A Stochastic Pi Calculus for Concurrent Objects

Céline Kuttler, Cédric Lhoussaine, Joachim Niehren

▶ **To cite this version:**

Céline Kuttler, Cédric Lhoussaine, Joachim Niehren. A Stochastic Pi Calculus for Concurrent Objects. [Technical Report] RR-6076, INRIA. 2006, pp.28. inria-00121104v2

HAL Id: inria-00121104

https://inria.hal.science/inria-00121104v2

Submitted on 22 Dec 2006 (v2), last revised 1 May 2007 (v3)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A Stochastic Pi Calculus for Concurrent Objects

Céline Kuttler — Cédric Lhoussaine — Joachim Niehren

N° 6076

December 2006

Thèmes SYM et BIO

ISSN 0249-6399 ISRN INRIA/RR--6076--FR+ENG

A Stochastic Pi Calculus for Concurrent Objects

Céline Kuttler [*] , Cédric Lhoussaine[†], Joachim Niehren[‡]

Thèmes SYM et BIO — Systèmes symboliques et Systèmes biologiques
Projet Mostrare

Rapport de recherche n° 6076 — December 2006 — 28 pages

Abstract: We present SpiCO, a new modeling and simulation language for system biology, based on the stochastic π-calculus. SpiCO supports higher level modeling via multi-profile concurrent objects with static inheritance. We present a semantics for SpiCO in terms of continuous time Markov chains, and show how to compile SpiCO back into the biochemical stochastic π-calculus while preserving semantics.

Key-words: programming languages, systems biology, modeling, simulation

[*] The Microsoft Research - University of Trento Centre for Computational and Systems Biology, Italy
[†] University of Lille 1, LIFL, Lille, France
[‡] INRIA Futurs, Lille, France, Mostrare project

Unité de recherche INRIA Futurs
Parc Club Orsay Université, ZAC des Vignes,
4, rue Jacques Monod, 91893 ORSAY Cedex (France)
Téléphone : +33 1 72 92 59 00 —Télécopie : +33 1 60 19 66 08

Un pi-calcul stochastique pour objects concurrents

Résumé : Nous présentons SpiCO, un nouveau langage de modélisation et de simulation de systèmes biologiques fondé sur le π-calcul stochastique. SpiCO permet une modélisation de haut niveau grâce à des objets concurrents à profiles multiples et un mécanisme d'héritage statique. Nous présentons une sémantique pour SpiCO en termes de chaînes de Markov à temps continu, et nous montrons comment compiler SpiCO dans le π-calcul stochastique biochimique tout en préservant la sémantique.

Mots-clés : langage de programmation, biologie systémique, modélisation, simulation

1 Introduction

Modeling and simulation of the dynamic interactions of molecular actors is a prime objective of systems biology. Discrete event based modeling approaches can capture decisive details of cellular control in the regime of small numbers of molecular actors. Deterministic approaches in contrast offer benefits when modeling the evolution of large populations, which has a longer tradition.

Regev and Shapiro [RS02] proposed the stochastic π-calculus as a modeling language for systems biology. This is Priami's [Pri95] refinement of the synchronous π-calculus [MPW92] by a notion of time. Processes of the π-calculus are seen as chemical solutions, in which molecules interact concurrently. The reaction speed is controled by stochastic parameters, as for earlier stochastic process algebras [Hil95], defining exponential distributions of waiting times. They turn process expressions into continuous time Markov chains (CTMCs). Process execution based on Gillespie's algorithm yields stochastic simulation [Gil76].

Two simulation engines for the (biochemical) stochastic π-calculus are available so far, SPiM [PC06] and BioSpi [PRSS01]. Both have been applied in a number of case studies of small to medium size, bounded by a dozen pages of program code [KN06, KNP+06, LPQ+04]. Alternative modeling languages as BioCham [CRFS05] or SBML [HFS+03] can directly specify systems of chemical reaction rules. This approach is simpler, yet seems less expressive with respect to concurrent control, i.e. intricate conditions for rule application.

From the *modeling* perspective in systems biology, the minimality of the π-calculus is sometimes unfortunate. Concurrent control soon requires sophisticated protocols (see [KN06]), non-trivial to both design and understand. Such protocols are at a low level and must be adapted upon model extension. This constitutes a major obstacle to up-scaling models.

In this paper, we present SPiCO, a new modeling language for systems biology extending on a *stochastic π-calculus for concurrent objects*. Indeed, SPiCO has been developed concomitantly with modeling case studies [Kut06a, Kut06b]. The main insight behind SPiCO is that concurrent objects (as in programming languages) appropriately represent interacting molecules for systems biology. Object *interfaces* avoid communication protocols, while object *inheritance* renders models more extensible. The technical contributions can be summarized as follows:

1. We present the core of SPiCO, a new stochastic π-calculus with input patterns. Input pattern originate from TyCO, a distributed programming language [PML+03, VT93] for typed concurrent objects in the asynchronous π-calculus. Stochastic rates are now assigned to pairs of channel and function names.

2. We define a *stochastic semantics* which assigns CTMCs to SPiCO's process expressions, while accounting for timed and instantaneous reactions. Technically, this is the most difficult part of the paper. Previous semantics for the stochastic π-calculus do not define CTMCs at all [PRSS01, PC06] or disregard instantaneous reactions [Pri95] essential for expressiveness and modeling. Experiments with SPiM confirm a correct treatment in implementations nevertheless.

