresh symbols that are the immediate successors of the symbols in M_1, M_3 and \mathcal{R} . Such fresh symbols are assigned to element variables by preserving their ordering. This implies that there is a unique possible instantiation of free variables and, consequently, this ensures finitary branching. As regards (iii), all the strings in a multiset represented by a maximal matching operator (in a left pattern of a rule) must be considered when the rule is applied. The set of all these strings corresponds to a single reactant. Moreover, the string represented by a unique matching operator is

ensured to occur only once in the multiset to be rewritten. In both cases, the considered strings contribute to the application rate, computed by *Rate*, by a factor $\binom{1}{1} = 1$, and, consequently, they can be omitted in the computation. Finally, as regards (iv), we have that the additional label (representing reactants) is necessary to distinguish two transitions performed by applying the same rule with the same rate but with a different instantiation of variables. For example, let $R : a \cdot x \mid b \cdot y \xrightarrow{2} a \cdot x$ and let $M \equiv a \cdot a \mid a \cdot b \mid b \cdot c$. We have $\text{Rate}(2, a \cdot a \mid b \cdot c, M) = \text{Rate}(2, a \cdot b \mid b \cdot c, M) = 2$. By applying R to M we obtain the transitions $M \xrightarrow{R, a \cdot a \mid b \cdot c, 2} a \cdot a \mid a \cdot b$ and $M \xrightarrow{R, a \cdot b \mid b \cdot c, 2} a \cdot a \mid a \cdot b$ that are distinguished only by the new label.

As an example, given multiset $M \equiv a \cdot b \cdot c \mid a \cdot b \cdot d \mid b \mid b$ and rules $R_1 : b \mid \{a \cdot x\}_X \xrightarrow{k_1} c \cdot y$, $R_2 : \{a \cdot b\} \xrightarrow{k_2} c$ we have that $FV(R_1) = \{y\}$, $BV(R_1) = \{x\} \cup \{x_i \mid i \in \mathbb{N}\}$, $\text{Symbols}(R_1) = \text{Symbols}(R_2) = \{a, b, c\}$, $FV(R_2) = \emptyset$ and $BV(R_2) = \emptyset$. Given a function ρ such that $\rho(X) = 2$ the expansion of patterns with respect to ρ is $\langle b \mid \{a \cdot x\}_X \rangle_\rho = b \mid a \cdot x \cdot \tilde{x}_1 \mid a \cdot x \cdot \tilde{x}_2$ and $\langle \{a \cdot b\} \rangle_\rho = a \cdot b$. Furthermore, given an instantiation function $\sigma \in \Sigma$ such that $\sigma = \{(x, b), (\tilde{x}_1, c), (\tilde{x}_2, d), (y, e)\}$, when applying R_1 to M , we may have the following transition of the semantics $M \xrightarrow{R_1, a \cdot b \cdot c \mid a \cdot b \cdot d \mid b, 2 \cdot k_1} c \cdot e \mid b$ where $\text{Rate}(k_1, b, a \cdot b \cdot c \mid a \cdot b \cdot d \mid b) = 2 \cdot k_1$. Differently, as regards the application of R_2 to M , we have that no possible transitions of the semantics can be derived because the constraint on the matching operators is not satisfied. Note that if either $a \cdot b \cdot d$ or $a \cdot b \cdot c$ would not have been in M , then also R_2 would be applicable with $b \mid b \mid c$ as result.

Similarly to sMSR, the notion of model in sSMSR is a pair $\langle M, \mathcal{R} \rangle$, where M is a string multiset and \mathcal{R} is a set of stochastic rewrite rules. It is easy to see that a sMSR model is also an sSMSR model. The following proposition states that the semantics of sMSR models is preserved by the semantics of sSMSR.

Proposition 9 *Given a set of sMSR rules \mathcal{R} and a multiset M , for any $R : M_1 \xrightarrow{k} M_2 \in \mathcal{R}$ it holds: $M \xrightarrow{R, r} M' \iff M \xrightarrow{R, M_1, r} M'$.*

4 Encoding Stochastic CLS into sSMSR

In this section we recall the definition of Stochastic CLS [13] and define its encoding into sSMSR.

Let \mathcal{E}_{cls} be a possibly infinite alphabet of symbols ranged over by a, b, c, \dots .
Terms T and *sequences* S are given by the following grammar:

$$\begin{aligned} T &::= S \mid (T)^L \mid T \mid T \mid T \\ S &::= \epsilon \mid a \mid S \cdot S \end{aligned}$$

The set of all terms is denoted with \mathcal{T} .

Let \equiv_S be the least congruence on Stochastic CLS sequences satisfying

$$S_1 \cdot (S_2 \cdot S_3) \equiv_S (S_1 \cdot S_2) \cdot S_3 \quad S \cdot \epsilon \equiv_S \epsilon \cdot S \equiv_S S$$

the *structural congruence* \equiv is the least congruence on Stochastic CLS terms including \equiv_S and satisfying

$$T_1 \mid T_2 \equiv T_2 \mid T_1 \quad T_1 \mid (T_2 \mid T_3) \equiv (T_1 \mid T_2) \mid T_3 \quad T \mid \epsilon \equiv T \quad (\epsilon)^L \mid \epsilon \equiv \epsilon$$

Let TV, SV and \mathcal{X} be infinite pairwise disjoint sets of variables called term, sequence and element variables, respectively. *Left patterns* P_L and *right patterns* P_R of Stochastic CLS are given by the following grammar:

$$\begin{aligned} P_L &::= SP \mid (P_X)^L \mid P_X \mid P_L \mid P_L \\ P_X &::= P_L \mid P_L \mid X \\ P_R &::= SP \mid (P_R)^L \mid P_R \mid P_R \mid P_R \mid X \\ SP &::= \epsilon \mid a \mid SP \cdot SP \mid \tilde{x} \mid x \end{aligned}$$

where X, \tilde{x} and x are generic elements of TV, SV and \mathcal{X} , respectively. The sets of all left and right patterns are denoted with \mathcal{P}_L and \mathcal{P}_R , respectively.

Let $Var(P)$ denote the set of variables occurring in a left or right pattern P .

A *stochastic rewrite rule* is a triple (P_L, P_R, k) , denoted $P_L \xrightarrow{k} P_R$, where $P_L \in \mathcal{P}_L, P_R \in \mathcal{P}_R, P_L \neq \epsilon, P_L \not\equiv P_R, k \in \mathbb{R}$ and such that $Var(P_R) \subseteq Var(P_L)$.

Fig. 1. The syntax of Stochastic CLS.

4.1 Definition of Stochastic CLS

Stochastic CLS is based on term rewriting. The syntax of terms and rewrite rules of Stochastic CLS is summarized in Figure 1. For the sake of simplicity, we consider the revised version of Stochastic CLS introduced in [3].

In Stochastic CLS we have a sequencing operator \cdot , a looping operator $(-)^L$, a parallel composition operator \mid and a containment operator \mid . Sequencing can be used to concatenate elements of the alphabet \mathcal{E}_{cls} . The empty sequence ϵ denotes the concatenation of zero symbols. By definition, looping and containment are always applied together, hence we can consider them as a single binary operator $(-)^L \mid$. Looping and containment allow the representation of membranes with their contents. For example, the term

$(a \mid b)^L \rfloor c$ represents a membrane with the elements a and b on its surface and containing the element c . Brackets can be used to indicate the order of application of the operators, and we assume $(_)^L \rfloor _$ to have precedence over $_ \mid _$. The structural congruence relation \equiv of Stochastic CLS expresses associativity of both \cdot and \mid , commutativity of the latter and the neutral role of ϵ with respect to all the operators. Patterns are terms extended with variables of three kinds: element variables \mathcal{X} , sequence variables SV and term variables TV . We denote by \mathcal{V} the set of all variables, $\mathcal{V} = \mathcal{X} \cup SV \cup TV$. The three kinds of variables can be instantiated into alphabet symbols, sequences and terms, respectively, by some instantiation function σ . Let Σ_{cls} be the set of all instantiation functions. In accordance with the restrictions on the use of term variables introduced in [3], we have two different kinds of patterns, left patterns and right patterns, to be used as left and right hand sides of rewrite rules, respectively. Actually, the restrictions are such that term variables cannot be used as components of a parallel composition at the top level of left pattern and at most one term variable can be used in a parallel composition of left patterns. A biological interpretation of these restrictions is given in [3]. The use of term variables in right patterns is not restricted. A stochastic rewrite rule is hence composed by a left pattern P_L , a right pattern P_R and a rate constant k . We assume the structural congruence to be trivially extended to patterns.

The semantics of Stochastic CLS is recalled in Figure 2, where \overline{T} denotes the set of components of the top level parallel composition of T . For example, if $T \equiv a \mid (a)^L \rfloor (b \mid c) \mid c \cdot d$, then $\overline{T} = \{a, (a)^L \rfloor (b \mid c), c \cdot d\}$. Moreover, $\mathbf{n}(T_1, T_2)$ is the analogous in Stochastic CLS of the corresponding function defined in Section 2 for sMSR.

In the definition of the semantics some difficulties arise due to the presence of term variables in the stochastic rewrite rules. For example, a stochastic rewrite rule as $(a \mid X)^L \rfloor (b \mid Y) \xrightarrow{k} (c \mid X)^L \rfloor Y$ is typically used to model a chemical reaction between a molecule a on the surface of some membrane (represented by the application of the looping operator) and a molecule b inside the membrane. The product of the reaction is a complex c placed on the membrane surface. In accordance with standard chemical kinetics, this reaction should have a rate that is proportional to number of possible combinations of a and b molecules, that is the product of the numbers of a and b molecules in the instantiation of X (plus one, represented by symbol a in the rule) and in the instantiation of Y (plus one, represented by symbol b in the rule), respectively. In general, given an instantiation function σ , the computation of the number of combinations of reactants of a left pattern P_L is given by $comb(P_L, \sigma)$.

Another difficulty arises in the definition of the semantics of the parallel composition operator. For example, the rule $a \mid b \xrightarrow{k} c$ can be applied to the term $a \mid b$ with 1 as the number of possible combinations of reactants. If the rule has

Let $comb : \mathcal{P}_L \times \Sigma_{cls} \rightarrow \mathbb{N}$ be recursively defined as follows:

$$\begin{aligned} comb(P_{L1} \mid P_{L2}, \sigma) &= comb(P_{L1}, \sigma) \cdot comb(P_{L2}, \sigma) \\ comb((P_{X1})^L \mid P_{X2}, \sigma) &= comb'(P_{X1}, \sigma) \cdot comb'(P_{X2}, \sigma) \\ comb(SP, \sigma) &= 1 \end{aligned}$$

where $comb'$ is defined as follows:

$$\begin{aligned} comb'(P_L \mid X, \sigma) &= \prod_{T \in \overline{P_L \sigma}} \binom{\mathbf{n}(P_L \sigma \mid \sigma(X), T)}{\mathbf{n}(P_L \sigma, T)} \cdot comb(P_L, \sigma) \\ comb'(P_L, \sigma) &= comb(P_L, \sigma) \end{aligned}$$

Moreover, let $binom : \mathcal{T} \times \mathcal{T} \times \mathcal{T} \rightarrow \mathbb{Q}$ be defined as follows:

$$binom(T_1, T_2, T_3) = \prod_{T \in \overline{T_1 \cup T_2 \cup T_3}} \prod_{i=1}^{\mathbf{n}(T_3, T)} \frac{\mathbf{n}(T_2, T) + i}{\mathbf{n}(T_2, T) - \mathbf{n}(T_1, T) + i}$$

Given a finite set of stochastic rewrite rules \mathcal{R} , let $\xrightarrow{R, T, r, b}$, with $R \in \mathcal{R}, T \in \mathcal{T}, r \in \mathbb{R}$ and $b \in \mathbb{Q}$, be the least labeled transition relation on terms closed with respect to \equiv and satisfying the following inference rules:

$$\begin{aligned} 1. & \frac{R : P_L \xrightarrow{k} P_R \in \mathcal{R} \quad \sigma \in \Sigma_{cls}}{P_L \sigma \xrightarrow{R, P_L \sigma, k \cdot comb(P_L, \sigma), 1} P_R \sigma} & 2. & \frac{T_1 \xrightarrow{R, T, r, b} T_2}{T_1 \mid T_3 \xrightarrow{R, T, r, b \cdot binom(T, T_1, T_3)} T_2 \mid T_3} \\ 3. & \frac{T_1 \xrightarrow{R, T, r, b} T_2}{(T_1)^L \mid T_3 \xrightarrow{R, (T_1)^L \mid T_3, r \cdot b, 1} (T_2)^L \mid T_3} & 4. & \frac{T_1 \xrightarrow{R, T, r, b} T_2}{(T_3)^L \mid T_1 \xrightarrow{R, (T_3)^L \mid T_1, r \cdot b, 1} (T_3)^L \mid T_2} \end{aligned}$$

then, the semantics of Stochastic CLS is the least labeled transition relation on terms $\xrightarrow{R, r}$, with $R \in \mathcal{R}$ and $r \in \mathbb{R}$, satisfying

$$\frac{T_1 \xrightarrow{R, T, r, b} T_2}{T_1 \xrightarrow{R, r \cdot b} T_2}$$

Fig. 2. The semantics of Stochastic CLS.

to be applied to $a \mid b \mid b$, then the number of combinations of reactants has to become 3, as there are three possible pairs of one a and one b . In general, let T_1 be the instantiation of the left pattern of a rewrite rule (namely, what represents the reactants of the modeled chemical reaction) and T_2 be a term, usually including T_1 , for which the number of combinations of reactants has been computed with $\prod_{T \in \overline{T_1}} \binom{\mathbf{n}(T_2, T)}{\mathbf{n}(T_1, T)}$ as result. Moreover, let T_3 be the term that has to be composed in parallel with T_2 . We have that the number of combinations of reactants in $T_2 \mid T_3$ should be $\prod_{T \in \overline{T_1}} \binom{\mathbf{n}(T_2, T) + \mathbf{n}(T_3, T)}{\mathbf{n}(T_1, T)}$. This

number can be obtained from the number of combinations in T_2 as follows:

$$\prod_{T \in \overline{T_1}} \binom{\mathbf{n}(T_2, T) + \mathbf{n}(T_3, T)}{\mathbf{n}(T_1, T)} = \prod_{T \in \overline{T_1}} \binom{\mathbf{n}(T_2, T)}{\mathbf{n}(T_1, T)} \cdot \text{binom}(T_1, T_2, T_3).$$

Now, the semantics of Stochastic CLS is defined as a labeled transition system, whose transition relation $\xrightarrow{R, r}$, with R a stochastic rewrite rule and $r \in \mathbb{R}$ a stochastic rate, is derived from an auxiliary transition relation $\xrightarrow{R, T, r, b}$, where R is a stochastic rewrite rule, T is obtained from the instantiation of the left pattern of R (representing reactants), $r \in \mathbb{R}$ is a stochastic rate and $b \in \mathbb{N}$ is the number of combinations of reactants T in the current state of the system. Such an auxiliary transition relation is used to compositionally compute the correct number of combinations of reactants, and consequently the correct stochastic rate of the transition. The total rate of a transition, computed in the main transition relation of the semantics, is given by the product $r \cdot b$.

A stochastic CLS model is composed by a term, representing the initial state of the described system, and a set of stochastic rewrite rules.

4.2 Encoding into sSMSR

Now we give two encoding functions that map Stochastic CLS terms and patterns into sSMSR terms and patterns, respectively. These encoding functions are defined by structural recursion and construct one sSMSR string (or string pattern) for each path from the root to a leaf of the abstract syntax tree of the considered Stochastic CLS term (or pattern). The idea of the encoding is to represent a path in the abstract syntax tree of a Stochastic CLS term (or pattern) as a string composed by $\bar{\lambda}_i$ and λ_i symbols representing applications of the looping operator. For example, the Stochastic CLS term $(a \mid b)^L \mid (c \mid d)$ will be translated into the sSMSR string multiset $\bar{\lambda}_1 \cdot a \mid \bar{\lambda}_1 \cdot b \mid \lambda_1 \cdot c \mid \lambda_1 \cdot d$. We do not use any symbol to represent applications of the parallel composition operator \mid of Stochastic CLS as it is directly translated into union of string multisets (or multiset patterns). The same holds for the sequencing operator \cdot of Stochastic CLS that is directly translated into sSMSR string (or string pattern) concatenation. This technique of constructing strings representing paths in an abstract syntax tree is the same used in [8,10] to define enhanced semantics for the study of causality properties.

For the sake of simplicity, we assume that the alphabet and sets of variables of Stochastic CLS are included in those of sSMSR as follows: $\mathcal{E}_{cls} \subset \mathcal{E}$, $\mathcal{X} \subset \mathcal{V}_{\mathcal{E}}$, $SV \subset \mathcal{V}_{\mathcal{S}}$ and $TV \subset \mathcal{V}_{\mathcal{M}}$. We assume also that $\mathcal{V}_{\mathcal{S}}$ contains a special variable Δ that will be used in the encoding of rewrite rules and, finally, we assume that \mathcal{E} contains a special symbol \bullet that will be used in both the encodings of terms and patterns. In order to encode paths in the abstract

syntax tree of Stochastic CLS terms and patterns we assume that the sSMSR alphabet \mathcal{E} contains two symbols $\bar{\lambda}$ and λ and all the natural numbers \mathbb{N} . The two symbols $\bar{\lambda}$ and λ are used to distinguish between the two operands of a Stochastic CLS containment operator, and the natural numbers are used to distinguish two different applications of such an operator. The two symbols $\bar{\lambda}$ and λ will be always followed by either a natural number or an element variable. To simplify the notation we will write λ_i and λ_x for $\lambda \cdot i$ and $\lambda \cdot x$, respectively, $\bar{\lambda}_i$ and $\bar{\lambda}_x$ for $\bar{\lambda} \cdot i$ and $\bar{\lambda} \cdot x$, respectively.

In the definition of the encoding functions we will use an auxiliary injection function $\triangleright : \mathcal{SP} \times \mathcal{MP} \rightarrow \mathcal{MP}$ that inserts a string pattern SP as a prefix of all the elements of a multiset pattern MP . The same function can be applied also to a string and a string multiset rather than to two patterns. The injection function is represented with infix notation and is recursively defined as follows:

$$\begin{aligned} SP_1 \triangleright SP_2 &= SP_1 \cdot SP_2 \\ SP \triangleright (MP_1 \mid MP_2) &= (SP \triangleright MP_1) \mid (SP \triangleright MP_2) \\ SP_1 \triangleright \{\!\!| SP_2 \!\!\}_X &= \{\!\!| SP_1 \cdot SP_2 \!\!\}_X \\ SP_1 \triangleright \{SP_2\} &= \{SP_1 \cdot SP_2\} \end{aligned}$$

Now we define the encoding of Stochastic CLS terms.