3. We identify multi-profile objects with expressions in the core of SpiCO. Each profile comes with its own interface, similarly as TyCO's non-uniform objects [RV00]. Beyond these, multi-profile objects allow choice with mixed input and output on possibly different channels and synchronous communication.

4. We define a notion of inheritance for multi-profile objects that is compiled into the core of SpiCO. We present a module system for SpiCO providing syntax for definitions of objects with inheritance.

5. We discuss a programming technique by which to model mutual exclusion of molecular binding at overlapping sites, as frequently in systems biology. Its essence lies in escaping inconsistent intermediate states by instantaneous reactions, that are applied before timed ones. This solves tedious *atomicity* problems, without introducing transactions (as [CP06]).

6. We show how to encode SpiCO into the biochemical stochastic π-calculus, so that we can run SpiCO programs in SPiM or BioSpi. The main problem here is to encode input patterns, such that the stochastic semantics is preserved.

Other languages for diverse modeling tasks in systems biology have been proposed recently. Beta binders [PQ05] derive from the π-calculus, enable interactions by type coincidence rather than channel name equality. At the same time, beta binders can express some spatial aspects at membranes, an aspect approached by ambient style calculi [Car05, RPS$^+$04]. An ad hoc predecessor of SpiCO was presented in [DK05], introducing the idea of multi-profile objects for systems biology. At this time, however, it was not yet clear how to express such objects in a more conservative language with proper syntax and semantics.

Outline. The core of SpiCO is introduced in Section 2 and illustrated for modeling molecular binding at overlapping sites in Section 3. SpiCO's multi-profile objects with inheritance are discussed in Section 4. CTMCs for chemical reactions in Section 5 motivate the stochastic semantics of SpiCO in Section 6. In Section 7, we show how to encode input patterns by a naming discipline.

2 A Stochastic Pi-calculus with Input Patterns

We present the core of SpiCO (Core SpiCO), which is a new stochastic π-calculus with input patterns. This latter language element was previously proposed for typed concurrent objects (TyCO) in the asynchronous π-calculus by Vasconcelos and Tokoro [PML$^+$03, VT93]. Input patterns are motivated by pattern matching in functional programming languages of the ML family, but closely tied to communication in a smart manner. A process can only receive tuples for which it provides a matching input pattern.

The vocabulary consists of an infinite set of *channel names* $\mathcal{N} = \{x, y, z, \ldots\}$, a set of *process names* A, and a set of *function names* $f \in \mathcal{F}$. It fixes arities for process and

Processes	P	::=	$P_1 \mid P_2$	parallel composition
		|	**new** $x{:}\rho.\ P$	channel creation
		|	$C_1 + \ldots + C_n$	sum ($n \geq 0$)
		|	$A(\tilde{x})$	application
Guarded processes	C	::=	$x?f(\tilde{y}).P$	pattern input
		|	$x!f(\tilde{y}).P$	tuple output
Definitions	D	::=	$A(\tilde{y}) \triangleq P$	

Table 1: Syntax of Core SPICO

function names. We write A/n or f/n for a symbol of arity $n \geq 0$. In order to account for *stochastic rates*, the vocabulary comprises functions $\rho : \mathscr{F} \rightarrow]0, \infty]$ to define *stochastic rates* for every channel. If some function ρ is assigned to x then $\rho(f)$ is the rate of the pair (x, f).

The syntax of Core SPICO is defined in Table 1. We write $\tilde{x}$ for finite, possibly empty sequences of channels $x_1, \ldots, x_n$ where $n \geq 0$. Whenever we use tuples $f(\tilde{x})$ or terms $A(\tilde{x})$ we assume that the number of arguments (the length of $\tilde{x}$) is equal to the respective arity of f or A. Process expressions are ranged over by P. The only atomic expression (that cannot be decomposed into others) is the guarded choice of length $n = 0$ that we write as $\mathbf{0}$. Expressions $P_1|P_2$ denote the parallel composition of processes P_1 and P_2. A term **new** $x{:}\rho.\ P$ describes the introduction of a new channel x scoping over P; the rate function ρ fixes stochastic rates $\rho(f)$ for all pairs (x, f) where $f \in \mathscr{F}$. We can omit rate functions ρ in the declaration of a channel x if all reactions on x are instantaneous, i.e. $\rho(f) = \infty$ for all $f \in \mathscr{F}$. An expression $A(\tilde{x})$ applies the definition of a parametric process A with actual parameters $\tilde{x}$.

A sum of guarded processes $C_1 + \ldots + C_n$ offers a choice between $n \geq 0$ communication alternatives $C_1, \ldots, C_n$. A guarded input $x?f(\tilde{y})$ describes a communication act, ready to receive a tuple that is constructed by f over x. The channels $\tilde{y}$ in input guards serve as pattern variables; these are bound variables that are replaced by the channels received as input. An output guarded process $x!f(\tilde{y}).P$ describes a communication act willing to send tuple $f(\tilde{y})$ over channel x and continue as P.

A definition of parametric process has the form $A(\tilde{x}) \triangleq P$ where A is a process name with $\tilde{x}$ as formal parameters (that is a sequence of bound channels). Note that, for modeling convenience, we allow P to contain free channel names besides the parameters in $\tilde{x}$. The set of *free channel names* for processes P and guarded processes C are denoted by $fv(P)$ and $fv(C)$ respectively. There are three scope baring constructs: new binder **new** $x{:}\rho.\ P$, input patterns $\_?f(\tilde{x}).P$, and definitions $A(\tilde{x}) \triangleq P$. Formally, these sets are defined by induction on the structure of such expressions in Table 2.

We define the stochastic-free operational semantics of the π-calculus in terms of a binary relation between expressions, the so called (one step) reduction. We will later refine it to a

$$
\begin{array}{l}
\mathit{fv}(x?f(\tilde{y}).P) = \{x\} \cup (\mathit{fv}(P) - \{\tilde{y}\}) \\
\mathit{fv}(P_1 \mid P_2) = \mathit{fv}(P_1) \cup \mathit{fv}(P_2) \\
\mathit{fv}(x!f(\tilde{y}).P) = \{x\} \cup \mathit{fv}(P) \cup \{\tilde{y}\} \\
\mathit{fv}(\mathbf{new}\ x{:}\rho.\ P) = \mathit{fv}(P) - \{x\} \\
\mathit{fv}(C_1 + \ldots + C_n) = \mathit{fv}(C_1) \cup \ldots \cup \mathit{fv}(C_n)
\end{array}
$$

Table 2: Free variables

$$
\begin{array}{ll}
(P_1|P_2)|P_3 \equiv P_1|(P_2|P_3) & P_1|P_2 \equiv P_2|P_1 \\
\ldots + C_1 + C_2 + \ldots \equiv \ldots + C_2 + C_1 + \ldots & P|\mathbf{0} \equiv P \\
\mathbf{new}\ x{:}\rho.\ (P_1|P_2) \equiv P_1|\ \mathbf{new}\ x{:}\rho.\ P_2 \quad \text{if } x \notin \mathit{fv}(P_1) & P_1 \equiv P_2 \text{ if } P_1 \equiv_\alpha P_2 \\
\mathbf{new}\ x_1{:}\rho_1.\ \mathbf{new}\ x_2{:}\rho_2.\ P \equiv \mathbf{new}\ x_2{:}\rho_2.\ \mathbf{new}\ x_1{:}\rho_1.\ P & \text{if } x_1 \neq x_2
\end{array}
$$

Table 3: Axioms of the structural congruence

ternary relation adding stochastic labels. The reduction relation is closed under the usual structural congruence between expressions.

The *structural congruence* is the smallest relation induced by the axioms in Table 3. It identifies expressions modulo associativity and commutativity of parallel composition, i.e. the order in $P_1|\ldots|P_n$ does not matter, order independence of alternatives in sums, and scope extrusion. We also assume congruence of α-convertible processes, i.e. that can be obtained from another by renaming bound names, without capturing free names. (See for instance [Hon92] for a formal definition of α-conversion.)

Table 4 defines the *reduction relation*. The first axiom tells how to interpret choices; it comprises channel communication and pattern matching. It applies to two complementary matching alternatives in parallel choices, an output alternative $x!f(\tilde{y}).P_1$ willing to send a term $f(\tilde{y})$ and an input pattern $x?f(\tilde{z}).P_2$ on the same channel x; this pattern matches in that it is built using the same function symbol f. Reduction cancels all other alternatives, substitutes the pattern's variables $\tilde{z}$ by the received channels $\tilde{y}$ in the continuation P_2 of the input, and reduces the result in parallel with the continuation of the output P_1.

Only matching tuples can be received over a channel. Other sending attempts suspend until a suitable input pattern becomes available. This fact proves extremely useful for concurrent modeling. Upon reception, tuples are immediately decomposed, in contrast to the π-calculus with data terms [BPV05].

The application axiom unfolds one of the definitions of the parametric processes in a given set Δ. An application $A(\tilde{y})$ reduces in one step to definition P in which the formal parameters $\tilde{y}$ were replaced by the actual parameters $\tilde{x}$. Parametric definitions may be recursive, e.g. A may occur in P. Reduction can be applied in arbitrary contexts, however not under choices or in definitions.

Communication, choice, and pattern matching:

$$x!f(\tilde{y}).P_1 + \ldots \mid \; x?f(\tilde{z}).P_2 + \ldots \quad \rightarrow \quad P_1 \mid P_2[\tilde{z} \mapsto \tilde{y}] \qquad \text{if } \tilde{z} \text{ free for } \tilde{y} \text{ in } P_2$$

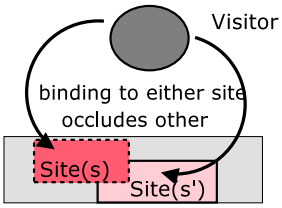
Application of definitions:

$$A(\tilde{x}) \quad \rightarrow \quad P[\tilde{y} \mapsto \tilde{x}] \qquad \text{if } A(\tilde{y}) \triangleq P \text{ in } \Delta, \text{ and } \tilde{y} \text{ free for } \tilde{x} \text{ in } P$$

Context and congruence closure:

$$\frac{P \rightarrow P'}{\textbf{new } c{:}\rho.\ P \rightarrow \textbf{new } c{:}\rho.\ P'} \qquad \frac{P \rightarrow P'}{P \mid Q \rightarrow P' \mid Q} \qquad \frac{P \equiv P' \quad P' \rightarrow Q' \quad Q \equiv Q'}{P \rightarrow Q}$$

Table 4: Reduction relation for a finite set of definitions Δ



(a) *unbound state*

Site(s,s')	Site(s',s)	interaction	delay
free	free	s.bind()	timed
bound	free	s'.block()	immediate
bound	blocked	s.unbind()	timed
free	blocked	s'.unblock()	immediate
free	free		