Definition 10 (Encoding of terms) *The encoding of Stochastic CLS terms into sSMSR string multisets is given by the function $\lfloor _ \rfloor : \mathcal{T} \rightarrow \mathcal{M}_{\mathcal{S}} \times \wp(\mathbb{N})$ recursively defined as follows:*

$$\begin{aligned} \lfloor S \rfloor &= (\bullet \cdot S, \emptyset) \\ \lfloor T_1 \mid T_2 \rfloor &= (M_1 \mid M_2, I_1 \cup I_2) \quad \text{where } \lfloor T_i \rfloor = (M_i, I_i) \text{ and } I_1 \cap I_2 = \emptyset \\ \lfloor (T_1)^L \rfloor \mid T_2 \rfloor &= (\bar{\lambda}_i \triangleright M_1 \mid \lambda_i \triangleright M_2, I_1 \cup I_2 \cup \{i\}) \\ &\quad \text{where } \lfloor T_j \rfloor = (M_j, I_j), j \in \{1, 2\}, I_1 \cap I_2 = \emptyset \text{ and } i \in \mathbb{N} \setminus (I_1 \cup I_2) \end{aligned}$$

The encoding of terms translates a Stochastic CLS term T into a pair (M, I) where M is the actual result of the translation, namely the sSMSR multiset corresponding to T , and I is the set of natural numbers that occur in M . Notice that a sequence S of Stochastic CLS is encoded into the sSMSR sequence $S' \cdot \bullet \cdot S$ where S' represents the path in the abstract syntax tree, S is its leaf and the symbol \bullet is used as a separator between S' and S . The set I of numbers is used in the definition of the encoding to ensure that different applications of the looping operator in T will be translated into occurrences of $\bar{\lambda}_i$ and λ_i having different indexes. In the following, we will ignore this set of natural numbers and we will use $\lfloor T \rfloor$ to denote only the sSMSR string multiset M .

Now we define the encoding of Stochastic CLS patterns into sSMSR multiset patterns. It is defined only on right patterns \mathcal{P}_R , but, since $\mathcal{P}_L \subset \mathcal{P}_R$, it can be applied also to left patterns \mathcal{P}_L .

Definition 11 (Encoding of patterns) *The encoding of Stochastic CLS right (and left) patterns into sSMSR multiset patterns is given by the function $\llbracket _ \rrbracket : \mathcal{P}_R \rightarrow \mathcal{MP} \times \wp(\mathcal{V}_\mathcal{E}) \times \wp(\mathcal{V}_\mathcal{E})$ recursively defined as follows:*

$$\begin{aligned}
\llbracket SP \rrbracket &= (\bullet \cdot SP, \emptyset, \text{Var}(SP)) \\
\llbracket X \rrbracket &= (\{\bullet\}_X, \emptyset, \mathcal{V}_S(X)) \\
\llbracket P_{R1} \mid P_{R2} \rrbracket &= (MP_1 \mid MP_2, \Gamma_1 \cup \Gamma_2, \Gamma'_1 \cup \Gamma'_2) \\
&\quad \text{where } \llbracket P_{Ri} \rrbracket = (MP_i, \Gamma_i, \Gamma'_i) \\
&\quad \text{and } \Gamma_1 \cap \Gamma_2 = \Gamma'_1 \cap \Gamma'_2 = \Gamma_1 \cap \Gamma'_2 = \emptyset \\
\llbracket (P_{R1})^L \mid P_{R2} \rrbracket &= (\overline{\lambda}_x \triangleright MP_1 \mid \lambda_x \triangleright MP_2, \{x\} \cup \Gamma_1 \cup \Gamma_2, \Gamma'_1 \cup \Gamma'_2) \\
&\quad \text{where } \llbracket P_{Ri} \rrbracket = (MP_i, \Gamma_i, \Gamma'_i) \\
&\quad \text{and } x \in \mathcal{V}_\mathcal{E} \setminus (\Gamma_1 \cup \Gamma'_1 \cup \Gamma_2 \cup \Gamma'_2 \cup \mathcal{X}) \\
&\quad \text{and } \Gamma_1 \cap \Gamma_2 = \Gamma'_1 \cap \Gamma'_2 = \Gamma_1 \cap \Gamma'_2 = \emptyset
\end{aligned}$$

where the auxiliary encoding function $\langle _ \rangle : \mathcal{P}_R \rightarrow \mathcal{MP} \times \wp(\mathcal{V}_\mathcal{E}) \times \wp(\mathcal{V}_\mathcal{E})$ is recursively defined as follows:

$$\begin{aligned}
\langle SP \rangle &= (\{\bullet \cdot SP\}, \emptyset, \text{Var}(SP)) \\
\langle X \rangle &= (\{\bullet\}_X, \emptyset, \mathcal{V}_S(X)) \\
\langle P_{R1} \mid P_{R2} \rangle &= (MP_1 \mid MP_2, \Gamma_1 \cup \Gamma_2, \Gamma'_1 \cup \Gamma'_2) \\
&\quad \text{where } \langle P_{Ri} \rangle = (MP_i, \Gamma_i, \Gamma'_i) \\
&\quad \text{and } \Gamma_1 \cap \Gamma_2 = \Gamma'_1 \cap \Gamma'_2 = \Gamma_1 \cap \Gamma'_2 = \emptyset \\
\langle (P_{R1})^L \mid P_{R2} \rangle &= (\overline{\lambda}_x \triangleright MP_1 \mid \lambda_x \triangleright MP_2, \{x\} \cup \Gamma_1 \cup \Gamma_2, \Gamma'_1 \cup \Gamma'_2) \\
&\quad \text{where } \langle P_{Ri} \rangle = (MP_i, \Gamma_i, \Gamma'_i) \\
&\quad \text{and } x \in \mathcal{V}_\mathcal{E} \setminus (\Gamma_1 \cup \Gamma'_1 \cup \Gamma_2 \cup \Gamma'_2 \cup \mathcal{X}) \\
&\quad \text{and } \Gamma_1 \cap \Gamma_2 = \Gamma'_1 \cap \Gamma'_2 = \Gamma_1 \cap \Gamma'_2 = \emptyset
\end{aligned}$$

The encoding of patterns translates a Stochastic CLS right pattern P_R into a triple (MP, Γ, Γ') , where MP is the actual result of the translation, namely the sSMSR multiset pattern corresponding to P_R , the set Γ contains all the element variables that are used in MP as subscripts of some $\overline{\lambda}$ and λ symbols, and the set Γ' contains all the other variables that may appear in MP . The set Γ is used to ensure that different applications of the looping operator in P_L will be translated into occurrences of symbols $\overline{\lambda}$ and λ having different subscripts. The set Γ' , instead, will be used in the following to translate Stochastic CLS rewrite rules. In what follows, when we do not represent explicitly the triple (MP, Γ, Γ') obtained from the encoding, we will use $\llbracket T \rrbracket$ and $\langle T \rangle$ to denote only the sSMSR multiset pattern MP .

A Stochastic CLS model consisting of an initial term T and a set of stochastic rewrite rules $\mathcal{R} = \{P_{L1} \xrightarrow{k_1} P_{R1}, \dots, P_{Ln} \xrightarrow{k_n} P_{Rn}\}$ can be translated into an sSMSR model consisting of the initial string multiset $\llbracket T \rrbracket$ and a set of stochastic rewrite rules that are the translations of the rules in \mathcal{R} . The translation of

a stochastic rewrite rule $P_{Li} \xrightarrow{k_i} P_{Ri}$ of the Stochastic CLS model is the sSMSR stochastic rewrite rule

$$\Delta \triangleright MP_1 \xrightarrow{k} \Delta \triangleright MP_2$$

where $\llbracket P_{Li} \rrbracket = (MP_1, \Gamma_1, \Gamma'_1)$, $\llbracket P_{Ri} \rrbracket = (MP_2, \Gamma_2, \Gamma'_2)$, $\Gamma_1 \cap \Gamma_2 = \Gamma'_1 \cap \Gamma'_2 = \Gamma_1 \cap \Gamma'_2 = \emptyset$ and $\Delta \notin \Gamma_1 \cup \Gamma'_1 \cup \Gamma_2 \cup \Gamma'_2$.

We now give an example of translation of a Stochastic CLS model into sSMRS.

Example 12 Let $T = b \cdot c \mid (a)^L \mid (b \mid b)$ and $\mathcal{R} = \{R_1, R_2\}$ where

$$R_1 : b \cdot \tilde{x} \mid b \cdot \tilde{y} \xrightarrow{k_1} c \quad R_2 : (a)^L \mid (b \mid X) \xrightarrow{k_2} b \cdot b$$

be the initial term and the set of stochastic rewrite rules of a Stochastic CLS model. Two possible evolutions of the model in accordance with the semantics of Stochastic CLS are

$$T \xrightarrow{R_1, k_1} b \cdot c \mid (a)^L \mid c \quad \text{and} \quad T \xrightarrow{R_2, k_2 \cdot 2} b \cdot c \mid b \cdot b \xrightarrow{R_1, k_1} c.$$

The translation of the considered Stochastic CLS model is an sSMSR model whose initial string multiset is $\llbracket T \rrbracket = \bullet \cdot b \cdot c \mid \bar{\lambda}_1 \cdot \bullet \cdot a \mid \lambda_1 \cdot \bullet \cdot b \mid \lambda_1 \cdot \bullet \cdot b$ and whose set of stochastic rewrite rules is $\mathcal{R}_{\llbracket \cdot \rrbracket} = \{R_1^{\llbracket \cdot \rrbracket}, R_2^{\llbracket \cdot \rrbracket}\}$, where $R_1^{\llbracket \cdot \rrbracket}$ and $R_2^{\llbracket \cdot \rrbracket}$ are the translations of R_1 and R_2 , respectively.

$$R_1^{\llbracket \cdot \rrbracket} : \Delta \cdot \bullet \cdot b \cdot \tilde{x} \mid \Delta \cdot \bullet \cdot b \cdot \tilde{y} \xrightarrow{k_1} \Delta \cdot \bullet \cdot c$$

$$R_2^{\llbracket \cdot \rrbracket} : \Delta \cdot \bar{\lambda}_x \cdot \bullet \cdot a \mid \{\Delta \cdot \lambda_x \cdot \bullet \cdot b\} \mid \{\Delta \cdot \lambda_x \cdot \bullet\}_X \xrightarrow{k_2} \Delta \cdot \bullet \cdot b \cdot b.$$

The evolutions of the sSMSR model corresponding to the above shown evolutions of the Stochastic CLS model are the following:

$$\llbracket T \rrbracket \xrightarrow{R_1^{\llbracket \cdot \rrbracket}, \lambda_1 \cdot \bullet \cdot b \mid \lambda_1 \cdot \bullet \cdot b, k_1} \bullet \cdot b \cdot c \mid \bar{\lambda}_1 \cdot \bullet \cdot a \mid \lambda_1 \cdot \bullet \cdot c$$

$$\llbracket T \rrbracket \xrightarrow{R_2^{\llbracket \cdot \rrbracket}, \bar{\lambda}_1 \cdot \bullet \cdot a \mid \lambda_1 \cdot \bullet \cdot b \mid \lambda_1 \cdot \bullet \cdot b, k_2 \cdot 2} \bullet \cdot b \cdot c \mid \bullet \cdot b \cdot b \xrightarrow{R_1^{\llbracket \cdot \rrbracket}, \bullet \cdot b \cdot c \mid \bullet \cdot b \cdot b, k_1} \bullet \cdot c$$

Now we give a theorem stating the correctness of the encoding of Stochastic CLS into sSMSR. We cannot prove that each transition of the semantics of a Stochastic CLS model has a corresponding transition in the semantics of sSMSR model obtained by the encoding. The cause of this is that different occurrences of the same term $(T_1)^L \mid T_2$ in a Stochastic CLS term are translated into sSMSR string multisets that are different because of the introduction of unique indexes performed by the encoding. This means that, even if the application of a Stochastic CLS rule to any of the occurrences of $(T_1)^L \mid T_2$ may produce the same result, the application of the corresponding rule in its encoding into sSMSR may produce results that differ in some indexes. However, if

we sum up the rates of all the sSMSR transitions corresponding to a Stochastic CLS transition, we obtain the rate of such a Stochastic CLS transition. In order to clarify this point, we give the following example.

Example 13 Let the Stochastic CLS term T be $(a)^L \rfloor (b \mid b) \mid (a)^L \rfloor (b \mid b)$ and the Stochastic CLS rule R be $(a)^L \rfloor (b \mid X) \xrightarrow{k} (a)^L \rfloor (c \mid X)$. We have that T can only perform the transition $T \xrightarrow{R, k \cdot 4} (a)^L \rfloor (b \mid b) \mid (a)^L \rfloor (b \mid c)$. The encoding of T is $\llbracket T \rrbracket = \bar{\lambda}_1 \cdot \bullet \cdot a \mid \lambda_1 \cdot \bullet \cdot b \mid \lambda_1 \cdot \bullet \cdot b \mid \bar{\lambda}_2 \cdot \bullet \cdot a \mid \lambda_2 \cdot \bullet \cdot b \mid \lambda_2 \cdot \bullet \cdot b$, and the translation of R is $R^{\llbracket \cdot \rrbracket} = \{\Delta \cdot \bar{\lambda}_x \cdot \bullet \cdot a\} \mid \{\Delta \cdot \lambda_x \cdot \bullet \cdot b\} \mid \{\Delta \cdot \lambda_x \cdot \bullet\} \rfloor_X \xrightarrow{k} \{\Delta \cdot \bar{\lambda}_x \cdot \bullet \cdot a\} \mid \{\Delta \cdot \lambda_x \cdot \bullet \cdot c\} \mid \{\Delta \cdot \lambda_x \cdot \bullet\} \rfloor_X$. Now, by the semantics of sSMSR, we have that $\llbracket T \rrbracket$ can perform two transitions, namely $\llbracket T \rrbracket \xrightarrow{R^{\llbracket \cdot \rrbracket}, \bar{\lambda}_1 \cdot \bullet \cdot a \mid \lambda_1 \cdot \bullet \cdot b \mid \lambda_1 \cdot \bullet \cdot b, k \cdot 2} \bar{\lambda}_1 \cdot \bullet \cdot a \mid \lambda_1 \cdot \bullet \cdot b \mid \lambda_1 \cdot \bullet \cdot c \mid \bar{\lambda}_2 \cdot \bullet \cdot a \mid \lambda_2 \cdot \bullet \cdot b \mid \lambda_2 \cdot \bullet \cdot b$ and $\llbracket T \rrbracket \xrightarrow{R^{\llbracket \cdot \rrbracket}, \bar{\lambda}_2 \cdot \bullet \cdot a \mid \lambda_2 \cdot \bullet \cdot b \mid \lambda_2 \cdot \bullet \cdot b, k \cdot 2} \bar{\lambda}_1 \cdot \bullet \cdot a \mid \lambda_1 \cdot \bullet \cdot b \mid \lambda_1 \cdot \bullet \cdot b \mid \bar{\lambda}_2 \cdot \bullet \cdot a \mid \lambda_2 \cdot \bullet \cdot b \mid \lambda_2 \cdot \bullet \cdot c$. In both cases the reached state is a possible encoding of the state reached by T and we have that the sum of the rates of the two sSMSR transitions corresponds to the rate of the Stochastic CLS transition.

In order to group all the sSMSR transitions corresponding to an individual Stochastic CLS transition we introduce the following definition.

Definition 14 Given a finite set of stochastic rewrite rules $\mathcal{R} \subseteq \mathfrak{R}$, the labeled transition relation $\xrightarrow{R, r}$, with $R \in \mathcal{R}$ and $r \in \mathbb{R}$, is the least relation on sSMSR string multisets satisfying the following inference rule:

$$T = \left\{ (M'', r) \mid M \xrightarrow{R, M'', r} M'' \wedge \exists f : \mathbb{N} \rightarrow \mathbb{N}. M'' = M' \{f(i_1)/i_1\} \dots \{f(i_n)/i_n\} \right\} /_{\equiv}$$

$$M \xrightarrow{R, r'} M'$$

where $\{\cdot\} /_{\equiv}$ denotes the set containing one only (M'', r) for the equivalence class of \equiv in \mathcal{M} represented by M'' .

Now, the correctness theorem can be formulated as follows. The proof is in Appendix A.

Theorem 15 Given a finite set of Stochastic CLS rewrite rules \mathcal{R} and $T, T' \in \mathcal{T}$, it holds

$$T \xrightarrow{R, r} T' \iff \llbracket T \rrbracket \xrightarrow{R^{\llbracket \cdot \rrbracket}, r} \llbracket T' \rrbracket$$

where $R^{\llbracket \cdot \rrbracket}$ is the translation of R into sSMSR.

Let $Ch = \{x, y, z, n, m, \dots\}$ be the infinite set of all the SPi channels, *processes* of SPi are defined by the following syntax

$$\begin{aligned} P, Q &::= \nu x P \mid P \mid Q \mid \Sigma \mid !\pi.P \\ \Sigma &::= \mathbf{0} \mid \pi.P + \Sigma \\ \pi &::= x(n) \mid x\langle m \rangle \end{aligned}$$

and we denote with \mathcal{P} the set of all SPi processes.

Processes of SPi are equipped with a *structural congruence relation* \equiv which is the least congruence relation on processes satisfying the following rules:

$$\begin{aligned} P \mid \mathbf{0} &\equiv P & P \mid Q &\equiv Q \mid P & P \mid (Q \mid R) &\equiv (P \mid Q) \mid R \\ !\pi.P &\equiv \pi.(P \mid !\pi.P) & \nu x \mathbf{0} &\equiv \mathbf{0} & \nu x \nu y P &\equiv \nu y \nu x P \\ x \notin fn(P) &\Rightarrow \nu x(P \mid Q) &\equiv P \mid \nu x Q \\ \Sigma &\equiv \Sigma' \Rightarrow \pi.P + \Sigma &\equiv \pi.P + \Sigma' & \pi.P + \pi'.P' + \Sigma &\equiv \pi'.P' + \pi.P + \Sigma \end{aligned}$$

The semantics of SPi is the least labeled transition relation \xrightarrow{r} with $r \in \mathbb{R}$, closed with respect to \equiv and satisfying the following inference rules:

$$\begin{aligned} P &\xrightarrow{r} P' \Rightarrow \nu x P \xrightarrow{r} \nu x P' \\ P &\xrightarrow{r} P' \Rightarrow Q \mid P \xrightarrow{r} Q \mid P' \\ x\langle n \rangle.P + \Sigma \mid x\langle m \rangle.Q + \Sigma' &\xrightarrow{rate(x)} P \mid Q_{\{n/m\}} \end{aligned}$$

Fig. 3. The syntax and the semantics of SPi.

5 Encoding the Stochastic π -calculus into sSMSR

In this section we recall the definition of the Stochastic π -calculus (SPi) as given in [17] and define its encoding into sSMSR.

5.1 Definition of SPi

SPi is a stochastic extension of the π -calculus [14] used as the input language of the simulator of biological systems SPiM [21]. The syntax and the semantics of SPi are recalled in Figure 3. Processes are based on input and output actions on channels, denoted $x(n)$ and $x\langle m \rangle$, respectively. In the syntax of processes, $\nu x P$ is the restriction of channel x in process P , $P \mid Q$ is the parallel composition of processes, Σ is the summation of actions, $!\pi.P$ is the replication of process $\pi.P$ and $\mathbf{0}$ is the null process.