(b) *Sample execution*

Figure 1: Overlapping sites s and s'

The syntax of the biochemical stochastic π-calculus is the same as ours except for function names. We can express polyadic input and output by using dummy function names UNIT_i for all arities $i \geq 0$ in following shortcuts for all sequences $\tilde{y}$ of channel names of length i:

$$x?(\tilde{y}).P \quad =_{\text{def}} \quad x?(\text{UNIT}_i(\tilde{y})).P \qquad \text{and} \qquad x!(\tilde{y}).P \quad =_{\text{def}} \quad x!(\text{UNIT}_i(\tilde{y})).P$$

3 Molecular Binding at Overlapping Sites

We introduce the modeling power of SpiCO by examples that occur frequently in systems biology. We consider overlapping sites that allow for a single visitor at a time (overlapping semaphores). This situation frequently appears in systems biology when modeling molecular binding.

We restrict ourselves to two overlapping sites s and s' as illustrated in Figure 1(a). Each of them can be either free, bound, or blocked. Only free sites can be bound by visitors. In this case, the other overlapping site must become blocked. We model sites as multi-profile object with three profiles Site_free, Site_bound, and Site_blocked. Figure 2 presents their definitions by sums in the π-calculus. Beside of its own identity me, a site is parametrized by the identity of the other overlapping site. The defining sums specify interfaces for profiles,

```
module 'overlapping sites'
export Site with bind/0, unbind/0
define
   Site(me,other) ≜ Site_free(me,other)
   Site_free(me,other) ≜
      me?bind().Site_bound(me,other) // timed
    + me?block().Site_blocked(me,other)   // immediate
    + other!unblock().Site_free(me,other) // immediate

   Site_bound(me,other) ≜
      me?unbind().Site_free(me,other) // timed
    + other!block().Site_bound(me,other)   // immediate

   Site_blocked(me,other) ≜
      me?unblock().Site_free(me,other)   // immediate
```

Figure 2: Overlapping sites module

i.e., which functions are offered or applied and on which channels. Profile Site_free for instance, offers functions bind and block by which it can become bound or blocked, and can apply function unblock of the other site.

Multi-profile objects yield an elegant solution to express semaphores (sites with at most one visitor). A visitor can only bind to free sites since no other profile offers the bind function. This exploits the clever coupling between pattern matching and synchronization by input patterns.

The most tedious aspect of overlapping sites is to keep states consistent. Whenever a site gets bound, its overlapping peer must **immediately** become blocked, i.e. without any elapse of simulated time. The actor Site_bound(me,other) enforces this by applying function block on its peer. The stochastic rate of this function needs must thus be ∞. This technique works only if immediate transitions have priority over time-consuming ones, and under the assumption that function block is immediate. Highest priority of immediate transitions is guaranteed by the stochastic semantics to come (see rule (SUM) in Table 5). This way, we solve a tedious atomicity problems while avoiding heavier extensions of the π-calculus by transactions [CP06].

One possible sequence of state changes is given in Figure 1(b). Initially, we assume a parallel composition of two free sites and two free visitors. The first parameter of Site_free refers to its identity and the second to its peer's:

	Site_free(s,s′)	| Site_free(s′,s)	| Visitor_free	| Visitor_free
$\rightarrow$	Site_bound(s,s′)	| Site_free(s′,s)	| Visitor_free	| Visitor_at(s)
$\not\rightarrow$	Site_bound(s,s′)	| Site_bound(s′,s)	| Visitor_at(s)	| Visitor_at(s′)

The first reduction step is an application of function bind of s by the second Visitor_free defined in Figure 3, which consumes time. Now Site_free(s',s) has a potential choice between

```
module 'visitors for sites s or s' '
public s s'
export Visitor
define
    Visitor() ≙ Visitor_free()
    Visitor_free()≙ s!bind().Visitor_at(s)
                  + s'!bind().Visitor_at(s')
    Visitor_at(site) ≙ site!unbind().Visitor_free()
```

Figure 3: Visitors module for sites s and s'

a time consuming transition where function bind of s' is applied by the first Visitor_free, and an immediate transition applying function block of s' by Site_bound(s,s'). Priority is given to immediate transitions, so only the latter function can be applied.

$$\xrightarrow{\infty} \mathsf{Site\_bound(s,s')} \mid \mathsf{Site\_blocked(s',s)} \mid \mathsf{Visitor\_at(s)} \mid \mathsf{Visitor\_free}$$

Thereby, it becomes impossible to enter into an erroneous configuration in which both Sites are bound.

4 Multi-profile Objects with Inheritance

The full SpiCO language features multi-profile objects with static inheritance. In this section, We define these concepts formally and show how to compile them to Core SpiCO.