SPi processes represent biological entities, and a communication on a channel in SPi represents a chemical reaction. For this reason every channel x in SPi is associated with a reaction rate denoted $rate(x)$, and every transition in

the semantics of SPi is labeled with the rate of the channel on which the communication has been performed. Such a semantics does not compute the number of processes that can communicate on a given channel and bring to the same destination process. Such a number of processes corresponds to the number of combinations of reactants of the reactions modeled by the communications and, according to standard chemical kinetics, it is necessary to compute the actual rate of the reactions. For instance, given processes $P_1 \equiv x\langle n \rangle.P + x\langle n \rangle.P \mid x\langle m \rangle.Q$ and $P_2 \equiv x\langle n \rangle.P \mid x\langle m \rangle.Q$, both P_1 and P_2 can reduce to the same process $P \mid Q_{\{n/m\}}$ with reduction $\xrightarrow{\text{rate}(x)}$, but the reduction should be two times faster in process P_1 than in process P_2 .

In [17] the solution of this problem is the definition of a notion of *channel activity* of a process P on a channel x , $Act_x(P)$. Such a notion is used in the implementation of the SPiM simulator to stochastically select the next reaction channel. Formally $Act_x(P) = (In_x(P) \cdot Out_x(P)) - Mix_x(P)$ where $In_x(P)$ and $Out_x(P)$ are the enabled inputs and outputs on channel x in P , respectively, and $Mix_x(P)$ are the enabled combinations of inputs and outputs on x that belong to the same summation (hence, that cannot interact with each other).

5.2 Encoding into sSMSR

SPi processes that can be used as input of the simulator are usually closed, namely all their channels are restricted. As a consequence, for the sake of simplicity in the encoding we assume processes to be closed. Without loss of generality we assume that all the replications have the form $!\pi.(\nu x P)$, where x does not occur free in P and does not occur in π . Moreover, we assume that all the restricted channels have different names. These assumptions will ensure that each process is encoded as a different sSMSR string. In the following, we call process *component* either an action π or the process $\mathbf{0}$. In particular, in SPi processes $\mathbf{0}$, $\pi.P$ and $!\pi'.P'$ we call $\mathbf{0}$, π and π' *top-level components* and those in P and P' *inner-level components*.

In order to define the encoding of a SPi process into an sSMSR model we assume an infinite set of identifiers $\mathcal{A} = \{A, B, C, \dots\}$. We assume \mathcal{A} and \mathbb{N} to be contained in the sSMSR alphabet \mathcal{E} . As regards sSMSR variables we assume that for each channel $c \in Ch$ there exists an element variable $c \in \mathcal{V}_{\mathcal{E}}$. We will not use any sequence or multiset variables in the encoding of SPi.

The encoding of a SPi process P will consist of two steps. Initially, we will construct from P a set of *process descriptions*, that are pairs (SP, IP) where SP is an sSMSR sequence pattern and IP is an *intermediate process description* whose syntax will be defined in the following. Subsequently, we will translate

the set of process descriptions into a set of sSMSR stochastic rewrite rules R_P . The process P will be translated also into a string multiset M_P containing instantiations of the string patterns occurring in the constructed set of process descriptions. The sSMSR model $\langle M_P, R_P \rangle$ will be the result of the translation of P into sSMSR.

We define intermediate process descriptions and process descriptions as follows.

Definition 16 (Intermediate Process Descriptions) Intermediate process descriptions IP are defined by the following grammar:

$$\begin{aligned} \pi & ::= x(m) \mid x\langle m \rangle \\ IP & ::= \pi.MP \mid !\pi.MP \mid IP + IP \mid \mathbf{0} \end{aligned}$$

where $x, m \in Ch$ and $MP \in \mathcal{MP}$. We denote with \mathcal{IP} the set of all the intermediate process descriptions.

Definition 17 (Process Descriptions) A process description is a pair (SP, IP) where $SP \in \mathcal{SP}$ and $IP \in \mathcal{IP}$. We denote with \mathcal{D} the set of all process descriptions.

An intermediate process description is a (possibly empty) summation of (possibly replicated) actions followed by an sSMSR multiset pattern. A process description is the association of a string pattern with an intermediate process description.

We now define a recursive encoding function \mathcal{I} that gives the process descriptions of a process.

Definition 18 (Process Description Encoding) The recursive encoding func-

tion $\mathcal{I} : \mathcal{P} \times \mathcal{SP} \mapsto \wp(\mathcal{D}) \times \wp(\mathcal{D}) \times \wp(\mathcal{A})$ is defined as follows:

$$\begin{aligned}
\mathcal{I}(\mathbf{0}, SP) &= (\emptyset, \{(ID, \mathbf{0})\}, \{ID\}) \quad \text{where } ID \in \mathcal{A} \\
\mathcal{I}(\nu x P, SP) &= \mathcal{I}(P, SP \cdot x) \\
\mathcal{I}(\pi.P + \Sigma, SP) &= (D \cup D_1 \cup D', \{(ID \cdot SP, \pi.(SP^1 \mid \dots \mid SP^n)) + IP'_1\}, E \cup E_1 \cup \{ID\}) \\
&\quad \text{where } (D, D', E) = \begin{cases} \mathcal{I}(P, SP), & \text{if } \pi \equiv x\langle v \rangle \\ \mathcal{I}(P, SP \cdot v), & \text{if } \pi \equiv x(v) \end{cases} \\
&\quad \text{and } D' = \{(SP^1, IP^1), \dots, (SP^n, IP^n)\} \\
&\quad \text{and } \mathcal{I}(\Sigma, SP) = (D_1, D'_1, E_1) \\
&\quad \text{and } D'_1 = \{(SP'_1, IP'_1)\} \\
&\quad \text{and } ID \in \mathcal{A} \setminus (E \cup E_1) \\
\mathcal{I}(!\pi.P, SP) &= (D \cup D', \{(ID \cdot SP, !\pi.(SP^1 \mid \dots \mid SP^n))\}, E \cup \{ID\}) \\
&\quad \text{where } (D, D', E) = \begin{cases} \mathcal{I}(P, SP), & \text{if } \pi \equiv x\langle v \rangle \\ \mathcal{I}(P, SP \cdot v), & \text{if } \pi \equiv x(v) \end{cases} \\
&\quad \text{and } D' = \{(SP^1, IP^1), \dots, (SP^n, IP^n)\} \\
&\quad \text{and } ID \in \mathcal{A} \setminus E \\
\mathcal{I}(P_1 \mid P_2, SP) &= (D_1 \cup D_2, D'_1 \cup D'_2, E_1 \cup E_2) \\
&\quad \text{where } \mathcal{I}(P_i, SP) = (D_i, D'_i, E_i) \text{ and } E_1 \cap E_2 = \emptyset
\end{aligned}$$

The encoding function \mathcal{I} takes a process P and a string pattern SP and gives a triple (D, D', E) , where D' is the set of descriptions of the top-level components of P , D is the set of descriptions of the inner-level components of P , and E is the set of identifiers used to build D and D' . The sequence pattern SP is used to keep a trace, in the descriptions in D and D' , of both the restricted channels and the channels m for any input action $x(m)$.

Notice that when the process P is the parallel composition $P_1 \mid P_2$, the set of descriptions of the top-level components D' is the union of the descriptions of the top-level components of P_1 and P_2 . When the process P is a summation Σ , D' is given by the summation of the intermediate process descriptions of the top-level components of the summands. The process descriptions of a closed SPi process P are $D \cup D'$ where $\mathcal{I}(P, \epsilon) = (D, D', E)$.

We give an example to show the encoding technique. Let $P = \nu x \nu y (x\langle y \rangle.\mathbf{0} \mid x(z).\mathbf{0})$ be the process which can communicate on channel x yielding the process $\mathbf{0}$. Process P is composed by four components, namely $x\langle y \rangle$, $\mathbf{0}$, $x(z)$ and $\mathbf{0}$ where $x\langle y \rangle$ and $x(z)$ are its top-level components. By definition of \mathcal{I} we obtain for P the sets of process descriptions $\{(A \cdot x \cdot y, x\langle y \rangle.B \cdot x \cdot y), (C \cdot x \cdot y, x(z).D \cdot x \cdot y \cdot z)\}$ and $\{(B \cdot x \cdot y, \mathbf{0}), (D \cdot x \cdot y, \mathbf{0})\}$. Notice that an intermediate process description of P can be composed by a SPi action π , the top-level component, followed by a union of sequence patterns. Each of these represents the description, obtained by the encoding, of one of the top-level components in the continuation of π .

We denote with Act the set of all the possible SPi actions, namely $Act = \{x(y) \mid x, y \in Ch\} \cup \{x\langle y \rangle \mid x, y \in Ch\}$, and with π a generic action of Act . We now define an auxiliary function $\eta : \mathcal{D} \times Act \mapsto \wp(MP)$ such that $\eta((SP, MP), \pi)$ computes the set of all multiset patterns that appear into MP as continuation of any action identified by π . For instance $\eta((SP, x\langle v \rangle.MP + x(v).MP' + x\langle v \rangle.MP''), x\langle v \rangle) = \{MP, MP''\}$ because the process described by SP can execute the action $x\langle v \rangle$ with both the continuations MP and MP'' . The function η is defined as follows:

$$\begin{aligned} \eta((SP, \mathbf{0}), \pi) &= \emptyset \\ \eta((SP, \pi'.P), \pi) &= \emptyset \quad \text{if } \pi' \neq \pi \\ \eta((SP, \pi.(SP_1 \mid \dots \mid SP_n)), \pi) &= \cup_{i=1}^n \{SP_i\} \\ \eta((SP, !\pi.(SP_1 \mid \dots \mid SP_n)), \pi) &= \cup_{i=1}^n \{SP_i\} \\ \eta((SP, \pi.(SP_1 \mid \dots \mid SP_n) + IP), \pi) &= \cup_{i=1}^n \{SP_i\} \cup \eta((SP, IP), \pi) \end{aligned}$$

Note that η is defined on all cases of the intermediate process description in its first argument. Recall that an intermediate process description is, with respect to the definition of the function \mathcal{I} , either the process $\mathbf{0}$, or an action (possibly prefixed by the replication operator) followed by a union of sequence patterns, or a summation of intermediate process descriptions. We can now define the encoding of a closed SPi process P as an sSMSR model $\langle M_P, R_P \rangle$.

Definition 19 (Process Encoding) *Given a SPi process P , let $\mathcal{I}(P, \epsilon) = (D, D', E)$ be the process description encoding of P . We define an sSMSR model $\langle M_P, R_P \rangle$ as the encoding of P , where the term M_P and the rules R_P are computed as follows:*

- let $D' = \{(SP_1, IP_1), \dots, (SP_n, IP_n)\}$ and let $\gamma : Ch \mapsto \mathbb{N}$ be an injective function. The sSMSR string multiset M_P is defined as

$$M_P \equiv SP_{1\{\gamma(c)/c\}} \mid \dots \mid SP_{n\{\gamma(c)/c\}}$$

where $\{\gamma(c)/c\}$ denotes the substitution of all channels $c \in Ch$ with $\gamma(c)$;

- the set of sSMSR stochastic rewrite rules R_P is defined as $R_P = \bigcup_{x \in Ch} R_P^x$ where R_P^x denotes the set of rules that model a communication over channel x , namely

$$\begin{aligned} R_P^x &= \{SP_1 \mid SP_2 \xrightarrow{r} SP'_1 \mid SP'_2 \mid P_1 \mid P_2\{y/z\} \text{ such that} \\ &\quad \forall D_1 \in D \cup D'. \forall y, z \in Ch. \eta(D_1, x\langle y \rangle) \neq \emptyset. \\ &\quad \forall D_2 \in D \cup D'. \eta(D_2, x(z)) \neq \emptyset. \\ &\quad \forall (P_1, P_2) \in \eta(D_1, x\langle y \rangle) \times \eta(D_2, x(z)). \\ &\quad \text{where } D_i = (SP_i, IP_i) \text{ and } r = \text{rate}(x) \\ &\quad \text{and } SP'_i = \begin{cases} SP_i, & \text{if } IP_i = !x(z).P_i \text{ or } !x\langle y \rangle.P_i \\ \epsilon, & \text{otherwise} \end{cases} \\ &\quad \} \end{aligned}$$

The process encoding creates a rule for each possible pair of process descriptions containing an input and an output on the same channel. This means that it creates a rule also for the two actions of a process $x\langle m \rangle.x\langle n \rangle.\mathbf{0}$ even if they cannot interact with each other. The semantics will ensure that these rules will never be applied. Moreover, the rules created by composing some replicated action $!\pi.P$ contain some non empty patterns SP'_1 or SP'_2 in its right hand side. These patterns reintroduce, in accordance with the SPi semantics, the string pattern of the process description containing the replication. Finally, the substitution $\{y/z\}$ corresponds to the substitution that is performed in the SPi semantics when a communication occurs.

We remark that, since in SPi there is no notion of membrane, in the encoding of process we do not make any use of the matching operators of sSMSR. We now give the following proposition: given the encoding of a process P , for any pair of channels, the corresponding set of rules obtained by the encoding are pairwise disjoint.

Proposition 20 $\forall P \in \mathcal{P}. \forall x, y \in Ch. x \neq y \Rightarrow R_P^x \cap R_P^y = \emptyset.$

As an example, we show now the encoding and some steps of computation of a SPi process P , built by using channels $Ch = \{x, z, w, v, k, y\}$, such that

$$P \equiv \nu x \nu z \nu w (!x\langle v \rangle.\nu k\langle \nu y \rangle(y\langle v \rangle.\mathbf{0} \mid y\langle v \rangle.\mathbf{0})) \mid x\langle z \rangle.\mathbf{0} + x\langle w \rangle.\mathbf{0}$$

The process P can communicate on channel x the value z or w depending on the chosen action, namely $x\langle z \rangle$ or $x\langle w \rangle$. After communicating, the process replicates its left side, denoted as $!x\langle v \rangle.P'$ with $P' \equiv \nu k \nu y (y\langle v \rangle.\mathbf{0} \mid y\langle v \rangle.\mathbf{0})$, and generates a new process P' restricted on channels k and y . The restriction on channel k appears in P' by the assumptions we made on the SPi processes that can be encoded into sSMSR. Process P' can communicate on channel y the value v which has been bound, by the communication on channel x , to value z or w .

With respect to the SPi semantics the behavior of P is described by the following transitions:

$$\begin{aligned} P &\equiv \nu x \nu z \nu w (!x\langle v \rangle.P' \mid x\langle z \rangle.\mathbf{0} + x\langle w \rangle.\mathbf{0}) \equiv \\ &\nu x \nu z \nu w (x\langle v \rangle.(!x\langle v \rangle.P' \mid P') \mid x\langle z \rangle.\mathbf{0} + x\langle w \rangle.\mathbf{0}) \\ &\xrightarrow{\text{rate}(x)} \nu x \nu z \nu w (!x\langle v \rangle.P' \mid P'_{\{z/v\}}) \equiv \\ &\nu x \nu z \nu w (!x\langle v \rangle.P' \mid \nu k \nu y (y\langle z \rangle.\mathbf{0} \mid y\langle z \rangle.\mathbf{0})) \\ &\xrightarrow{\text{rate}(y)} \nu x \nu z \nu w (!x\langle v \rangle.P' \mid \mathbf{0}) \equiv \\ &\nu x \nu z \nu w (!x\langle v \rangle.(!x\langle v \rangle.P' \mid P')) \end{aligned}$$

The sSMSR model emulating P is obtained by computing the process de-

scription of P , namely $\mathcal{I}(P, \epsilon) = (D, D', E)$, where

$$\begin{aligned} D &= \{(N_1, \mathbf{0}), (N_2, \mathbf{0}), (N_3, \mathbf{0}), (N_4, \mathbf{0}), \\ &\quad (B \cdot x \cdot z \cdot w \cdot v \cdot k \cdot y, y(v).N_1), (C \cdot x \cdot z \cdot w \cdot v \cdot k \cdot y, y(v).N_2)\} \\ D' &= \{(A \cdot x \cdot z \cdot w, x(z).N_3 + x(w).N_4), \\ &\quad (D \cdot x \cdot z \cdot w, !x(v).(B \cdot x \cdot z \cdot w \cdot v \cdot k \cdot y \mid C \cdot x \cdot z \cdot w \cdot v \cdot k \cdot y))\} \\ E &= \{A, B, C, D, N\} \end{aligned}$$

Notice that, due to the function \mathcal{I} , there exist four different descriptions for the component $\mathbf{0}$ which are identified by the sequence patterns N_1 , N_2 , N_3 and N_4 .

If we assume a function γ such that $\{(x, 1), (z, 2), (w, 3)\} \subset \gamma$, the sSMSR term M which represents the encoding of P is obtained by computing

$$\begin{aligned} M &\equiv A \cdot x \cdot z \cdot w_{\{\gamma(c)/c\}} \mid D \cdot x \cdot z \cdot w_{\{\gamma(c)/c\}} \\ &\equiv A \cdot 1 \cdot 2 \cdot 3 \mid D \cdot 1 \cdot 2 \cdot 3 \end{aligned}$$

The set of sSMSR stochastic rewriting rules obtained by the encoding are the following:

- (1) $A \cdot x \cdot z \cdot w \mid D \cdot x \cdot z \cdot w \xrightarrow{\text{rate}(x)} D \cdot x \cdot z \cdot w \mid B \cdot x \cdot z \cdot w \cdot z \cdot k \cdot y \mid C \cdot x \cdot z \cdot w \cdot z \cdot k \cdot y \mid N_3$
- (2) $A \cdot x \cdot z \cdot w \mid D \cdot x \cdot z \cdot w \xrightarrow{\text{rate}(x)} D \cdot x \cdot z \cdot w \mid B \cdot x \cdot z \cdot w \cdot w \cdot k \cdot y \mid C \cdot x \cdot z \cdot w \cdot w \cdot k \cdot y \mid N_4$
- (3) $B \cdot x \cdot z \cdot w \cdot v \cdot k \cdot y \mid C \cdot x \cdot z \cdot w \cdot v \cdot k \cdot y \xrightarrow{\text{rate}(y)} N_1 \mid N_2$

where rule (1) describes the communication on x of value z , rule (2) the communication on x of value w , and rule (3) the communication on y of value v . Notice that the free variables k and y in rule (1) and (2) will be instantiated, with respect to the semantics of sSMSR, with fresh symbols. In particular, the assumed restriction on channel k provides the fact that each copy of this process will have a different value $\sigma(k)$ for the used instantiation function σ used in the application of the rule. This allows the exact number of combinations of reactants (corresponding to the activity of SPi channels) to be taken into account. Analogously, the generation of a fresh value for the channel y reflects the fact that the channel is restricted in process P .

The sSMSR computation corresponding to the shown SPi computation is

$$\begin{aligned} M &\equiv A \cdot 1 \cdot 2 \cdot 3 \mid D \cdot 1 \cdot 2 \cdot 3 \\ &\xrightarrow{1, A \cdot 1 \cdot 2 \cdot 3 \mid D \cdot 1 \cdot 2 \cdot 3, \text{rate}(x)} D \cdot 1 \cdot 2 \cdot 3 \mid B \cdot 1 \cdot 2 \cdot 3 \cdot 2 \cdot 5 \cdot 6 \mid C \cdot 1 \cdot 2 \cdot 3 \cdot 2 \cdot 5 \cdot 6 \mid N_3 \\ &\xrightarrow{3, B \cdot 1 \cdot 2 \cdot 3 \cdot 2 \cdot 5 \cdot 6 \mid C \cdot 1 \cdot 2 \cdot 3 \cdot 2 \cdot 5 \cdot 6, \text{rate}(y)} D \cdot 1 \cdot 2 \cdot 3 \mid N_3 \mid N_1 \mid N_2 \end{aligned}$$

Notice that the first communication, namely the passing of the value z on channel x , is modeled by building two processes where the value of variable v

has been substituted by the value $\sigma(z)$, namely 2. Furthermore, as the built processes share the restricted channels k and y , the values $\sigma(k)$ and $\sigma(y)$ are fresh values for the sSMSR term, namely 5 and 6. Such a behavior is correct because, for any pair of processes identified by B and C that could be created by multiple instances of processes identified by A and D , channels k and y are local and, consequently, distinguishable.