SpiCO supports the paradigm of "molecules as concurrent objects" a refinement of the paradigm "molecules as processes" by Regev and Shapiro. *Object classes* correspond to species of molecules. A class of a *multi-profile object* is a set of definitions by sums, each of which defines a profile.

$$\begin{aligned} \mathsf{Obj\_p}_1(\tilde{\mathsf{x}}_1) &\triangleq \mathsf{C}^1_1 + \ldots + \mathsf{C}^1_{n_1} \\ &\ldots \\ \mathsf{Obj\_p}_m(\tilde{\mathsf{x}}_m) &\triangleq \mathsf{C}^m_1 + \ldots + \mathsf{C}^m_{n_m} \end{aligned}$$

A major advantage of object-orientation for systems biological is model extensibility by object inheritance. Numerous examples are elaborated in [Kut06a]. A simpler case is given in Figure 4. This is a promoter, a Dna region controlling transcription initiation, which overlaps with an operator region. A promoter is thus like an overlapping site, except that it can initiate transcription when bound by a polymerase. This new functionality is added by inheritance. We next define inheritance for multi-profile objects. We extend class Obj to Obj2 as follows:

```
Obj2 extends Obj
Obj2_p1(z̃1) extended by C^1_{k1+1} + ... + C^1_{l1}
    ...
Obj2_pn(z̃n) extended by C^n_{kn+1} + ... + C^n_{ln}
```

```
module 'repressible promoter'
   import Site from 'overlapping sites'
export
   Promoter extends Site by initiate/0
define
   Promoter_bound(me,other) extended by
     me?initiate().Promoter_free(me,other)
```

Figure 4: Promoters inherit from overlapping sites

This specification with inheritance can be compile into definitions of Core SpiCO:

$\mathsf{Obj2\_p_1}(\tilde{z_1}) \triangleq C_1^1 + \ldots + C_{l_1}^1\ [\mathsf{Obj} \mapsto \mathsf{Obj2}]$

$\ldots$

$\mathsf{Obj2\_p_n}(\tilde{z_n}) \triangleq C_1^n + \ldots + C_{l_n}^n\ [\mathsf{Obj} \mapsto \mathsf{Obj2}]$

The substitution renames all recursive calls to profiles $\mathsf{Obj\_p_i}$ into recursive calls to $\mathsf{Obj2\_p_i}$ for $1 \leq i \leq n$.

SpiCO provides a module system for grouping sets of definitions together so that they can be extended by multiple inheritance. Modules import definitions from others as usual. Such module dependencies can be resolved statically, as long as they remain acyclic, which SpiCO assumes. The details of the module systems are out of the scope of this paper.

5 Markov Chains for Chemical Reactions

The stochastic semantics of our π-calculus is guided by the analogy to continuous time Markov chains (CTMCs) for chemical reactions.

We first recall CTMCs with countably infinite state spaces. We assume a countable set S called the *state space*. A *continuous time stochastic process* with states $q \in S$ is a family $\{X_t \mid t \in \mathbb{R}^+\}$ of random variables with values in S. These define probabilities $Pr(X_t \in S')$ for all subsets $S' \subseteq S$, i.e. the probability that the process is in some state of S' at time t. A *continuous time Markov chain (CTMC)* is a continuous time stochastic process, with memoryless sojourn times for all states. More formally, a CTMC over S is a continuous time stochastic process $\{X_t \mid t \in \mathbb{R}^+\}$ with states in S, that satisfies the Markov property, i.e. for all $q_0, \ldots, q_{n+1} \in S$ and all time points $0 \leq t_0 < \ldots < t_{n+1}$:

$$Pr(X_{t_{n+1}} = q_{n+1} \mid X_{t_n} = q_n, \ldots, X_{t_0} = q_0) = Pr(X_{t_{n+1}} = q_{n+1} \mid X_{t_n} = q_n)$$

The probabilistic behavior of a CTMC is determined by the distribution of its initial states (at time 0) and its *transition rates*. The transition rate r from state q to state q' is a value that "scales how the (one step) transition probability between q and q' increases with time" [Her02]. We write $q \xrightarrow{r} q'$ in this case. For simplicity, we consider CTMCs with a single initial state. These can be identified with a Markovian transition system $(S, (\xrightarrow{r})_{r \in R^+}, q_0)$ where $q_0 \in S$ is the initial state and $\xrightarrow{r} \subseteq S \times S$ are transition relations for all $r \in \mathbb{R}^+$, such

that for all $q, q' \in S$ there exists at most one $r \in R^+$ satisfying $q \xrightarrow{r} q'$.

The stochastic time evolution of a CTMC can be computed by Gillespie's *first reaction method* (1976) [Gil76] if each state permits only a finite number of transitions, as we assume in the sequel. At time 0 the process starts in state q_0. Suppose that the process has moved to state q at time point t and let $\{q \xrightarrow{r_i} q_i\}_i$ be all (finitely many) transitions starting in q. Draw delays $t_i > 0$ for all i from an exponential distribution with rate r_i. Draw with equal probability some j, with minimal t_j. Move to state q_j at time point $t + t_j$.

Gillespie's direct method equivalently determines the stochastic behavior of a CTMC [Gil76]. In state q at time t it first computes the delay until the next transition (called *sojourn time*), by drawing a number from the exponential distribution with rate $\downarrow s =_{\text{def}} \sum_{q \xrightarrow{r_i} q_i} r_i$. Second, the state q_j to go to is drawn with probability $Pr(q \to q_j) =_{\text{def}} r_j / \sum_{q \xrightarrow{r'} q'} r'$ if $q \xrightarrow{r_j} q_j$ and 0 otherwise.