We give now some theorems stating the soundness and the completeness of the encoding of SPi into sSMSR and provide the relationship between the labels of the transitions of the two semantics. We start by showing the soundness and completeness of the encoding. The proof is given in Appendix B.

Theorem 21 *Given a SPi process P and its sSMSR encoding $\langle M_P, R_P \rangle$, it holds*

$$P \xrightarrow{r} P' \iff M_P \xrightarrow{R, M, v} M_{P'}.$$

In order to define the relationship between the labels on the transitions of the semantics of SPi and sSMSR we introduce some auxiliary notions. Given an sSMSR term M and a set of rules \mathcal{R} we define the *exit rate* of state M as follows:

$$\text{ExitRate}(M, \mathcal{R}) = \sum_{\{v \mid M \xrightarrow{R, M', v} M' \wedge R \in \mathcal{R}\}} v$$

In a state M the exit rate is equal to the sum of all the rates for any possible transitions of the semantics that can be derived in such a state with respect to the set of rules R .

The following theorem states the relationship between the labels on the semantics of SPi and sSMSR. The proof is given in Appendix C.

Theorem 22 *Given a SPi process P encoded into the sSMSR model $\langle M_P, R_P \rangle$, for any $x \in Ch$ it holds: $\text{rate}(x) \cdot \text{Act}_x(P) = \text{ExitRate}(M_P, R_P^x)$.*

Corollary 23 $\sum_{x \in Ch} \text{rate}(x) \cdot \text{Act}_x(P) = \text{ExitRate}(M_P, R_P)$.

6 Related works and conclusions

We have proposed Stochastic String MultiSet Rewriting (sSMSR) as an intermediate language for the simulation of biomolecular systems. sSMSR is an extension of multiset rewriting with strings as multiset elements and richer rewrite rules. Higher level formalisms for biological systems descriptions can be translated into sSMSR and efficient simulators for sSMSR can be developed. We have defined the encodings of Stochastic CLS and of the Stochastic

π -calculus into sSMSR, and we have proved soundness and completeness of both the encodings.

In [22] the formalism $\pi@$ is presented. It is a calculus designed to be a core language for analysing formalisms which model localisation and compartmentalisation. As example BioAmbients and Brane Calculi, two formalisms belonging to the class of process calculi, are encoded into $\pi@$. Furthermore, in [23] the encoding of catalytic P Systems into $\pi@$ is given. An implementation of a stochastic version of $\pi@$ as an extension of the SPiM simulator has been planned. With respect to our proposal we notice that $\pi@$, as defined in [22], is not stochastic and that the encoding of term rewriting systems such as Stochastic CLS does not seem to be easy.

Other variants of multiset rewriting that we have considered before defining sSMSR are the first order multiset rewriting [7] and Gamma [1]. The former is multiset rewriting enriched with the possibility of creating fresh symbols, and the latter extends multiset rewriting with side conditions in rewrite rules. Even if these features make the formalisms Turing-complete, we believe that they are not sufficient to make the encoding of other formalisms easy enough. In fact, as regards both the formalisms, multiset elements may be structured, but the lack of operators on the structure of elements makes the description of changes in structure of the modeled biological system quite difficult.

As future work we plan to develop a stochastic simulator based on sSMSR and to develop analysis and verification techniques for this language. These techniques could be used, via translation into sSMSR, to study properties of systems described by higher level formalisms.

Acknowledgments

This research has been partially supported by MiUR PRIN 2006 Project “Biologically Inspired Systems and Calculi and their Applications (BISCA)”.

References

- [1] J.P. Banatre, P. Fradet, D. Le Metayer, Gamma and the Chemical Reaction Model: Fifteen Years After, in: Proceedings of the Workshop on Multiset Processing, WMP 2000, in: Lecture Notes in Computer Science, vol. 2235, Springer, 2001, pp. 17–44.
- [2] R. Barbuti, G. Caravagna, A. Maggiolo-Schettini, P. Milazzo, An Intermediate Language for the Simulation of Biological Systems, in: Proceedings of the 1th

- International Workshop From Biology To Concurrency and Back, FBTC07, in: *Electronic Notes in Theoretical Computer Science* 194 (2008), 19–34.
- [3] R. Barbuti, G. Caravagna, A. Maggiolo-Schettini, P. Milazzo, G. Pardini, The Calculus of Looping Sequences, in: M. Bernardo, P. Degano and G. Zavattaro (Eds.), *Formal Methods for Computational Systems Biology*, *Lecture Notes in Computer Science*, vol. 5016, Springer, 2008, 387–423.
 - [4] R. Barbuti, A. Maggiolo-Schettini, P. Milazzo, A. Troina, A Calculus of Looping Sequences for Modelling Microbiological Systems, in: *Fundamenta Informaticae* 72 (2006), 21–35.
 - [5] R. Barbuti, A. Maggiolo-Schettini, P. Milazzo, A. Troina, The Calculus of Looping Sequences for Modeling Biological Membranes, in: *Proceedings of the 8th Workshop on Membrane Computing, WMC8*, in: *Lecture Notes in Computer Science*, vol. 4860, Springer, 2007, pp. 54–76.
 - [6] L. Cardelli, Brane Calculi. Interactions of Biological Membranes, in: *Proceedings of the International Conference on Computational Methods in Systems Biology, CMSB 04*, in: *Lecture Notes in Computer Science*, vol. 3082, Springer, 2005, pp. 257–280.
 - [7] I. Cervesato, The Logical Meeting Point of Multiset Rewriting and Process Algebra, Tech. Rep. CHACS-5540-153, Center for High Assurance Computer Systems, Naval Research Laboratory, 2004.
 - [8] M. Curti, P. Degano, C. Priami, C.T. Baldari, Modelling Biochemical Pathways through Enhanced π -calculus, *Theoret. Comput. Sci.* 325 (2004) 111–140.
 - [9] V. Danos, C. Laneve, Formal Molecular Biology, *Theoret. Comput. Sci.* 325 (2004), 69–110.
 - [10] P. Degano, C. Priami, Enhanced Operational Semantics: A Tool for Describing and Analyzing Concurrent Systems, *ACM Computing Surveys* 33 (2001), 135–176.
 - [11] D. Gillespie, Exact Stochastic Simulation of Coupled Chemical Reactions, *J. of Physical Chemistry* 81 (1977), 2340–2361.
 - [12] F. Martinelli, S. Bistarelli, I. Cervesato, G. Lenzini, R. Marangoni, Representing Biological Systems Through Multiset Rewriting, in: *Proceedings of the 9th Workshop on Computer Aided Systems Theory, EUROCAST’03*, in: *Lecture Notes in Computer Science*, vol. 2809, Springer, 2004, pp. 415–426.
 - [13] P. Milazzo, Qualitative and Quantitative Formal Modeling of Biological Systems. Ph.D. Thesis, University of Pisa, Pisa, Italy, 2007.
 - [14] R. Milner, *Communicating and Mobile Systems: the π -Calculus*, Cambridge University Press, 1999.
 - [15] G. Păun, *Membrane Computing. An Introduction*. Springer, 2002.

- [16] The P Systems web page: <http://psystems.disco.unimib.it/>.
- [17] A. Phillips, L. Cardelli, A Correct Abstract Machine for the Stochastic π -calculus, in: Proceedings of the Workshop on Concurrent Models in Molecular Biology, BIO-CONCUR'04.
- [18] C. Priami, A. Regev, W. Silvermann, E. Shapiro, Application of a Stochastic Name-Passing Calculus to Representation and Simulation of Molecular Processes, Information Processing Letters 80 (2001), 25–31.
- [19] A. Regev, E.M. Panina, W. Silverman, L. Cardelli, E.Y. Shapiro, BioAmbients: an abstraction for biological compartments, Theoret. Comput. Sci. 325(1) (2004), 141-167.
- [20] The CLSm web page: <http://www.di.unipi.it/~milazzo/biosims/>.
- [21] The SPiM web page: <http://research.microsoft.com/~aphillip/spim/>.
- [22] C. Versari, A Core Calculus for a Comparative Analysis of Bio-Inspired Calculi, in: Proceedings of the European Symposium on Programming, ESOP'07, in: Lecture Notes in Computer Science, vol. 4421, Springer, 2007, pp. 411–425.
- [23] C. Versari, Encoding catalytic P systems in $\pi@$, in: Proceedings of the Workshop on Membrane Computing and Biologically Inspired Process Calculi, MeCBIC 2006, in: Electronic Notes in Theoretical Computer Science 171(2) (2007), 171–186.

A Proof of Theorem 15

In order to prove Theorem 15 we introduce some auxiliary lemmata and definitions.

Lemma 24 $\langle SP \triangleright MP \rangle_\rho = SP \triangleright \langle MP \rangle_\rho$ and $\diamond(SP \triangleright MP) = SP \triangleright \diamond(MP)$.

PROOF. Trivial structural induction on MP .

Definition 25 (σ -compliance) Let σ be a Stochastic CLS instantiation function. An sSMSR instantiation function $\sigma^{\llbracket \cdot \rrbracket}$ and a pattern expansion parameter function $\rho^{\llbracket \cdot \rrbracket}$ are σ -compliant if and only if they satisfy the following constraints:

$$\sigma^{\llbracket \cdot \rrbracket}(v) = \begin{cases} \sigma(v) & \text{if } v \in \mathcal{X} \cup SV \\ S_i & \text{if } v = \tilde{x}_i \in \mathcal{V}_S(X) \text{ and } \llbracket \sigma(X) \rrbracket = S_1 \mid \dots \mid S_n \end{cases}$$

$$\rho^{\llbracket \cdot \rrbracket}(X) = n \quad \text{if } X \in TV \text{ and } \llbracket \sigma(X) \rrbracket = S_1 \mid \dots \mid S_n$$

Lemma 26 *Given $\sigma \in \Sigma_{cls}$, there exists a unique pattern expansion parameter function $\rho^{[\cdot]}$ that is σ -compliant.*

PROOF. Follows immediately from the definition of σ -compliance.

Lemma 27 *Given $P_R \in \mathcal{P}_R$ and $\sigma \in \Sigma_{cls}$, there exist $\sigma^{[\cdot]}$ and $\rho^{[\cdot]}$ that are a σ -compliant sSMSR instantiation function and a σ -compliant pattern expansion parameter function, respectively, such that $\lfloor P_R \sigma \rfloor \equiv \langle \langle P_R \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]}$.*

PROOF. We first prove by structural induction on P_R that there exist $\sigma^{[\cdot]}$ and $\rho^{[\cdot]}$ such that $\lfloor P_R \sigma \rfloor \equiv \langle \langle P_R \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]}$. By Lemma 26 we have that $\rho^{[\cdot]}$ is known and we have only to show that $\sigma^{[\cdot]}$ exists such that the thesis holds.

• Base cases:

• Let $P_R = SP$;

We prove that $\lfloor SP \sigma \rfloor \equiv \langle \langle SP \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]}$ holds for any σ -compliant $\sigma^{[\cdot]}$. Since $SP\sigma$ is a sequence, $\lfloor SP \sigma \rfloor = SP\sigma$. By definition of $\langle \cdot \rangle$ we have $\langle \langle SP \rangle \rangle = \{SP\}$. By definition of pattern expansion, we have $\langle \{SP\} \rangle_{\rho^{[\cdot]}} = SP$ and since SP contains only variables in $\mathcal{X} \cup SP$ it holds $SP\sigma^{[\cdot]} = SP\sigma$.

• Let $P_R = X$;

We prove that $\lfloor \sigma(X) \rfloor \equiv \langle \langle X \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]}$ holds for any σ -compliant $\sigma^{[\cdot]}$. Let $\lfloor \sigma(X) \rfloor = S_1 \mid \dots \mid S_n$. By definition of $\langle \cdot \rangle$ we have $\langle \langle X \rangle \rangle = \{\epsilon\}_X$. Since $\rho^{[\cdot]}$ is σ -compliant we have $\rho^{[\cdot]}(X) = n$, hence $\langle \{\epsilon\}_X \rangle_{\rho^{[\cdot]}} = \epsilon \cdot \tilde{x}_1 \mid \dots \mid \epsilon \cdot \tilde{x}_n \equiv \tilde{x}_1 \mid \dots \mid \tilde{x}_n$. Since $\sigma^{[\cdot]}$ is σ -compliant we have $\sigma^{[\cdot]}(\tilde{x}_i) = S_i$, hence $(\tilde{x}_1 \mid \dots \mid \tilde{x}_n) \sigma^{[\cdot]} = S_1 \mid \dots \mid S_n$.

• Induction cases:

• Let $P_R = P_{R1} \mid P_{R2}$;

We prove that there exists a σ -compliant $\sigma^{[\cdot]}$ such that $\lfloor (P_{R1} \mid P_{R2}) \sigma \rfloor \equiv \langle \langle P_{R1} \mid P_{R2} \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]}$ holds. It is easy to see that $(P_{R1} \mid P_{R2})\sigma = P_{R1}\sigma \mid P_{R2}\sigma$. By definition of $\lfloor \cdot \rfloor$ we have $\lfloor P_{R1}\sigma \mid P_{R2}\sigma \rfloor = \lfloor P_{R1}\sigma \rfloor \mid \lfloor P_{R2}\sigma \rfloor$. Similarly, by definition of $\langle \cdot \rangle$ we have $\langle \langle P_{R1} \mid P_{R2} \rangle \rangle = \langle \langle P_{R1} \rangle \rangle \mid \langle \langle P_{R2} \rangle \rangle$. Moreover, by definition of pattern expansion we have $\langle \langle \langle P_{R1} \rangle \rangle \mid \langle \langle P_{R2} \rangle \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]} = (\langle \langle P_{R1} \rangle \rangle_{\rho^{[\cdot]}} \mid \langle \langle P_{R2} \rangle \rangle_{\rho^{[\cdot]}}) \sigma^{[\cdot]}$ that is equal to $\langle \langle P_{R1} \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]} \mid \langle \langle P_{R2} \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]}$.

By induction hypothesis we have that $\lfloor P_{Ri}\sigma \rfloor \equiv \langle \langle P_{Ri} \rangle \rangle_{\rho^{[\cdot]}} \sigma_i^{[\cdot]}$ for some σ -compliant $\sigma_1^{[\cdot]}$ and $\sigma_2^{[\cdot]}$. Now, by definition of $\langle \cdot \rangle$ we have that the only variables that may occur both in $\langle \langle P_{R1} \rangle \rangle_{\rho^{[\cdot]}} \sigma_1^{[\cdot]}$ and $\langle \langle P_{R2} \rangle \rangle_{\rho^{[\cdot]}} \sigma_2^{[\cdot]}$ are those occurring in both P_{R1} and P_{R2} , and since both $\sigma_1^{[\cdot]}$ and $\sigma_2^{[\cdot]}$ are σ -compliant, we have that they must agree on the instantiation of those variables. As a consequence, let us consider a function $\sigma^{[\cdot]}$ such that $\sigma^{[\cdot]}(v) = \sigma_i^{[\cdot]}(v)$ for all v occurring in $\langle \langle P_{Ri} \rangle \rangle$. We have that both $\lfloor P_{R1}\sigma \rfloor \equiv \langle \langle P_{R1} \rangle \rangle_{\sigma^{[\cdot]}}$ and $\lfloor P_{R2}\sigma \rfloor \equiv \langle \langle P_{R2} \rangle \rangle_{\sigma^{[\cdot]}}$ hold, and hence $\lfloor P_{R1}\sigma \rfloor \mid \lfloor P_{R2}\sigma \rfloor \equiv \langle \langle P_{R1} \rangle \rangle_{\sigma^{[\cdot]}} \mid \langle \langle P_{R2} \rangle \rangle_{\sigma^{[\cdot]}}$ holds.

• Let $(P_{R1})^L \rfloor P_{R2}$;

We prove that there exists a σ -compliant $\sigma^{[\cdot]}$ such that $\lfloor ((P_{R1})^L \rfloor P_{R2})\sigma \rfloor \equiv \langle \langle (P_{R1})^L \rfloor P_{R2} \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]} \rangle$ holds. It is easy to see that $((P_{R1})^L \rfloor P_{R2})\sigma = (P_{R1}\sigma)^L \rfloor (P_{R2}\sigma)$. By definition of $\lfloor \cdot \rfloor$ we have $\lfloor (P_{R1}\sigma)^L \rfloor (P_{R2}\sigma) \rfloor = \bar{\lambda}_j \triangleright \lfloor P_{R1}\sigma \rfloor \mid \lambda_j \triangleright \lfloor P_{R2}\sigma \rfloor$ for some $j \in \mathbb{R}$ such that j does not occur in $\lfloor P_{R1}\sigma \rfloor$ and $\lfloor P_{R2}\sigma \rfloor$. Similarly, by definition of $\langle \cdot \rangle$ we have $\langle (P_{R1})^L \rfloor (P_{R2}) \rangle = \bar{\lambda}_x \triangleright \langle P_{R1} \rangle \mid \lambda_x \triangleright \langle P_{R2} \rangle$ for some $x \in \mathcal{V}_\mathcal{E} \setminus \mathcal{X}$ such that x does not occur in $\langle P_{R1} \rangle$ and $\langle P_{R2} \rangle$. Now, by definition of pattern expansion we have $\langle \bar{\lambda}_x \triangleright \langle P_{R1} \rangle \mid \lambda_x \triangleright \langle P_{R2} \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]} = \langle \langle \bar{\lambda}_x \triangleright \langle P_{R1} \rangle \rangle_{\rho^{[\cdot]}} \mid \langle \lambda_x \triangleright \langle P_{R2} \rangle \rangle_{\rho^{[\cdot]}} \rangle \sigma^{[\cdot]}$. It is easy to see that this is equal to $\langle \bar{\lambda}_x \triangleright \langle P_{R1} \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]} \mid \langle \lambda_x \triangleright \langle P_{R2} \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]}$, and by Lemma 24, we have that this, in turn, is equal to $(\bar{\lambda}_x \triangleright \langle \langle P_{R1} \rangle \rangle_{\rho^{[\cdot]}}) \sigma^{[\cdot]} \mid (\lambda_x \triangleright \langle \langle P_{R2} \rangle \rangle_{\rho^{[\cdot]}}) \sigma^{[\cdot]}$.

By definition of $\langle \cdot \rangle$ we have that $\langle (P_{R1})^L \rfloor P_{R2} \rangle$ ensures that x does not occur neither in $\langle P_{R1} \rangle$ nor in $\langle P_{R2} \rangle$. If we assume that $\sigma^{[\cdot]}(x) = j$ we obtain that $(\bar{\lambda}_x \triangleright \langle \langle P_{R1} \rangle \rangle_{\rho^{[\cdot]}}) \sigma^{[\cdot]} \mid (\lambda_x \triangleright \langle \langle P_{R2} \rangle \rangle_{\rho^{[\cdot]}}) \sigma^{[\cdot]}$ is equal to $\bar{\lambda}_j \triangleright \langle \langle P_{R1} \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]} \mid \lambda_j \triangleright \langle \langle P_{R2} \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]}$. By the induction hypothesis we have $\lfloor P_{Ri}\sigma \rfloor \equiv \langle \langle P_{Ri} \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]}$ and hence $\bar{\lambda}_j \triangleright \lfloor P_{R1}\sigma \rfloor \mid \lambda_j \triangleright \lfloor P_{R2}\sigma \rfloor \equiv \bar{\lambda}_j \triangleright \langle \langle P_{R1} \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]} \mid \lambda_j \triangleright \langle \langle P_{R2} \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]}$.

The proof that there exist $\sigma^{[\cdot]}$ and $\rho^{[\cdot]}$ which are σ -compliant and such that $\lfloor P_R\sigma \rfloor \equiv \langle \langle P_R \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]}$ holds is similar, but with SP rather than $\{SP\}$ in the first base case and with the use of the just proved truth of $\lfloor P_R\sigma \rfloor \equiv \langle \langle P_R \rangle \rangle_{\rho^{[\cdot]}} \sigma^{[\cdot]}$ rather than the application of the induction hypothesis in the fourth case.

Lemma 28 *It holds*

$$\text{Rate}(k, \diamond(\llbracket P_L \rrbracket)\sigma, \langle \llbracket P_L \rrbracket \rangle_{\rho}\sigma) = k \cdot \prod_{S \in \Omega} \binom{\mathbf{n}(\langle \llbracket P_L \rrbracket \rangle_{\rho}\sigma, S)}{\mathbf{n}(\diamond(\llbracket P_L \rrbracket)\sigma, S)}$$

where $\Omega = \overline{\diamond(\llbracket P_L \rrbracket)\sigma} \cap \overline{\langle \llbracket P_L \rrbracket \rangle_{\rho}\sigma \setminus \diamond(\llbracket P_L \rrbracket)\sigma}$. The same holds with $\llbracket \cdot \rrbracket$ replaced by $\langle \cdot \rangle$.

PROOF.

$$\begin{aligned} \text{Rate}(k, \diamond(\llbracket P_L \rrbracket)\sigma, \langle \llbracket P_L \rrbracket \rangle_{\rho}\sigma) &= k \cdot \prod_{S \in \overline{\diamond(\llbracket P_L \rrbracket)\sigma}} \binom{\mathbf{n}(\langle \llbracket P_L \rrbracket \rangle_{\rho}\sigma, S)}{\mathbf{n}(\diamond(\llbracket P_L \rrbracket)\sigma, S)} \\ &= k \cdot \prod_{S \in \Omega} \binom{\mathbf{n}(\langle \llbracket P_L \rrbracket \rangle_{\rho}\sigma, S)}{\mathbf{n}(\diamond(\llbracket P_L \rrbracket)\sigma, S)} \cdot \prod_{S \in \overline{\diamond(\llbracket P_L \rrbracket)\sigma} \setminus \Omega} \binom{\mathbf{n}(\langle \llbracket P_L \rrbracket \rangle_{\rho}\sigma, S)}{\mathbf{n}(\diamond(\llbracket P_L \rrbracket)\sigma, S)} \end{aligned}$$

If $S \notin \Omega$, then $\mathbf{n}(\langle \llbracket P_L \rrbracket \rangle_{\rho} \sigma, S) = \mathbf{n}(\diamond(\llbracket P_L \rrbracket) \sigma, S)$, hence we obtain

$$k \cdot \prod_{S \in \Omega} \begin{pmatrix} \mathbf{n}(\langle \llbracket P_L \rrbracket \rangle_{\rho} \sigma, S) \\ \mathbf{n}(\diamond(\llbracket P_L \rrbracket) \sigma, S) \end{pmatrix} \cdot \prod_{S \in \overline{\diamond(\llbracket P_L \rrbracket) \sigma} \setminus \Omega} \begin{pmatrix} \mathbf{n}(\diamond(\llbracket P_L \rrbracket) \sigma, S) \\ \mathbf{n}(\diamond(\llbracket P_L \rrbracket) \sigma, S) \end{pmatrix}$$

that is equal to

$$k \cdot \prod_{S \in \Omega} \begin{pmatrix} \mathbf{n}(\langle \llbracket P_L \rrbracket \rangle_{\rho} \sigma, S) \\ \mathbf{n}(\diamond(\llbracket P_L \rrbracket) \sigma, S) \end{pmatrix}.$$

The proof of the case in which $\llbracket \cdot \rrbracket$ is replaced by $\langle \cdot \rangle$ is analogous.

Lemma 29 *Given $P_{L1}, P_{L2} \in \mathcal{P}_L, k \in \mathbb{R}, \sigma \in \Sigma$ and $\rho : \mathcal{V}_{\mathcal{M}} \rightarrow \mathbb{N}$, the following two equalities hold*

- (a) $Rate(k2, \diamond(\llbracket P_{L1} \mid P_{L2} \rrbracket) \sigma, \langle \llbracket P_{L1} \mid P_{L2} \rrbracket \rangle_{\rho} \sigma) = Rate(k, \diamond(\llbracket P_{L1} \rrbracket) \sigma, \langle \llbracket P_{L1} \rrbracket \rangle_{\rho} \sigma) \cdot Rate(k, \diamond(\llbracket P_{L2} \rrbracket) \sigma, \langle \llbracket P_{L2} \rrbracket \rangle_{\rho} \sigma) ;$
- (b) $Rate(k2, \diamond(\llbracket (P_{X1})^L \mid P_{X2} \rrbracket \rangle_{\rho} \sigma, \langle \llbracket (P_{X1})^L \mid P_{X2} \rrbracket \rangle_{\rho} \sigma) = Rate(k, \diamond(\langle P_{X1} \rangle) \sigma, \langle \langle P_{X1} \rangle \rangle_{\rho} \sigma) \cdot Rate(k, \diamond(\langle P_{X2} \rangle) \sigma, \langle \langle P_{X2} \rangle \rangle_{\rho} \sigma) .$

The same equations hold with $\llbracket \cdot \rrbracket$ replaced by $\langle \cdot \rangle$.

PROOF. We start with the proof of equation (a). By Lemma 28 we have that $Rate(k2, \diamond(\llbracket P_{L1} \mid P_{L2} \rrbracket) \sigma, \langle \llbracket P_{L1} \mid P_{L2} \rrbracket \rangle_{\rho} \sigma)$ is equal to

$$k2 \prod_{S \in \Omega} \begin{pmatrix} \mathbf{n}(\langle \llbracket P_{L1} \mid P_{L2} \rrbracket \rangle_{\rho} \sigma, S) \\ \mathbf{n}(\diamond(\llbracket P_{L1} \mid P_{L2} \rrbracket) \sigma, S) \end{pmatrix} \quad (\text{A.1})$$

where $\Omega = \overline{\diamond(\llbracket P_{L1} \mid P_{L2} \rrbracket) \sigma} \cap \overline{(\langle \llbracket P_{L1} \mid P_{L2} \rrbracket \rangle_{\rho} \sigma \setminus \diamond(\llbracket P_{L1} \mid P_{L2} \rrbracket) \sigma)}$. Now, we have

$$\begin{aligned} \diamond \llbracket P_{L1} \mid P_{L2} \rrbracket \sigma &= \diamond(\llbracket P_{L1} \rrbracket \mid \llbracket P_{L2} \rrbracket) \sigma = \\ &(\diamond(\llbracket P_{L1} \rrbracket) \mid \diamond(\llbracket P_{L2} \rrbracket)) \sigma = \diamond(\llbracket P_{L1} \rrbracket) \sigma \mid \diamond(\llbracket P_{L2} \rrbracket) \sigma \end{aligned}$$

and

$$\begin{aligned} \langle \llbracket P_{L1} \mid P_{L2} \rrbracket \rangle_{\rho} \sigma &= \langle \llbracket P_{L1} \rrbracket \mid \llbracket P_{L2} \rrbracket \rangle_{\rho} \sigma = \\ &(\langle \llbracket P_{L1} \rrbracket \rangle_{\rho} \mid \langle \llbracket P_{L2} \rrbracket \rangle_{\rho}) \sigma = \langle \llbracket P_{L1} \rrbracket \rangle_{\rho} \sigma \mid \langle \llbracket P_{L2} \rrbracket \rangle_{\rho} \sigma . \end{aligned}$$

By these equations and by using simple arithmetics of multisets we can derive:

$$\begin{aligned} &\langle \llbracket P_{L1} \mid P_{L2} \rrbracket \rangle_{\rho} \sigma \setminus \diamond(\llbracket P_{L1} \mid P_{L2} \rrbracket) \sigma \\ &= (\langle \llbracket P_{L1} \rrbracket \rangle_{\rho} \sigma \mid \langle \llbracket P_{L2} \rrbracket \rangle_{\rho} \sigma) \setminus (\diamond(\llbracket P_{L1} \rrbracket) \sigma \mid \diamond(\llbracket P_{L2} \rrbracket) \sigma) \\ &= (\langle \llbracket P_{L1} \rrbracket \rangle_{\rho} \sigma \setminus \diamond(\llbracket P_{L1} \rrbracket) \sigma) \cap (\langle \llbracket P_{L1} \rrbracket \rangle_{\rho} \sigma \setminus \diamond(\llbracket P_{L2} \rrbracket) \sigma) \\ &\quad \cup (\langle \llbracket P_{L2} \rrbracket \rangle_{\rho} \sigma \setminus \diamond(\llbracket P_{L2} \rrbracket) \sigma) \cap (\langle \llbracket P_{L2} \rrbracket \rangle_{\rho} \sigma \setminus \diamond(\llbracket P_{L1} \rrbracket) \sigma) \end{aligned}$$

Now, $\langle \llbracket P_{L1} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L1} \rrbracket)\sigma$ denotes the multiset of all and only those strings that occur in the instantiation of some unique or maximal matching in $\llbracket P_{L1} \rrbracket$. The instantiation of a unique matching is ensured to be unique, hence it does not occur in $\diamond(\llbracket P_{L1} \rrbracket \mid P_{L2})\sigma$. As regards the maximal matchings, they are encoding of some term variables in P_{L1} . By the definition of left patterns we have that term variables can occur only in a operand of a containment operator. Moreover, the definition of $\llbracket \cdot \rrbracket$ ensures that the strings obtained by the encoding of an application of a containment operator differ from all the other strings by the index of some λ_i or $\bar{\lambda}_i$ symbol they contain. In particular, they will be different from all the strings obtained by the encoding of P_{L2} . As a consequence, we have $(\langle \llbracket P_{L1} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L1} \rrbracket)\sigma) \cap \diamond(\llbracket P_{L2} \rrbracket)\sigma = \emptyset$, that implies $(\langle \llbracket P_{L1} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L1} \rrbracket)\sigma) \subseteq (\langle \llbracket P_{L1} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L2} \rrbracket)\sigma)$, that implies $(\langle \llbracket P_{L1} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L1} \rrbracket)\sigma) \cap (\langle \llbracket P_{L1} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L2} \rrbracket)\sigma) = \langle \llbracket P_{L1} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L1} \rrbracket)\sigma$. The same reasoning holds by inverting the roles of P_{L1} and P_{L2} , and finally we obtain:

$$\begin{aligned} \langle \llbracket P_{L1} \rrbracket \mid P_{L2} \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L1} \rrbracket \mid P_{L2})\sigma = \\ (\langle \llbracket P_{L1} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L1} \rrbracket)\sigma) \cup (\langle \llbracket P_{L2} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L2} \rrbracket)\sigma). \end{aligned}$$

Let us consider again formula A.1. Now, we can write

$$\begin{aligned} \Omega &= \overline{\diamond(\llbracket P_{L1} \rrbracket \mid P_{L2})\sigma} \cap \overline{(\langle \llbracket P_{L1} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L1} \rrbracket)\sigma) \cup (\langle \llbracket P_{L2} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L2} \rrbracket)\sigma)} \\ &= (\overline{\diamond(\llbracket P_{L1} \rrbracket \mid P_{L2})\sigma} \cap \overline{(\langle \llbracket P_{L1} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L1} \rrbracket)\sigma)}) \\ &\quad \cup (\overline{\diamond(\llbracket P_{L1} \rrbracket \mid P_{L2})\sigma} \cap \overline{(\langle \llbracket P_{L2} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L2} \rrbracket)\sigma)}) \\ &= (\overline{\diamond(\llbracket P_{L1} \rrbracket)\sigma} \cap \overline{(\langle \llbracket P_{L1} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L1} \rrbracket)\sigma)}) \\ &\quad \cup (\overline{\diamond(\llbracket P_{L2} \rrbracket)\sigma} \cap \overline{(\langle \llbracket P_{L2} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L2} \rrbracket)\sigma)}). \end{aligned}$$

where the last equality is again a consequence of $(\langle \llbracket P_{L1} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{L1} \rrbracket)\sigma) \cap \diamond(\llbracket P_{L2} \rrbracket)\sigma = \emptyset$ (and the same with P_{L1} and P_{L2} inverted). Formula A.1 can now be rewritten as

$$\prod_{i=1} 2k \prod_{S \in \Omega_i} \begin{pmatrix} \mathbf{n}(\langle \llbracket P_{L1} \rrbracket \mid P_{L2} \rangle_{\rho\sigma}, S) \\ \mathbf{n}(\diamond(\llbracket P_{L1} \rrbracket \mid P_{L2})\sigma, S) \end{pmatrix}$$

where $\Omega_i = \overline{\diamond(\llbracket P_{Li} \rrbracket)\sigma} \cap \overline{(\langle \llbracket P_{Li} \rrbracket \rangle_{\rho\sigma} \setminus \diamond(\llbracket P_{Li} \rrbracket)\sigma)}$. This is equal to

$$\prod_{i=1} 2k \prod_{S \in \Omega_i} \begin{pmatrix} \mathbf{n}(\langle \llbracket P_{L1} \rrbracket \rangle_{\rho\sigma} \mid \langle \llbracket P_{L2} \rrbracket \rangle_{\rho\sigma}, S) \\ \mathbf{n}(\diamond(\llbracket P_{L1} \rrbracket)\sigma \mid \diamond(\llbracket P_{L2} \rrbracket)\sigma, S) \end{pmatrix}$$

that is

$$\prod_{i=1} 2k \prod_{S \in \Omega_i} \begin{pmatrix} \mathbf{n}(\langle \llbracket P_{L1} \rrbracket \rangle_{\rho\sigma}, S) + \mathbf{n}(\langle \llbracket P_{L2} \rrbracket \rangle_{\rho\sigma}, S) \\ \mathbf{n}(\diamond(\llbracket P_{L1} \rrbracket)\sigma, S) + \mathbf{n}(\diamond(\llbracket P_{L2} \rrbracket)\sigma, S) \end{pmatrix}.$$

As we have already observed, if S is obtained by the instantiation of some term variable in P_{Li} , then it does not occur in P_{Lj} with $i \neq j$, hence $\mathbf{n}(\langle \llbracket P_{Lj} \rrbracket \rangle_{\rho\sigma}, S) =$

$\mathbf{n}(\diamond(\llbracket P_{L_j} \rrbracket)\sigma, S) = 0$. As a consequence, the formula can be simplified into

$$\prod_{i=1} 2k \prod_{S \in \Omega_i} \begin{pmatrix} \mathbf{n}(\langle \llbracket P_{L_i} \rrbracket \rangle_\rho \sigma, S) \\ \mathbf{n}(\diamond(\llbracket P_{L_i} \rrbracket)\sigma, S) \end{pmatrix}$$

that is exactly $Rate(k, \diamond(\llbracket P_{L_1} \rrbracket)\sigma, \langle \llbracket P_{L_1} \rrbracket \rangle_\rho \sigma) \cdot Rate(k, \diamond(\llbracket P_{L_2} \rrbracket)\sigma, \langle \llbracket P_{L_2} \rrbracket \rangle_\rho \sigma)$.

As regards equation (b), by following the line of the proof of (a) we can exploit Lemma 28 to obtain a formula analogous to A.1, but with $\llbracket P_{L_1} \mid P_{L_2} \rrbracket$ replaced by $\llbracket (P_{X_1})^L \mid P_{X_2} \rrbracket$, and we can rewrite $\diamond(\llbracket (P_{X_1})^L \mid P_{X_2} \rrbracket)$ and $\langle \llbracket (P_{X_1})^L \mid P_{X_2} \rrbracket \rangle_\rho \sigma$ (by applying also Lemma 28) so to obtain:

$$\diamond(\llbracket (P_{X_1})^L \mid P_{X_2} \rrbracket)\sigma = (\bar{\lambda}_x \triangleright \diamond(\langle \llbracket P_{X_1} \rrbracket \rangle_\rho \sigma) \mid (\lambda_x \triangleright \diamond(\langle \llbracket P_{X_2} \rrbracket \rangle_\rho \sigma)))\sigma$$

and

$$\langle \llbracket (P_{X_1})^L \mid P_{X_2} \rrbracket \rangle_\rho \sigma = (\bar{\lambda}_x \triangleright \langle \langle \llbracket P_{X_1} \rrbracket \rangle_\rho \sigma \mid (\lambda_x \triangleright \langle \langle \llbracket P_{X_2} \rrbracket \rangle_\rho \sigma \rangle_\rho \sigma))\sigma$$

for some variable $x \in \mathcal{V}_{\mathcal{E}} \setminus \mathcal{X}$ that does not occur neither in $\langle \llbracket P_{X_1} \rrbracket \rangle_\rho \sigma$ nor in $\langle \llbracket P_{X_2} \rrbracket \rangle_\rho \sigma$.