We next illustrate CTMCs for systems of chemical reaction rules. We start from a set of chemical species X, Y, Z and a set of chemical reaction rules of the following form, where $r \in \mathbb{R}^+$, reserving the symbol $+$ for choice:

$$X \mid Y \xrightarrow{r} Z_1 \mid \ldots \mid Z_k$$

Chemical solutions P are multisets of species, where each occurrence in the multiset represents a molecule of the species. Chemical rules as above apply as follows to a chemical solution P. Each pair of molecules of species X and Y can interact at rate r, yielding one molecule of each of the species $Z_1, \ldots, Z_k$. The solution obtained is $P - \{\!|\, X, Y \,|\!\} \cup \{\!|\, Z_1, \ldots, Z_k \,|\!\}$. According to the *Chemical Law of Mass Action*, the speed of a chemical reaction in a solution is proportional to the number of possible interactions of its reactants in the solution. It is distributed exponentially, and defines a CTMC with chemical solutions as states and the following transitions:

$$P \xrightarrow{n \cdot r} \begin{cases} P - \{\!|\, X, Y \,|\!\} \\ \cup \{\!|\, Z_1, \ldots, Z_k \,|\!\} \end{cases} \quad \text{where} \quad n = \begin{cases} \sharp(X \in P) \times \sharp(Y \in P) & \text{if } X \neq Y \\ \binom{\sharp(X \in P)}{2} & \text{else} \end{cases}$$

$\binom{m}{2} = \frac{1}{2}\, m\, (m-1)$ counts the number of two-element subsets in sets of cardinality m.

6 Stochastic Semantics

We define the stochastic semantics of Core SPICO by associating a π-calculus process with a CTMC. The states of this Markov chain are the (countably infinite) set of congruence classes of π-calculus processes with respect to structural congruence. This differs from [Pri95] where two congruent processes are associated with two different states. Since congruent processes are behaviorally equivalent we believe that their associated stochastic states should not be distinguished neither. Moreover, in [Pri95], the author proposes a *labelled semantics* where labels are so-called *proof terms*, i.e. (possibly long) strings used to localize interacting sub-

terms. Those labels are necessary to properly calculate interaction rates. We instead propose a *reduction semantics*, a kind of semantics known to be more intuitive and elegant. Still, we temporarily use labels but in a much simpler form: a label is an integer or a tuple of four integers. Finally, and contrary to [Pri95], our semantics takes into account immediate transitions which we emphasized the importance in the biological example in section 3. Such transitions require specific consideration: we show how they can be removed in order to obtain an equivalent Markovian transition system. The theorem 1 states the correctness of the transformation.

6.1 Transition relations

We first consider the *fragment* of the π-calculus without proper summation, parametric processes, infinite rates, and **new**-binders. The remaining processes are parallel compositions $C_1 \mid \ldots \mid C_n$. The structural congruence turns them into multisets of guarded processes, i.e. into chemical solutions whose species are guarded processes.

Suppose we know the rate functions $\varrho(x)$ for all channels x. The π-calculus with input patterns then defines the following chemical reaction rule:

$$x?f(\tilde{z}).Q_1 \mid x!f(\tilde{y}).Q_2 \xrightarrow{\varrho(x)(f)} Q_1[\tilde{z} \mapsto \tilde{y}] \mid Q_2$$

This defines a CTMC. For example, assume n molecules of a first species $x!f().P_1$ and m of another different one $x!f().P_2$, which all want to react with a single molecule of a third kind $x?f().P$. The Markovian transitions are:

$$\prod_{i=1}^{n} x!f().P_1 \mid \prod_{i=1}^{m} x!f().P_2 \mid x?f().P \begin{cases} \xrightarrow{n\times\varrho(x)(f)} & \prod_{i=1}^{n-1} x!f().P_1 \mid \prod_{i=1}^{m} x!f().P_2 \mid P \\ \xrightarrow{m\times\varrho(x)(f)} & \prod_{i=1}^{n} x!f().P_1 \mid \prod_{i=1}^{m-1} x!f().P_2 \mid P \end{cases}$$

We first discuss time consuming transitions $P \xrightarrow{r} P'$ where $r \in \mathbb{R}^+$. These capture everything, except parametric process unfolding and invocation of functions of rate ∞.

We first define labeled reduction steps $P \xrightarrow[w]{s} Q$ where P and Q are in prenex normal form, that is a parallel composition of sums where restrictions have been pushed ahead and in which bound variables are renamed apart. The rate function $\varrho(x)$ is then read off from the quantifier prefix in rule (NEW).

Definition 1 *P is in prenex normal form (pnf for short) iff $P =$ **new** $\widetilde{x{:}\rho}.\ (P_1 \mid \ldots \mid P_m)$ where each P_i either is an application $A(\tilde{y})$, or a sum $C_1 + \ldots + C_n$ where each C_j is in pnf, or a guarded process $x?f(\tilde{y}).Q$ or $x!f(\tilde{y}).Q$ where Q is in pnf. Moreover, a definition $A(\tilde{y}) \triangleq P$ is in pnf iff P is in pnf.*

What remains from pnf after removing top-level **new**-binders are multisets of sums and applications. All applications must have been reduced before time consuming transitions apply, so we have a multiset of sums. Each sum is like a molecule, except that each of its choices offers its own interactions.