By observing that any string in $(\bar{\lambda}_x \triangleright \diamond(\langle \llbracket P_{X_1} \rrbracket \rangle_\rho \sigma))\sigma$ is obviously different (in its first symbol) from any string in $(\lambda_x \triangleright \diamond(\langle \llbracket P_{X_2} \rrbracket \rangle_\rho \sigma))\sigma$, and that the same holds for $(\bar{\lambda}_x \triangleright \langle \langle \llbracket P_{X_1} \rrbracket \rangle_\rho \sigma \rangle_\rho \sigma)$ and $(\lambda_x \triangleright \langle \langle \llbracket P_{X_2} \rrbracket \rangle_\rho \sigma \rangle_\rho \sigma)$, we can follow the reasoning given in the proof of (a) to conclude that $Rate(k2, \diamond(\llbracket (P_{X_1})^L \mid P_{X_2} \rrbracket)\sigma, \langle \llbracket (P_{X_1})^L \mid P_{X_2} \rrbracket \rangle_\rho \sigma)$ is equal to

$$k2 \prod_{S \in \Omega_1} \begin{pmatrix} \mathbf{n}(\langle \bar{\lambda}_x \triangleright \langle \langle \llbracket P_{X_1} \rrbracket \rangle_\rho \sigma \rangle_\rho \sigma, S) \\ \mathbf{n}(\bar{\lambda}_x \triangleright \diamond(\langle \llbracket P_{X_1} \rrbracket \rangle_\rho \sigma), S) \end{pmatrix} \prod_{S \in \Omega_2} \begin{pmatrix} \mathbf{n}(\langle \lambda_x \triangleright \langle \langle \llbracket P_{X_2} \rrbracket \rangle_\rho \sigma \rangle_\rho \sigma, S) \\ \mathbf{n}(\lambda_x \triangleright \diamond(\langle \llbracket P_{X_2} \rrbracket \rangle_\rho \sigma), S) \end{pmatrix}$$

with $\Omega_1 = \overline{(\bar{\lambda}_x \triangleright \diamond(\langle \llbracket P_{X_1} \rrbracket \rangle_\rho \sigma))\sigma} \cap \overline{(\bar{\lambda}_x \triangleright \langle \langle \llbracket P_{X_1} \rrbracket \rangle_\rho \sigma \rangle_\rho \sigma) \setminus (\bar{\lambda}_x \triangleright \diamond(\langle \llbracket P_{X_1} \rrbracket \rangle_\rho \sigma))\sigma}$ and $\Omega_2 = \overline{(\lambda_x \triangleright \diamond(\langle \llbracket P_{X_2} \rrbracket \rangle_\rho \sigma))\sigma} \cap \overline{(\lambda_x \triangleright \langle \langle \llbracket P_{X_2} \rrbracket \rangle_\rho \sigma \rangle_\rho \sigma) \setminus (\lambda_x \triangleright \diamond(\langle \llbracket P_{X_2} \rrbracket \rangle_\rho \sigma))\sigma}$. Since all the strings in Ω_i start with the same symbol ($\bar{\lambda}_{\sigma(x)}$ and $\lambda_{\sigma(x)}$ for $i = 1$ and $i = 2$, respectively) we can rewrite the formula as

$$\prod_{i=1} 2k \prod_{S \in \Omega'_i} \begin{pmatrix} \mathbf{n}(\langle \llbracket P_{X_i} \rrbracket \rangle_\rho \sigma, S) \\ \mathbf{n}(\diamond(\llbracket P_{X_i} \rrbracket)\sigma, S) \end{pmatrix}$$

with $\Omega'_i = \overline{(\diamond(\llbracket P_{X_i} \rrbracket)\sigma)} \cap \overline{(\langle \llbracket P_{X_i} \rrbracket \rangle_\rho \sigma) \setminus (\diamond(\llbracket P_{X_i} \rrbracket)\sigma)}$, and this corresponds exactly to $Rate(k, \diamond(\llbracket P_{X_1} \rrbracket)\sigma, \langle \llbracket P_{X_1} \rrbracket \rangle_\rho \sigma) \cdot Rate(k, \diamond(\llbracket P_{X_2} \rrbracket)\sigma, \langle \llbracket P_{X_2} \rrbracket \rangle_\rho \sigma)$.

The proofs of both (a) and (b) when $\llbracket \cdot \rrbracket$ is replaced by $\langle \cdot \rangle$ are analogous.

Lemma 30 *Given $P_L \in \mathcal{P}_L, \sigma \in \Sigma_{cls}, k \in \mathbb{R}$ and the σ -compliant pattern expansion parameter function $\rho^{\llbracket \cdot \rrbracket}$, it holds*

$$k \cdot comb(P_L, \sigma) = \sum_{\sigma^{\llbracket \cdot \rrbracket} \in \Sigma_\sigma(P_L)} Rate(k, \diamond(\llbracket P_L \rrbracket)\sigma^{\llbracket \cdot \rrbracket}, \langle \llbracket P_L \rrbracket \rangle_{\rho^{\llbracket \cdot \rrbracket}} \sigma^{\llbracket \cdot \rrbracket})$$

where $\Sigma_\sigma(P_L)$ is one of the greatest sets of σ -compliant instantiation functions such that $\sigma_1^{\llbracket \cdot \rrbracket}, \sigma_2^{\llbracket \cdot \rrbracket} \in \Sigma_\sigma(P_L)$ implies $\diamond(\llbracket P_L \rrbracket)\sigma_1^{\llbracket \cdot \rrbracket} \equiv \diamond(\llbracket P_L \rrbracket)\sigma_2^{\llbracket \cdot \rrbracket}$ and $\sigma_1^{\llbracket \cdot \rrbracket}(v) = \sigma_2^{\llbracket \cdot \rrbracket}(v)$ for any $v \notin \text{Var}(\llbracket P_L \rrbracket)$.

PROOF. We first note that there are always infinite possible sets $\Sigma_\sigma(P_L)$ as there are infinite possible instantiations for the variables $v \notin \text{Var}(\llbracket P_L \rrbracket)$. However, the instantiations in $\Sigma_\sigma(P_L)$ are σ -compliant and this means that they agree in the instantiation of variables in $\mathcal{X} \cup SV \cup \mathcal{V}_S(TV)$. Moreover, by the definition of the encoding, $\text{Var}(\llbracket P_L \rrbracket) \subset (\mathcal{V}_\varepsilon \setminus \mathcal{X}) \cup \mathcal{X} \cup SV \cup \mathcal{V}_S(TV)$ where $\mathcal{V}_S(TV)$ is the union of all $\mathcal{V}_S(X)$ for all $X \in TV$ and $\mathcal{V}_\varepsilon \setminus \mathcal{X}$ is used to provide variables used as subscripts of λ and $\bar{\lambda}$ symbols. As a consequence, we have that the size of any $\Sigma_\sigma(P_L)$ is equal to the number of instantiations of the subscripts of the λ and $\bar{\lambda}$ symbols that correspond to structurally equivalent instantiations of $\llbracket P_L \rrbracket$. Now, it is easy to see that the result of $\text{Rate}(k, \diamond(\llbracket P_L \rrbracket)\sigma^{\llbracket \cdot \rrbracket}, \langle \llbracket P_L \rrbracket \rangle_{\rho^{\llbracket \cdot \rrbracket}}\sigma^{\llbracket \cdot \rrbracket})$ is the same for any σ -compliant $\sigma^{\llbracket \cdot \rrbracket}$, hence the lemma can be reformulated as

$$k \cdot \text{comb}(P_L, \sigma) = |\Sigma_\sigma(P_L)| \cdot \text{Rate}(k, \diamond(\llbracket P_L \rrbracket)\sigma^{\llbracket \cdot \rrbracket}, \langle \llbracket P_L \rrbracket \rangle_{\rho^{\llbracket \cdot \rrbracket}}\sigma^{\llbracket \cdot \rrbracket}) \quad (\text{A.2})$$

Let us first prove A.2 with $\llbracket \cdot \rrbracket$ replaced by $\langle \cdot \rangle$. We prove this by induction on the structure of P_L .

- Base case:
 - Let $P_L = SP$;
It holds $k \cdot \text{comb}(SP, \sigma) = k$. Moreover, $|\Sigma_\sigma(P_L)| = 1$ as $\langle SP \rangle$ does not contain any λ and $\bar{\lambda}$. Hence, we have to prove $\text{Rate}(k, \diamond(\langle P_L \rangle)\sigma^{\llbracket \cdot \rrbracket}, \langle \langle P_L \rangle \rangle_{\rho^{\llbracket \cdot \rrbracket}}\sigma^{\llbracket \cdot \rrbracket}) = k$. We have $\diamond(\langle SP \rangle)\sigma^{\llbracket \cdot \rrbracket} = \diamond(\{SP\})\sigma^{\llbracket \cdot \rrbracket} = \epsilon\sigma^{\llbracket \cdot \rrbracket} = \epsilon$ and $\langle \langle SP \rangle \rangle_{\rho^{\llbracket \cdot \rrbracket}}\sigma^{\llbracket \cdot \rrbracket} = \langle \{SP\} \rangle_{\rho^{\llbracket \cdot \rrbracket}}\sigma^{\llbracket \cdot \rrbracket} = SP\sigma^{\llbracket \cdot \rrbracket}$ and $\text{Rate}(k, \epsilon, SP\sigma^{\llbracket \cdot \rrbracket}) = k$.
- Induction cases:
 - Let $P_L = P_{L1} \mid P_{L2}$;
We first note that $k \cdot \text{comb}(P_{L1} \mid P_{L2}, \sigma) = k \cdot \text{comb}(P_{L1}, \sigma) \cdot \text{comb}(P_{L2}, \sigma) = (k \cdot \text{comb}(P_{L1}, \sigma)) \cdot (k \cdot \text{comb}(P_{L2}, \sigma)) \cdot \frac{1}{k}$. Now, by induction hypothesis we have that $k \cdot \text{comb}(P_{Li}, \sigma) = |\Sigma_\sigma(P_{Li})| \cdot \text{Rate}(k, \diamond(\langle P_{Li} \rangle)\sigma^{\llbracket \cdot \rrbracket}, \langle \langle P_{Li} \rangle \rangle_{\rho^{\llbracket \cdot \rrbracket}}\sigma^{\llbracket \cdot \rrbracket})$, hence $k \cdot \text{comb}(P_{L1}, \sigma) \cdot \text{comb}(P_{L2}, \sigma) = |\Sigma_\sigma(P_{L1} \mid P_{L2})| \cdot \text{Rate}(k, \diamond(\langle P_{L1} \rangle)\sigma^{\llbracket \cdot \rrbracket}, \langle \langle P_{L1} \rangle \rangle_{\rho^{\llbracket \cdot \rrbracket}}\sigma^{\llbracket \cdot \rrbracket}) \cdot \text{Rate}(k, \diamond(\langle P_{L2} \rangle)\sigma^{\llbracket \cdot \rrbracket}, \langle \langle P_{L2} \rangle \rangle_{\rho^{\llbracket \cdot \rrbracket}}\sigma^{\llbracket \cdot \rrbracket}) \cdot \frac{1}{k}$. By Lemma 29 we have that this is equivalent to $|\Sigma_\sigma(P_{L1} \mid P_{L2})| \cdot \text{Rate}(k, \diamond(\langle P_{L1} \mid P_{L2} \rangle)\sigma^{\llbracket \cdot \rrbracket}, \langle \langle P_{L1} \mid P_{L2} \rangle \rangle_{\rho^{\llbracket \cdot \rrbracket}}\sigma^{\llbracket \cdot \rrbracket}) \cdot \frac{1}{k}$, that is $|\Sigma_\sigma(P_{L1} \mid P_{L2})| \cdot \text{Rate}(k, \diamond(\langle P_{L1} \mid P_{L2} \rangle)\sigma^{\llbracket \cdot \rrbracket}, \langle \langle P_{L1} \mid P_{L2} \rangle \rangle_{\rho^{\llbracket \cdot \rrbracket}}\sigma^{\llbracket \cdot \rrbracket})$.
 - Let $P_L = (P_{X1})^L \mid P_{X2}$;
We have $k \cdot \text{comb}((P_{X1})^L \mid P_{X2}, \sigma) = k \cdot \text{comb}(P_{X1}, \sigma) \cdot \text{comb}(P_{X2}, \sigma)$ that corresponds to $(k \cdot \text{comb}'(P_{X1}, \sigma)) \cdot (k \cdot \text{comb}'(P_{X2}, \sigma)) \cdot \frac{1}{k}$. Now, we have four cases depending on the syntax of P_{X1} and P_{X2} . We only consider the case in which $P_{X1} = P_{L1} \mid X$ and $P_{X2} = P_{L2}$ as it is the most interesting. In this case $(k \cdot \text{comb}'(P_{X1}, \sigma)) \cdot (k \cdot \text{comb}'(P_{X2}, \sigma)) \cdot \frac{1}{k}$ is equal to

$(k \cdot \prod_{T \in \overline{P_{L1}\sigma}} \binom{\mathbf{n}((P_L|X)\sigma, T)}{P_L\sigma, T}) \cdot \text{comb}(P_{L1})$) $(k \cdot \text{comb}(P_{L2}) \cdot \frac{1}{k})$. By induction hypothesis we have that this is equal to $\text{Rate}(k, \diamond(\langle P_{L1} \rangle)\sigma^{\llbracket 1 \rrbracket}, \langle \langle P_{L1} \rangle \rangle_{\rho^{\llbracket 1 \rrbracket}}\sigma^{\llbracket 1 \rrbracket}) \cdot \text{Rate}(k, \diamond(\langle P_{L2} \rangle)\sigma^{\llbracket 1 \rrbracket}, \langle \langle P_{L2} \rangle \rangle_{\rho^{\llbracket 1 \rrbracket}}\sigma^{\llbracket 1 \rrbracket}) \cdot \prod_{T \in \overline{P_{L1}\sigma}} \binom{\mathbf{n}((P_{L1}|X)\sigma)}{\mathbf{n}(P_{L1}\sigma)}$. Now, if $T \in \overline{P_{L1}\sigma}$ is a sequence $SP\sigma$, then there are as many such sequences in $P_{L1}\sigma$ and in $\sigma(X)$ as $\langle \langle SP \rangle \rangle_{\rho^{\llbracket 1 \rrbracket}}$ in $\diamond(\langle P_{L1} | X \rangle)\sigma^{\llbracket 1 \rrbracket}$ and $\langle \langle \bullet \rangle \rangle_{\rho^{\llbracket 1 \rrbracket}}$, respectively. Otherwise, if T is a term $((P_{L3})^L \rfloor P_{L4})\sigma$, then, since each occurrence of $\langle \langle (P_{L3})^L \rfloor P_{L4} \rangle \rangle_{\rho^{\llbracket 1 \rrbracket}}$ in $\diamond(P_{L1} | X)\sigma^{\llbracket 1 \rrbracket}$ has a different index i used as subscript of its λ and $\bar{\lambda}$ symbols, we have that there are as many $((P_{L3})^L \rfloor P_{L4})\sigma$ in $(P_{L1} | X)\sigma$ as possible instantiations of $\sigma^{\llbracket 1 \rrbracket}$ in $\Sigma_\sigma(P_{L1} | X)$. As a consequence, we can write

$$\prod_{T \in \overline{P_{L1}\sigma}} \binom{\mathbf{n}((P_L | X)\sigma, T)}{P_L\sigma, T} = |\Sigma_\sigma(P_{L1} | X)| \cdot \prod_{S \in \diamond(\langle P_L | X \rangle)} \binom{\mathbf{n}(\langle \langle P_L | X \rangle \rangle_{\rho^{\llbracket 1 \rrbracket}}\sigma^{\llbracket 1 \rrbracket}, S)}{\mathbf{n}(\langle \langle P_L | X \rangle \rangle_{\rho^{\llbracket 1 \rrbracket}}\sigma^{\llbracket 1 \rrbracket}, S)}$$

and use this to obtain Equation A.2.

Now, we split Theorem 15 into soundness and completeness, and prove them separately.

Theorem 31 (Soundness) *Given a finite set of Stochastic CLS rewrite rules \mathcal{R} and $T, T' \in \mathcal{T}$, it holds $T \xrightarrow{R, r} T' \implies \llbracket T \rrbracket \xrightarrow{R^{\llbracket 1 \rrbracket}, r} \llbracket T' \rrbracket$ where $R^{\llbracket 1 \rrbracket}$ is the translation of R into sSMSR.*

PROOF. By definition of the semantics of Stochastic CLS we have that, in order to prove the theorem, we have to prove that $T \xrightarrow{R, T'', r', b} T'$ implies $\llbracket R \rrbracket \xrightarrow{R^{\llbracket 1 \rrbracket}, r}$ with $r' \cdot b = r$. We prove this by induction on the derivation of $T \xrightarrow{R, T'', r', b} T'$. Let us first consider the case of the closure of the semantics with respect to \equiv . In this case we have that the transition performed by T is derived by applying one of the inference rules of the semantics of Stochastic CLS to a term T''' such that $T \equiv T'''$. It is easy to see that the application of most of the axioms of the structural congruence of Stochastic CLS can be simulated by the application of axioms of the structural congruence of sSMSR. This does not hold for axioms $T \mid \epsilon \equiv T$ and $(\epsilon)^L \rfloor \epsilon \equiv \epsilon$ of Stochastic CLS. However, the application of these axioms does not enable the application of any new rewrite rule.

Now, we have to consider the four cases corresponding to the inference rules of the semantics of Stochastic CLS. In all these cases we assume $R = P_L \xrightarrow{k} P_R$ and, consequently, $R^{\llbracket 1 \rrbracket} = \llbracket P_L \rrbracket \xrightarrow{k} \llbracket P_R \rrbracket$.

- Let the last inference rule used to derive $T \xrightarrow{R, T'', r', b} T'$ be rule 1; we have that there exists σ such that $T \equiv P_L\sigma$ and $T' \equiv P_R\sigma$. Moreover, we have

$r' = k \cdot \text{comb}(P_L \sigma)$ and $b = 1$. By Lemma 27 we have $\langle \llbracket P_L \rrbracket \rangle_{\rho^{\llbracket 1 \rrbracket}} \sigma^{\llbracket 1 \rrbracket} \equiv \llbracket P_L \sigma \rrbracket$ and $\langle \llbracket P_R \rrbracket \rangle_{\rho^{\llbracket 1 \rrbracket}} \sigma^{\llbracket 1 \rrbracket} \equiv \llbracket P_R \sigma \rrbracket$, where $\sigma^{\llbracket 1 \rrbracket}$ and $\rho^{\llbracket 1 \rrbracket}$ are σ -compliant. This means that $R^{\llbracket 1 \rrbracket}$ is applicable to $\llbracket P_L \sigma \rrbracket$, that is $\llbracket T \rrbracket$, and consequently $\llbracket T \rrbracket \equiv \langle \llbracket P_L \rrbracket \rangle_{\rho^{\llbracket 1 \rrbracket}} \sigma^{\llbracket 1 \rrbracket} \xrightarrow{R^{\llbracket 1 \rrbracket}, \diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}, \text{Rate}(k, \diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}, \langle \llbracket P_L \rrbracket \rangle_{\rho^{\llbracket 1 \rrbracket}} \sigma^{\llbracket 1 \rrbracket})} \langle \llbracket P_R \rrbracket \rangle_{\rho^{\llbracket 1 \rrbracket}} \sigma^{\llbracket 1 \rrbracket} \equiv \llbracket T' \rrbracket$. By Definition 14 we have that there exist as many transitions like this as possible different σ -compliant instantiation functions that, once applied to $\langle \llbracket P_L \rrbracket \rangle_{\rho^{\llbracket 1 \rrbracket}} \sigma^{\llbracket 1 \rrbracket}$, give structurally congruent results. This means that in $\llbracket R \rrbracket \xrightarrow{R^{\llbracket 1 \rrbracket}, r} \llbracket T' \rrbracket$, we have $r = \sum_{\sigma^{\llbracket 1 \rrbracket} \in \Sigma_{\sigma}(P_L)} \text{Rate}(k, \diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}, \langle \llbracket P_L \rrbracket \rangle_{\rho^{\llbracket 1 \rrbracket}} \sigma^{\llbracket 1 \rrbracket})$ with $\Sigma_{\sigma}(P_L)$ defined as in Lemma 30, and hence, by applying such a lemma, we obtain that $r = k \cdot \text{comb}(P_L, \sigma) = r' \cdot b$.

- Let the last inference rule used to derive $T \xrightarrow{R, T'', r', b} T'$ be rule 2; as a consequence $T = T_1 \mid T_3$, $T' = T_2 \mid T_3$ and $T_1 \xrightarrow{R, T'', r', b'} T_2$ with $b = b' \cdot \text{binom}(T'', T_1, T_3)$. By definition of $\llbracket \cdot \rrbracket$ we have $\llbracket T_1 \mid T_3 \rrbracket = \llbracket T_1 \rrbracket \mid \llbracket T_3 \rrbracket$ and $\llbracket T_2 \mid T_3 \rrbracket = \llbracket T_2 \rrbracket \mid \llbracket T_3 \rrbracket$. By induction hypothesis we have that $\llbracket T_1 \rrbracket \xrightarrow{R^{\llbracket 1 \rrbracket}, r''} \llbracket T_2 \rrbracket$ with $r'' = r' \cdot b'$. We have to prove that $\llbracket T_1 \rrbracket \mid \llbracket T_3 \rrbracket \xrightarrow{R^{\llbracket 1 \rrbracket}, r} \llbracket T_2 \rrbracket \mid \llbracket T_3 \rrbracket$ with $r = r' \cdot b = r' \cdot b' \cdot \text{binom}(T'', T_1, T_3) = r'' \cdot \text{binom}(T'', T_1, T_3)$. Now, by definition of Rate we have that $\text{Rate}(k, \diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}, \llbracket T_1 \rrbracket \mid \llbracket T_3 \rrbracket)$ is

$$k \frac{\prod_{S \in \diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}} \binom{\mathbf{n}(\llbracket T_1 \rrbracket \mid \llbracket T_3 \rrbracket, S)}{\mathbf{n}(\diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}, S)}}{\prod_{S \in \diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}} \binom{\mathbf{n}(\llbracket T_1 \rrbracket, S) + \mathbf{n}(\llbracket T_3 \rrbracket, S)}{\mathbf{n}(\diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}, S)}}$$

that is, by assuming $\overline{\text{binom}}$ to be the analogous of binom defined on sSMSR string multisets,

$$k \frac{\prod_{S \in \diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}} \binom{\mathbf{n}(\llbracket T_1 \rrbracket, S)}{\mathbf{n}(\diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}, S)}}{\prod_{S \in \diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}} \binom{\mathbf{n}(\llbracket T_1 \rrbracket, S)}{\mathbf{n}(\diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}, S)}} \cdot \overline{\text{binom}}(\diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}, \llbracket T_1 \rrbracket, \llbracket T_3 \rrbracket)$$

that is $\text{Rate}(k, \diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}, \llbracket T_1 \rrbracket) \cdot \overline{\text{binom}}(\diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}, \llbracket T_1 \rrbracket, \llbracket T_3 \rrbracket)$. As a consequence, $r = r'' \cdot \overline{\text{binom}}(\diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}, \llbracket T_1 \rrbracket, \llbracket T_3 \rrbracket)$. Since $\diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}$ represents the reactants of $R^{\llbracket 1 \rrbracket}$, that are represented by T'' in the Stochastic CLS transition, it holds $\overline{\text{binom}}(\diamond(\llbracket P_L \rrbracket) \sigma^{\llbracket 1 \rrbracket}, \llbracket T_1 \rrbracket, \llbracket T_3 \rrbracket) = \text{binom}(T'', T_1, T_3)$. This implies $r = r'' \cdot \text{binom}(T'', T_1, T_3)$ that is exactly $r' \cdot b$.

- Let the last inference rule used to derive $T \xrightarrow{R, T'', r', b} T'$ be 3; we have $b = 1$, $T = (T_1)^L \mid T_3$, $T' = (T_2)^L \mid T_3$, $T'' = T$ and $T_1 \xrightarrow{R, T''', r'/b', b'} T_2$ for some b' and T''' . By induction hypothesis we know that $\llbracket T_1 \rrbracket \xrightarrow{R^{\llbracket 1 \rrbracket}, r'} \llbracket T_2 \rrbracket$. Now, $\llbracket (T_1)^L \rrbracket \mid T_3 = \bar{\lambda}_i \triangleright \llbracket T_1 \rrbracket \mid \lambda_i \triangleright \llbracket T_3 \rrbracket$ for some fresh index i . It is easy to see that $\bar{\lambda}_i \triangleright \llbracket T_1 \rrbracket \xrightarrow{R^{\llbracket 1 \rrbracket}, r'} \bar{\lambda}_i \triangleright \llbracket T_2 \rrbracket$, as all the introduced occurrences of symbol $\bar{\lambda}_i$ can be added to the instantiation of Δ in $R^{\llbracket 1 \rrbracket}$. As a consequence, since $\bar{\lambda}_i \triangleright \llbracket T_1 \rrbracket$ and $\lambda_i \triangleright \llbracket T_3 \rrbracket$ does not share any string, it also holds $\bar{\lambda}_i \triangleright \llbracket T_1 \rrbracket \mid \lambda_i \triangleright \llbracket T_3 \rrbracket \xrightarrow{R^{\llbracket 1 \rrbracket}, r'} \bar{\lambda}_i \triangleright \llbracket T_2 \rrbracket \mid \lambda_i \triangleright \llbracket T_3 \rrbracket$, that is $T \xrightarrow{R^{\llbracket 1 \rrbracket}, r'} T'$.

- Let the last inference rule used to derive $T \xrightarrow{R, T'', r', b} T'$ be 4; this case is analogous to the previous one.

Theorem 32 (Completeness) *Given a finite set of Stochastic CLS rewrite rules \mathcal{R} and $T, T' \in \mathcal{T}$, it holds $\llbracket T \rrbracket \xrightarrow{R^{[\cdot]}, r} \llbracket T' \rrbracket \implies T \xrightarrow{R, r} T'$ where $R^{[\cdot]}$ is the translation of R into sSMSR.*

PROOF. The proof of completeness is made easier by the fact that it is possible to define two (partial) functions $\llbracket \cdot \rrbracket^{-1}$ and $\llbracket \cdot \rrbracket^{-1}$ that can be used to translate back to Stochastic CLS the string multiset and the rewrite rules of sSMSR obtained by the encoding functions $\llbracket \cdot \rrbracket$ and $\llbracket \cdot \rrbracket$. The two functions $\llbracket \cdot \rrbracket^{-1} : \mathcal{M} \rightarrow \mathcal{T}$ and $\llbracket \cdot \rrbracket^{-1} : \mathcal{MP} \rightarrow \mathcal{P}_R$ are defined as the least functions satisfying the following rules:

$$\begin{aligned} \llbracket \bar{\lambda}_i \triangleright M_1 \mid \lambda_i \triangleright M_2 \rrbracket^{-1} &= \llbracket M_1 \rrbracket^{-1} \mid \llbracket M_2 \rrbracket^{-1} & \llbracket \bullet \cdot S \rrbracket^{-1} &= S \\ \llbracket \lambda_i \triangleright M \rrbracket^{-1} &= \llbracket M \rrbracket^{-1} & \llbracket \bar{\lambda}_i \triangleright M \rrbracket^{-1} &= \llbracket M \rrbracket^{-1} \end{aligned}$$

and

$$\begin{aligned} \llbracket \bar{\lambda}_x \triangleright MP_1 \mid \lambda_x \triangleright MP_2 \rrbracket^{-1} &= \llbracket MP_1 \rrbracket^{-1} \mid \llbracket MP_2 \rrbracket^{-1} & \llbracket \bullet \cdot SP \rrbracket^{-1} &= SP \\ \llbracket \lambda_x \triangleright MP \rrbracket^{-1} &= \llbracket MP \rrbracket^{-1} & \llbracket \bar{\lambda}_x \triangleright MP \rrbracket^{-1} &= \llbracket MP \rrbracket^{-1} \\ \llbracket \{\bullet\}_X \rrbracket^{-1} &= X & \llbracket \{\bullet \cdot SP\} \rrbracket^{-1} &= SP \end{aligned}$$

It is easy to see that $\llbracket \llbracket T \rrbracket \rrbracket^{-1} \equiv T$, $\llbracket \llbracket P_R \rrbracket \rrbracket^{-1} \equiv P_R$ and $\llbracket \llbracket P_R \rrbracket \rrbracket^{-1} \equiv P_R$ hold by the construction of $\llbracket \cdot \rrbracket^{-1}$ and $\llbracket \cdot \rrbracket^{-1}$. Now, $\llbracket T \rrbracket \xrightarrow{R^{[\cdot]}, r} \llbracket T' \rrbracket$ means that rule $R^{[\cdot]}$ can be applied to $\llbracket T \rrbracket$. The rewrite rule R can be obtained by applying $\llbracket \cdot \rrbracket^{-1}$ to the left and right patterns of $R^{[\cdot]}$, after removing all the occurrences of Δ . The definition of $\llbracket \cdot \rrbracket^{-1}$ and $\llbracket \cdot \rrbracket^{-1}$ ensures that also R can be applied to T , namely a transition $T \xrightarrow{R, r'} T'$ can be performed for some $r' \in \mathbb{R}$. Now, we only have to show that $r' = r$: by Theorem 31 we have that $T \xrightarrow{R, r'} T'$ implies $\llbracket T \rrbracket \xrightarrow{R^{[\cdot]}, r'} \llbracket T' \rrbracket$, but $\llbracket T \rrbracket \xrightarrow{R^{[\cdot]}, r} \llbracket T' \rrbracket$ is unique because, by definition, it groups all the transitions $\llbracket T \rrbracket \xrightarrow{R^{[\cdot]}, M, r''} \llbracket T' \rrbracket$ for any $M \in \mathcal{M}$ and $r'' \in \mathbb{R}$. As a consequence $r = r'$.

Theorem 15 is a direct consequence of Theorems 31 and 32.

B Proof of Theorem 21

We split Theorem 21 into soundness and completeness, and prove them separately.

Before giving the proofs we make a simple consideration: as in the encoding of any process P we never use, inside the reactants of a rule, any instance of the matching operators, then for any rule $(MP, MP', k) \in R_P$ it holds $\langle MP \rangle_\rho \equiv MP$ and $\langle MP' \rangle_\rho \equiv MP'$. Furthermore, due to the absence of the matching operators in the encoding of any SPi process, the constraint in the semantics of sSMSR which is used to provide correctness of the behavior of such an operator is always satisfied for any term representing the encoding of a SPi process. These considerations let us avoid to discuss on both the patterns expansion and on the mentioned constraint in all the following proofs. Furthermore, in the following proofs, we say top-level descriptions instead of descriptions of the top-level components for the sake of simplicity.

Theorem 33 (Soundness) *Given a SPi process P and its sSMSR encoding $\langle M_P, R_P \rangle$, it holds: $P \xrightarrow{r} P' \implies M_P \xrightarrow{R, M, v} M_{P'}$*

PROOF. We prove the theorem by induction on the rules of the semantics of SPi.

- Let $P \equiv x\langle n \rangle.P_1 + \Sigma \mid x\langle m \rangle.P_2 + \Sigma'$ and $P' \equiv P_1 \mid P_{2\{n/m\}}$, we prove $x\langle n \rangle.P_1 + \Sigma \mid x\langle m \rangle.P_2 + \Sigma' \xrightarrow{\text{rate}(x)} P_1 \mid P_{2\{n/m\}} \implies M_P \xrightarrow{R, M, v} M_{P'}$. By definition, the process description encoding of P , namely $\mathcal{I}(P, \epsilon)$, is such that the two computed descriptions of the top-level components will be denoted by two identifiers, let us assume them to be A and B . By the definition of the encoding of P the sSMSR term M_P representing the state of the system is, given any function γ , $M_P \equiv (A \mid B)_{\{\gamma(c)/c\}} \equiv A \mid B$ because no channels appear in both the patterns A and B . Furthermore, the sSMSR rewriting rule obtained by the encoding of P is $R \in R_P^x$ where $R : A \mid B \xrightarrow{\text{rate}(x)} MP_1 \mid MP_2 \mid S_A \mid S_B$. Multisets MP_1 and MP_2 are obtained as the union of sequence patterns representing the identifiers of the descriptions of the top-level components recursively computed by the function \mathcal{I} on the continuation of the input and output actions, respectively. Patterns S_A and S_B are either ϵ or A (and B respectively) if the action of the process was obtained by the application of the structural congruence relation for a replication action. In order to derive a transition of the semantics of sSMSR we must satisfy all the premises of the inference rule. We consider an instantiation function $\sigma = \{(n, v)\}$ where v is a fresh name for the rule R and for the term M_P . By the definition of the semantics of sSMSR we have $A \mid B \equiv M_1 \mid M_3$; as we have $\langle A \mid B \rangle_\rho \sigma \equiv A \mid B$, then $A \mid B \equiv M_1$ and $M_3 \equiv \epsilon$. As regards the constraints in the semantics of sSMSR we have that: the constraint on the pairs of free variables is satisfied because we have just one free variable in R , namely n , while the constraint on n is satisfied by the suitable choice of the value v . In order to derive the correct transition of the semantics we now divide the proof by cases on S_A and S_B :

- (a) if $S_A \equiv S_B \equiv \epsilon$ then both the input and output actions are not prefixed by any replication. In particular the descriptions of the top-level components of process P are $\{(A, x\langle n \rangle.(ID_1^{P_1} | \dots | ID_t^{P_1}) + \sum IP_\Sigma), (B, x\langle m \rangle.(ID_1^{P_2} \cdot m | \dots | ID_k^{P_2} \cdot m) + \sum IP_{\Sigma'})\}$. Patterns $ID_i^{P_1}$ with $i = 1, \dots, t$ and patterns $ID_j^{P_2}$ with $j = 1, \dots, k$ are the identifiers of the descriptions of the inner-level components recursively computed on P_1 and P_2 , respectively; patterns $\sum IP_\Sigma$ and $\sum IP_{\Sigma'}$ are the intermediate process descriptions recursively computed on Σ and Σ' , respectively. By satisfying all the premises of the inference rule we derive the sSMSR transition $A | B \xrightarrow{R, A|B, rate(x)} M_2$ where $M_2 \equiv ID_1^{P_1} | \dots | ID_t^{P_1} | ID_1^{P_2} \cdot v | \dots | ID_k^{P_2} \cdot v$ represents the encoding of P' .
- (b) if $S_A \equiv A$ and $S_B \equiv \epsilon$ then the output action is prefixed by a replication, namely was of the form $!x\langle n \rangle.\bar{P}$ and the process does not contain any action in Σ . By using the structural congruence relation on SPi process, the action is rewritten in $x\langle n \rangle.(\bar{P} | !x\langle n \rangle.\bar{P})$ with $P_1 \equiv (\bar{P} | !x\langle n \rangle.\bar{P})$. With respect to the encoding of P the description of the top-level component A is, in this case, of the form $(A, !x\langle n \rangle.(ID_1^{P_1} | \dots | ID_t^{P_1}))$. Patterns $ID_i^{P_1}$ with $i = 1, \dots, t$ are the identifiers of the descriptions of the inner-level components recursively computed on \bar{P} . The description of B is the same as in case (a) of the proof. The transition of the semantics of sSMSR derived in this case is $A | B \xrightarrow{R, A|B, rate(x)} M_2$, where $M_2 \equiv ID_1^{P_1} | \dots | ID_t^{P_1} | ID_1^{P_2} \cdot v | ID_k^{P_2} \cdot v | A$ represents the encoding of P' .
- (c) if $S_A \equiv \epsilon$ and $S_B \equiv B$ then the proof is analogous to the case (b) where only the input action is prefixed by a replication.
- (d) if $S_A \equiv A$ and $S_B \equiv B$ then the proof is analogous to the combination of cases (c) and (b) where both the input and the output actions are prefixed by a replication.

- We prove $\nu x P \xrightarrow{r} \nu x P' \implies M_{\nu x P} \xrightarrow{R, M, v} M_{\nu x P'}$ where $M_{\nu x P}$ and $M_{\nu x P'}$ denote the encoding of processes $\nu x P$ and $\nu x P'$, respectively. The rule of the SPi semantics has got the premise $P \xrightarrow{r} P'$; we assume the theorem on P and P' , namely we assume $P \xrightarrow{r} P' \implies M_P \xrightarrow{R', M', v'} M_{P'}$ where $M' \equiv ID_1^{M'} \cdot SP_1^{M'} | \dots | ID_k^{M'} \cdot SP_k^{M'}$. Notice that rule R' , used to derive the transition $M_P \xrightarrow{R', M', v'} M_{P'}$, belongs to the set of rules R_P which is, by the definition of \mathcal{I} , different from the set of rules $R_{\nu x P}$. By definition the process descriptions derived by the encoding of $\nu x P$ and of $\nu x P'$ are $\mathcal{I}(\nu x P, \epsilon) = \mathcal{I}(P, x) = (D, D', E)$ and $\mathcal{I}(\nu x P', \epsilon) = \mathcal{I}(P', x) = (D_1, D'_1, E_1)$, respectively. Let us assume $\mathcal{I}(P, x) = (D_2, D'_2, E_2)$ and $\mathcal{I}(P', x) = (D_3, D'_3, E_3)$, by the definition of \mathcal{I} it holds that $E = E_2$, $E_1 = E_3$, $\forall (ID \cdot SP, IP) \in D_2 \cup D'_2$. $(ID \cdot x \cdot SP, IP) \in D \cup D'$ and $\forall (ID \cdot SP, IP) \in D_3 \cup D'_3$. $(ID \cdot x \cdot SP, IP) \in D_1 \cup D'_1$. We show that, as we can derive the transition $M_P \xrightarrow{R', M', v'} M_{P'}$, then by modifying the values which satisfy the premises of the inference rule, we can derive the transition $M_{\nu x P} \xrightarrow{R, M, v} M_{\nu x P'}$. In particular, the

term $M_{\nu xP}$, obtained by the encoding of νxP , is $(ID_1 \cdot x \cdot SP_1 \mid \dots \mid ID_k \cdot x \cdot SP_k)_{\{\gamma(c)/c\}}$, while the term M_P , obtained by the encoding of P , is $(ID_1 \cdot SP_1 \mid \dots \mid ID_k \cdot SP_k)_{\{\gamma(c)/c\}}$. If the reactants of R' would have been instantiated by using an instantiation function σ in order to match the term M_P , then the instantiation function σ' used to apply rule R to $M_{\nu xP}$ is obtained by extending σ : $\sigma' = \sigma \cup \{(x, \gamma(x))\}$ for the same γ function used to encode P . As regards the constraints, we have that, as they are satisfied by the application of the rule R' with state M_P , then they are still satisfied by the the application of R in state $M_{\nu xP}$. Furthermore, as M_P is rewritten into $M_{P'} \equiv (ID'_1 \cdot SP'_1 \mid \dots \mid ID'_t \cdot SP'_t)_{\{\gamma(c)/c\}}$ then by the definition of \mathcal{I} it holds $M_{\nu xP'} \equiv (ID'_1 \cdot x \cdot SP'_1 \mid \dots \mid ID'_k \cdot x \cdot SP'_t)_{\{\gamma(c)/c\}}$. It is clear that is possible to derive the transition of the semantics of sSMSR:P $M_{\nu xP} \xrightarrow{R, M'_\nu, v} M_{\nu xP'}$, where $M'_\nu \equiv ID_1^{M'} \cdot \sigma'(x) \cdot SP_1^{M'} \mid \dots \mid ID_k^{M'} \cdot \sigma'(x) \cdot SP_k^{M'}$ and $M_{\nu xP}$ and $M_{\nu xP'}$ represent the encoding of νxP and $\nu xP'$, respectively.