Labeled reduction steps

$$\text{(COM)}\quad \frac{C_{i_1}^{j_1} = x?f(\tilde{z}).\mathbf{new}\ \widetilde{x_1{:}\rho_1}.\ Q_1 \qquad C_{i_2}^{j_2} = x!f(\tilde{y}).\mathbf{new}\ \widetilde{x_2{:}\rho_2}.\ Q_2}{\Pi_{i=1}^{n}\sum_{j=1}^{m_i} C_i^j \xrightarrow[i_1,j_1,i_2,j_2]{\varrho(x)(f)} \begin{cases} \mathbf{new}\ \widetilde{x_1{:}\rho_1}.\ \mathbf{new}\ \widetilde{x_2{:}\rho_2}. \\ (Q_1[\tilde{z}\mapsto\tilde{y}] \mid Q_2 \mid \Pi_{i=1,i\neq i_1,i_2}^{n}\sum_{j=1}^{m_i} C_i^j) \end{cases}}$$

where Q_1, Q_2 have no top-level **new**-binders and
$1 \le i_1 \neq i_2 \le n,\ 1 \le j_1 \le m_{i_1},\ 1 \le j_2 \le m_{i_2}$

$$\text{(APP)}\quad \frac{P_{i_1} = A(\tilde{y}) \qquad A(\tilde{x}) \triangleq \mathbf{new}\ \widetilde{z{:}\rho}.\ Q \text{ in } \Delta}{\Pi_{i=1}^{n} P_i \xrightarrow[i_1]{\infty} \mathbf{new}\ \widetilde{z{:}\rho}.\ (Q[\tilde{x}\mapsto\tilde{y}] \mid \Pi_{i=1,i\neq i_1}^{n} P_i)}$$

where Q has no top-level **new**-binders and $1 \le i_1 \le n$

$$\text{(NEW)}\quad \frac{P \xrightarrow[w]{s} Q \qquad \varrho(x) = \rho}{\mathbf{new}\ x{:}\rho.\ P \xrightarrow[w]{s} \mathbf{new}\ x{:}\rho.\ Q} \quad \text{where } s \in \mathbb{R}^+ \cup \{\infty\},\ w \in \mathbb{N} \cup \mathbb{N}^4$$

Time consuming transitions $(r, r' \in \mathbb{R}^+,\ w \in \mathbb{N}^4)$

$$\text{(SUM)}\quad \frac{P \equiv P' \qquad r = \sum_{P' \xrightarrow[w]{r'} Q' \equiv Q} r' \neq 0 \qquad \neg\exists R \exists w' \in \mathbb{N} \cup \mathbb{N}^4.\ P' \xrightarrow[w']{\infty} R}{P \xrightarrow{r} Q}$$

Immediate transitions

$$\text{(COUNT)}\quad \frac{P \equiv P' \qquad \begin{array}{l} n = \sharp\{w \in \mathbb{N} \cup \mathbb{N}^4 \mid P' \xrightarrow[w]{\infty} Q' \equiv Q\} \neq 0 \\ m = \sharp\{w \in \mathbb{N} \cup \mathbb{N}^4 \mid P' \xrightarrow[w]{\infty} Q''\} \end{array}}{P \xrightarrow{\infty(n/m)} Q}$$

Table 5: Timed transitions of Core SPICO with respect to a set Δ of definitions in prenex normal form, and a global assignment ϱ of channels to rate functions.

In $x?f().\mathbf{0} + x?f().\mathbf{0} \mid x!f().\mathbf{0}$ there are two possible interactions with rate $r = \varrho(x)(f)$ leading to the same state. We can think of $x?f().\mathbf{0} + x?f().\mathbf{0}$ as a protein with two identical domains, complementary to one domain of some other protein represented by $x!f().\mathbf{0}$. The overall rate of the interaction thus doubles:

$$\begin{array}{rrcl} & \text{x?f()}\ .\mathbf{0} + x?f().\mathbf{0} \mid \text{x!f()}\ .\mathbf{0} & \xrightarrow[1.1.2.1]{r} & \mathbf{0} \\ \text{and} & x?f().\mathbf{0} + \text{x?f()}\ .\mathbf{0} \mid \text{x!f()}\ .\mathbf{0} & \xrightarrow[1.2.2.1]{r} & \mathbf{0} \\ \text{sums up to} & x?f().\mathbf{0} + x?f().\mathbf{0} \mid x!f().\mathbf{0} & \xrightarrow{2r} & \mathbf{0} \end{array}$$

Rule (COM) defines labeled reductions $P \xrightarrow[i_1,j_1,i_2,j_2]{r} Q$ that distinguish communication actions with identical reactants and results, while using different occurrences of choice alternatives in sums. Those occurrences are identified by *labels* in $\mathbb{N}^4$ that specify the numbers of the reacting sums (i_1, i_2) and the reacting choices (j_1, j_2). Rule (SUM) defines transitions $P \xrightarrow{r} Q$ by summing up all rates of all different interactions leading from P to Q. These reduction rules are defined with care, so that corresponding interactions in structurally congruent processes are not counted twice.

We next turn to *immediate transitions* $P \xrightarrow{\infty(p)} Q$, where $p \in [0, 1]$ is a probability. Rule (SUM) ensures that time consuming transitions apply only after all immediate have been reduced. In this case, all calls $A(\tilde{y})$ on top level must have been reduced before. Note that this order is important for a proper count of the possible interactions. Indeed, if an application hides an interaction on some pattern, the application unfolding changes the rate of the action involving this pattern. Immediate transitions can be licensed by communication (COM), or by applications of parametric process definitions (APP). Their labels are in $\mathbb{N} \cup \mathbb{N}^4$. Note that the labeled reduction is independent of the choice of the prenex normal form as stated by the following lemma.

Lemma 1 *Let P and P' be in prenex normal form and $s \in \mathbb{R}^+ \cup \{\infty\}$. If $P \equiv P'$ and $P \xrightarrow[w]{s} Q$, then there exist $Q' \equiv Q$ and w' such that $P' \xrightarrow[w']{s} Q'$.*

Proof. Straightforward.