- We prove $Q \mid P \xrightarrow{r} Q \mid P' \implies M_{Q|P} \xrightarrow{R, M, v} M_{Q|P'}$ where $M_{Q|P}$ and $M_{Q|P'}$ denote the encoding of process $Q \mid P$ and $Q \mid P'$, respectively. The rule of the SPi semantics has got the premise $P \xrightarrow{r} P'$, we assume the theorem on P and P' , namely we assume $P \xrightarrow{r} P' \implies M_P \xrightarrow{R', M', v'} M_{P'}$. Notice that rule R' , used to derive the transition $M_P \xrightarrow{R', M', v'} M_{P'}$, belongs to the set of rules R_P which is, by the definition of \mathcal{I} , a subset of $R_{Q|P}$, namely $R_P \subseteq R_{Q|P}$. We must satisfy all the premises of the semantics of sSMSR in order to derive the correct transition. As regard the instantiation function σ , we can use the same instantiation function used to derive the transition of the inductive hypothesis. The state of the system, $M_1 \mid M_3$, is such that $M_1 \equiv M_P$ and $M_3 \equiv M_Q$ by definition of the function \mathcal{I} . In particular, let $\mathcal{I}(P, \epsilon) = (D_P, D'_P, E_P)$, $\mathcal{I}(Q, \epsilon) = (D_Q, D'_Q, E_Q)$, $\mathcal{I}(P', \epsilon) = (D_{P'}, D'_{P'}, E_{P'})$, $E_Q \cap E_P = \emptyset$ and $E_Q \cap E_{P'} = \emptyset$; by definition of \mathcal{I} we have that $\mathcal{I}(Q \mid P, \epsilon) = (D_Q \cup D_P, D'_Q \cup D'_P, E_Q \cup E_P)$ and that $\mathcal{I}(Q \mid P', \epsilon) = (D_Q \cup D_{P'}, D'_Q \cup D'_{P'}, E_Q \cup E_{P'})$. Furthermore, as $M_P \equiv M_1$ and $M_P \xrightarrow{R', M', v'} M_{P'}$, then $M_{P'} \equiv M_2$. As regards the constraints, let us assume an instantiation function σ' which extends the instantiation function σ used to satisfy the constraints in the derivation of the inductive hypothesis. Notice that there exist infinite substitutions σ' which satisfy the constraint in state $M_P \mid M_Q$. It is clear that it is possible to derive the transition of the semantics of sSMSR $M_{Q|P} \xrightarrow{R, M_P, v} M_2 \mid M_3$ where $M_2 \mid M_3 \equiv M_{P'} \mid M_Q \equiv M_{Q|P'}$.

Theorem 34 (Completeness) *Given a SPi process P and its sSMSR encoding (M_P, R_P) , it holds: $M_P \xrightarrow{R, M, v} M_{P'} \implies P \xrightarrow{r} P'$*

PROOF. Let us assume $M_P \equiv M_{P_1} \mid M_{P_3}$ and let us assume all the premises of the semantics rule of sSMSR. All the rules obtained by the encoding of a

SPi process P have as reactants a multiset pattern containing two sequence patterns, let us assume that the rule used to derive the sSMSR transition, $R : MP_1 \xrightarrow{k} MP_2$, is such that $MP_1\sigma \equiv SP_1\sigma \mid SP_2\sigma$ and $MP_2\sigma \equiv MP_{Q_1}\sigma \mid MP_{Q_2}\sigma \mid MP_{SP}\sigma$, by using the instantiation function σ assumed as premise. Say R models the communication of a process P_1 on the channel x , namely $R \in R_{P_1}^x$ and P'_1 and P''_1 are two processes of P_1 that can communicate on x . As regards R we have that SP_1 and SP_2 are the sequence patterns identifying the intermediate process descriptions of P'_1 and P''_1 computed by function \mathcal{I} ; furthermore, MP_{Q_1} and MP_{Q_2} are the union of sequence patterns identifying the encoding of the continuations of the communication actions. Finally, the multiset MP_{SP} denotes the fact that P'_1 and P''_1 could have a replication. We divide the proof by cases on the structure of M_{P_3} and of MP_{SP} .

- Let $M_{P_3} \equiv \epsilon$, we show that the sSMSR transition $M_{P_1} \xrightarrow{R, M, v} MP_{Q_1}\sigma \mid MP_{Q_2}\sigma \mid MP_{SP}\sigma$ models a communication inside the process encoded by the term M_{P_1} . In particular, the structure of such a process depends on the structure of the term MP_{SP} . We divide the proof by cases on the structure of MP_{SP} .
- (a) Let $MP_{SP} \equiv \epsilon$, then P'_1 and P''_1 have no replication action. The descriptions of the top-level components of P_1 would be, reflecting the structure of M_{P_1} , of the form

$$(SP_1, x\langle y \rangle.MP_{Q_1} + MP_{\Sigma}) \quad (SP_2, x(m).MP_{Q_2} + MP_{\Sigma'})$$

where MP_{Σ} and $MP_{\Sigma'}$ are the intermediate process descriptions of all the other possible actions of P_1 . Consequently, the structure of process P_1 is the following:

$$P_1 \equiv \nu C(x\langle y \rangle.Q_1 + \Sigma \mid x(m).Q_2 + \Sigma')$$

where νC is a sequence of restrictions for all the channels appearing in P_1 as the process is assumed to be closed. The fact that P_1 can communicate on x is due to the fact that $R \in R_{P_1}^x$. By the definition of the semantics of SPi it is possible to derive a transition by applying the rule for restricted processes once for each channel restriction appearing in C ; the derived transition is $P_1 \xrightarrow{rate(x)} Q_1 \mid Q_{2\{y/m\}}$. Notice that, by the structure of R , the multiset $MP_{Q_1}\sigma \mid MP_{Q_2}\sigma$ correctly denotes the encoding of $Q_1 \mid Q_{2\{y/m\}}$.

- (b) Let $MP_{SP} \equiv SP_1$, then P'_1 or P''_1 had a replication action. Let us assume that the replication appear inside process P'_1 and that P'_1 makes an output action. The descriptions of the top-level components of P_1 , reflecting the structure of M_{P_1} , are of the form

$$(SP_1, !x\langle y \rangle.MP_{Q_1}) \quad (SP_2, x(m).MP_{Q_2} + MP_{\Sigma})$$

where MP_{Σ} is the union of sequence patterns denoting the identifiers of all the other possible communications. Consequently, the structure of process

P_1 is the following:

$$P_1 \equiv \nu C(!x\langle y \rangle.Q_1 \mid x(m).Q_2 + \Sigma)$$

where νC is a sequence of restrictions for all the channels appearing in P_1 as the process is assumed to be closed. The fact that P_1 can communicate on x is due to the fact that $R \in R_P^x$. By the definition of the structural congruence relation on SPi processes it is possible to rewrite P_1 as $x\langle y \rangle.(Q_1 \mid !x\langle y \rangle.Q_1) \mid x(m).Q_2$. As in the case (a) of the proof it is possible, by applying the rule for restricted processes once for each channel restriction appearing in C , to derive the transition $P_1 \xrightarrow{rate(x)} Q_1 \mid !x\langle y \rangle.Q_1 \mid Q_2\{y/m\}$. Notice that, by the structure of R , the multiset $MP_{Q_1}\sigma \mid MP_{Q_2}\sigma \mid SP_1\sigma$ correctly denotes the encoding of $Q_1 \mid !x\langle y \rangle.Q_1 \mid Q_2\{y/m\}$.

- (c) Let $MP_{SP} \equiv SP_2$, then the proof is analogous to case (b) where the replication appearing in P_1' is an input action rather than an output one.
- (d) Let $MP_{SP} \equiv SP_1 \mid SP_2$, then the proof is a combination of both the cases (b) and (c) where both the processes have a replication action.

- Let $M_{P_3} \neq \epsilon$, then M_{P_3} is a term which represent the encoding of a SPi process, say Q . It is possible to prove, as in the case of $M_{P_3} \equiv \epsilon$, that the process described by the term M_{P_1} , rewritten by means of M_{P_2} , models a SPi communication. Thus it is possible to derive the SPi transition $P_1 \xrightarrow{r} P_2$ where P_1 and P_2 are the processes described by $MP_1\sigma$. With respect to the semantics of SPi it is possible to derive the transition $P_1 \mid Q \xrightarrow{r} P_2 \mid Q$.

The proof of Theorem 21 is a direct consequence of Theorems 33 and 34.

C Proof of Theorem 22

PROOF. Let $P \equiv P_1 \mid \dots \mid P_n$ such that the processes P_i with $i = 1, \dots, n$ are of the form of a summation, $P_i \equiv \Sigma_i$. By definition we know that $In_x(P) = \sum_{i=1}^n In_x(P_i)$ and that $Out_x(P) = \sum_{i=1}^n Out_x(P_i)$. All the possible communications of process P on channel x can be computed as:

$$\begin{aligned} In_x(P) \cdot Out_x(P) &= \left(\sum_{i=1}^n In_x(P_i) \right) \cdot \left(\sum_{i=1}^n Out_x(P_i) \right) \\ &= \left(\sum_{\substack{i,j=1 \\ i \neq j}}^n In_x(P_i) \cdot Out_x(P_j) \right) + \sum_{i=1}^n (In_x(P_i) \cdot Out_x(P_i)) \end{aligned}$$

The term $Mix_x(P)$, used to compute all the wrong communications with respect to the semantics of SPi, can be defined as

$$Mix_x(P) = \sum_{i=1}^n (In_x(P_i) \cdot Out_x(P_i))$$

The communications computed by $Mix_x(P)$ are wrong because are computed within the actions of the same process, P_i with $i = 1, \dots, n$. The channel activity of process P on channel x can be computed as

$$\begin{aligned} Act_x(P) &= (In_x(P) \cdot Out_x(P)) - Mix_x(P) \\ &= \sum_{i,j=1, i \neq j}^n In_x(P_i) \cdot Out_x(P_j) \end{aligned}$$

As we formally defined the channel activity of a SPi process, we can now prove the theorem by induction on the size of P .

(*Base case*) Let $P \equiv P_1 \mid P_2$ where P_1 and P_2 are processes in the form of a summation, $P_i \equiv \Sigma_i$. In particular each summation Σ_i can be either a summation of actions or a single replication; in this proof we assume that each summation is a summation of actions. The proof in the case of the replication, is a particular case of this one. By definition the channel activity of P is

$$Act_x(P) = In_x(P_1) \cdot Out_x(P_2) + In_x(P_2) \cdot Out_x(P_1)$$

Let the summations be the following

$$\begin{aligned} \Sigma_1 &\equiv \pi_1^{1,i} \cdot P_1 1 + \dots + \pi_{n1}^{1,i} \cdot P_{n1} 1 + \pi_1^{1,o} \cdot P_{n^i+1} 1 + \dots + \pi_{m1}^{1,o} \cdot P_{n1+m1} 1 + \Sigma \\ \Sigma_2 &\equiv \pi_2^{2,i} \cdot P_1 1 + \dots + \pi_{n1}^{2,i} \cdot P_{n1} 1 + \pi_1^{2,o} \cdot P_{n^i+1} 1 + \dots + \pi_{m1}^{2,o} \cdot P_{n1+m1} 1 + \Sigma' \end{aligned}$$

where $\pi_j^{1,i} \cdot P_j 1$ with $j = 1, \dots, n1$ are the $n1$ input actions on channel x of process P_1 , $\pi_k^{1,o} \cdot P_{n1+k} 1$ with $k = 1, \dots, m1$ are the $m1$ output actions on channel x of process P_1 and, finally, Σ denotes all the other actions, of process P_1 , on channels different from x . Analogously, $\pi_j^{2,i} \cdot P_j 2$ with $j = 1, \dots, n2$ are the $n2$ input actions on channel x of process P_2 , $\pi_k^{2,o} \cdot P_{n2+k} 1$ with $k = 1, \dots, m2$ are the $m2$ output actions on channel x of process P_2 and, finally, Σ' denotes all the other actions, of process P_2 , on channels different from x . Trivially, the channel activity of P is equal to $n^1 \cdot m2 + n2 \cdot m1$. Let us denote with $\overline{R}_P^x \subseteq R_P^x$ the set of rules describing all the possible communications on x in P between the top-level components of the process. It is trivial to notice that $ExitRate(M_P, R_P^x) = ExitRate(M_P, \overline{R}_P^x)$. Such a set is constructed, with respect to the encoding of SPi, as follows

$$\begin{aligned} \overline{R}_P^x &= \{SP_{P_1} \mid SP_{P_2} \xrightarrow{rate(x)} SP_{P_{1i}} \mid SP_{P_{1j}} \ \forall i = 1, \dots, n1 \ \wedge \ j = n2 + 1, \dots, n2 + m2\} \\ &\cup \{SP_{P_1} \mid SP_{P_2} \xrightarrow{rate(x)} SP_{P_{1i}} \mid SP_{P_{1j}} \ \forall i = m1 + 1, \dots, m1 + n1 \ \wedge \ j = 1, \dots, n2\} \end{aligned}$$

where SP_{P_1} , SP_{P_2} , $SP_{P_{1j}}$ and $SP_{P_{2i}}$ are the identifiers of the processes P_1 , P_2 , of the j -th process of P_1 and of the i -th process of P_2 , respectively. Such a set of rules represent all the possible communications in P on channel x of the top-level components. Furthermore, all the rules are different because, by definition of the encoding, all the identifiers are different. This yields the fact that the set of transitions that can be derived by the state M_P contains exactly one transition with rate $rate(x)$ for each rule, namely $n^1 \cdot m2 + n2 \cdot m1$ transitions with rate $rate(x)$.

(*Induction case*) Let $P \equiv P_1 \mid \dots \mid P_n \mid P_{n+1} \equiv \bar{P} \mid P_{n+1}$ with $n \geq 2$. In particular, as in the base case of the proof, each summation Σ_i can be either a summation of actions or a single replication; also in this proof we assume that each summation is a summation of actions being the proof in the case of the replication a particular case of this one. By definition the channel activity of P is

$$\begin{aligned}
Act_x(P) &= (In_x(P) \cdot Out_x(P)) - Mix_x(P) \\
&= \sum_{i=1}^{n+1} \sum_{j=1, i \neq j}^{n+1} In_x(P_i) \cdot Out_x(P_j) \\
&= \left(\sum_{i=1}^n \sum_{j=1, i \neq j}^n In_x(P_i) \cdot Out_x(P_j) \right) \\
&\quad + \sum_{i=1}^n In_x(P_{n+1}) \cdot Out_x(P_i) + \sum_{i=1}^n In_x(P_i) \cdot Out_x(P_{n+1}) \\
&= Act_x(\bar{P}) + \sum_{i=1}^n In_x(P_{n+1}) \cdot Out_x(P_i) + \sum_{i=1}^n In_x(P_i) \cdot Out_x(P_{n+1})
\end{aligned}$$

We assume the induction hypothesis on \bar{P} , namely we assume

$$Act_x(\bar{P}) = ExitRate(M_{\bar{P}}, R_{\bar{P}}^x) = ExitRate(M_{\bar{P}}, \bar{R}_{\bar{P}}^x)$$

We recall that, by the assumptions on the encodable SPi processes, each name of the encoded processes is different; we now divide the proof by cases on P_{n+1} :

- (a) If P_{n+1} does not contain any communication on channel x between any top-level component then, by definition of the encoding, $R_P^x = R_{\bar{P}}^x$. The rate computed with respect to the set of transitions that can be derived from state M_P is, trivially, the same that can be computed with respect to the transitions that can be derived by state $M_{\bar{P}}$ because P_{n+1} does not communicate with any process of \bar{P} on channel x . Formally, $Act_x(P) = Act_x(\bar{P})$ and the proof follows by the induction hypothesis.
- (b) Let P_{n+1} contains a communication on channel x between any top-level component then, by definition of the encoding, $R_P^x \supset R_{\bar{P}}^x$. Let us assume the form of each process P_j of P to be the following

$$\Sigma_j \equiv \pi_1^{j,i} \cdot P_1^j + \dots + \pi_{n_j}^{j,i} \cdot P_{n_j}^j + \pi_1^{j,o} \cdot P_{n_j+1}^j + \dots + \pi_{m_j}^{j,o} \cdot P_{n_j+m_j}^j + \Sigma_j^j$$

where the meaning of Σ_j is the same of the base case of the proof. By definition $In_x(P_i) = n^i$ and $Out_x(P_i) = m^i$ with $i = 1, \dots, n+1$, then the channel activity of x can be computed as

$$Act_x(P) = Act_x(\overline{P}) + \sum_{i=1}^n (n^{n+1} \cdot m^i) + \sum_{i=1}^n (n^i \cdot m^{n+1})$$

Let us denote with $\overline{R}_P^{x,i}$ the set of rules describing all the possible communications on x between any top-level component of the processes P_i and P_{n+1} . Such a set can be built similarly to what done for the set of rewriting rules in the base case of the proof. It holds that

$$\overline{R}_P^x = \left(\bigcup_{i=1}^n \overline{R}_P^{x,i} \right) \cup \overline{R}_{\overline{P}}^x$$

and, by the assumption on the encodable SPi processes, it holds that $(\bigcup_{i=1}^n \overline{R}_P^{x,i}) \cap \overline{R}_{\overline{P}}^x = \emptyset$. This because no possible communications of P_{n+1} could have been modeled by rules in $\overline{R}_{\overline{P}}^x$ as the encoding of P_{n+1} is different from the encoding of any other process in \overline{P} . We can compute the exit rate of state M_P with rules \overline{R}_P^x as follows:

$$\begin{aligned} ExitRate(M_P, \overline{R}_P^x) &= \sum_{\{r \mid M_P \xrightarrow{R, M', r} M_{P'} \wedge R \in \overline{R}_P^x\}} r \\ &= \sum_{\{r \mid M_P \xrightarrow{R, M', r} M_{P'} \wedge R \in \overline{R}_{\overline{P}}^x\}} r \\ &\quad + \sum_{\{r \mid M_P \xrightarrow{R, M', r} M_{P'} \wedge R \in \bigcup_{i=1}^n \overline{R}_P^{x,i}\}} r \end{aligned}$$

Notice that $ExitRate(M_{\overline{P}}, \overline{R}_{\overline{P}}^x) = \sum_{\{r \mid M_{\overline{P}} \xrightarrow{R, M', r} M_{P'} \wedge R \in \overline{R}_{\overline{P}}^x\}} r$, because, $M_{P_{n+1}} \not\equiv M_{P_i}$ by the assumptions on SPi processes. Then the rate computed for all the transitions that can be derived from state $M_{\overline{P}}$ is the same that can be computed for those derived from state M_P by applying rules of $\overline{R}_{\overline{P}}^x$. The exit rate can be then computed as follows:

$$ExitRate(M_P, \overline{R}_P^x) = ExitRate(M_{\overline{P}}, \overline{R}_{\overline{P}}^x) + \sum_{\{r \mid M_P \xrightarrow{R, M', r} M_{P'} \wedge R \in \bigcup_{i=1}^n \overline{R}_P^{x,i}\}} r$$

Finally, by the same considerations made in the base case of the proof and by the assumption on SPi processes, it holds that $\sum_{\{r \mid M_P \xrightarrow{R, M', r} M_{P'} \wedge R \in \bigcup_{i=1}^n \overline{R}_P^{x,i}\}} r = \sum_{i=1}^n (n^{n+1} \cdot m^i) + \sum_{i=1}^n (n^i \cdot m^{n+1})$. Note that such a set of transitions models all the possible communications of P_{n+1} with any other process of \overline{P} .