We merge labeled immediate transitions with rule (COUNT). Although being immediate we want to associate probabilities, which characterize the number of immediate interactions leading to a common state with respect to the total number of enabled immediate interactions. For instance, let $\varrho(x)(f) = \infty$, in $x?f().P + x?f().P \mid x?f().Q \mid x!f().\mathbf{0}$, for some $P \not\equiv Q$, the associated probabilities reflect that 2 out of 3 interactions lead to P, and 1 out of 3 to Q:

$$x?f().P + x?f().P \mid x?f().Q \mid x!f().\mathbf{0} \xrightarrow{\infty(2/3)} P$$
$$x?f().P + x?f().P \mid x?f().Q \mid x!f().\mathbf{0} \xrightarrow{\infty(1/3)} Q$$

Lemma 2 *The following properties hold:*

1. $\equiv \circ \xrightarrow{r} \circ \equiv \subseteq \xrightarrow{r}$,
2. $\equiv \circ \xrightarrow{\infty(p)} \circ \equiv \subseteq \xrightarrow{\infty(p)}$,
3. *if $P \xrightarrow{r} Q$ and $P \xrightarrow{r'} Q$ then $r = r'$,*
4. *if $P \xrightarrow{\infty(p_1)} Q$ and $P \xrightarrow{\infty(p_2)} Q$ then $p_1 = p_2$, and*
5. *for all P: $\sum_{P \xrightarrow{\infty(p)} Q} p = 1$.*

Proof.

1. Let $P \equiv \circ \xrightarrow{r} \circ \equiv Q$, then there are P' and Q' such that $P \equiv P'$, $P' \xrightarrow{r} Q'$ and $Q' \equiv Q$. Since $P' \xrightarrow{r} Q'$ is necessarily inferred by (SUM), there is $P'' \equiv P'$ (1) such that $r = \sum_{P'' \xrightarrow[w]{r'} Q'' \equiv Q'} r' \neq 0$ (2), and $\neg \exists R \exists w' \in \mathbb{N} \cup \mathbb{N}^4 . P'' \xrightarrow[w']{\infty} R$ (3). From $P' \equiv P$ and (1), we have $P \equiv P''$ (1'), and by $Q' \equiv Q$ and (2) we have $r = \sum_{P'' \xrightarrow[w]{r'} Q'' \equiv Q} r' \neq 0$ (2'). Then, by (1'), (2'), (3) and by (SUM), we conclude $P \xrightarrow{r} Q$.

2. Similar to the previous point, by rule (COUNT) and Lemma 1.

3. Suppose that $P \xrightarrow{r} Q$ and $P \xrightarrow{r'} Q$, then by rule (SUM), there are P_1 and P_2 such that $P_1 \equiv P \equiv P_2$ and $r = \sum_{P_1 \xrightarrow[w_1]{r_1} Q_1 \equiv Q} r_1$ and $r' = \sum_{P_2 \xrightarrow[w_2]{r_2} Q_2 \equiv Q} r_2$. However, by Lemma 1, there is $Q_1 \equiv Q$ such that $P_1 \xrightarrow[w_1]{r_1} Q_1$ iff there is $Q_2 \equiv Q$ such that $P_2 \xrightarrow[w_2]{r_2} Q_2$ and $r_1 = r_2$. Therefore, $r = r'$.

4. Similar to the previous point, using Lemma 1.

5. Assume, without loss of generality, that P is in prenex normal form. Let $\{Q_1, \ldots, Q_k\}$ be the set (up-to $\equiv$) of all the possible immediate and distinct derivatives of P, that is satisfying $\forall i \in \{1, \ldots, k\}$ $P \xrightarrow{\infty(p_i)} Q_i$ and $i \neq j \Rightarrow Q_i \not\equiv Q_j$. Let $X = \{w \in \mathbb{N} \cup \mathbb{N}^4 \mid P \xrightarrow[w]{\infty} Q''\}$ and $Y_i = \{w \in \mathbb{N} \cup \mathbb{N}^4 \mid P \xrightarrow[w]{\infty} Q' \equiv Q_i\}$. It is clear that, $i \neq j \Rightarrow Y_i \cap Y_j = \emptyset$ (since $Q_i \not\equiv Q_j$) and $\bigcup_{1 \leq i \leq k} Y_i = X$. Therefore $\sum_{1 \leq i \leq k} \sharp Y_i = \sharp X$, and, since $p_i = \sharp Y_i / \sharp X$, $\sum_{1 \leq i \leq k} p_i = 1$. □

6.2 CTMCs with immediate reactions

In the presence of immediate transitions, the reduction relation $\xrightarrow{r}$ does not define a Markovian transition system (in which all rates are finite). To capture the stochastic dynamics of processes, we instead define the *sojourn time parameters* (i.e. the parameter of an exponentially distributed probability which determine the sojourn time in a given state) and the *probabilities of state changes* for all P, Q as follows[1]:

$$\downarrow P \;=\; \begin{cases} \infty & \text{if } P \xrightarrow{\infty(p)} Q, \\ \sum_{P \xrightarrow{r} Q} r & \text{otherwise.} \end{cases} \qquad Pr(P \to Q) \;=\; \begin{cases} r / \sum_{P \xrightarrow{r'} Q'} r' & \text{if } P \xrightarrow{r} Q \\ p & \text{if } P \xrightarrow{\infty(p)} Q \\ 0 & \text{otherwise} \end{cases}$$

[1] we assume if X is exponentially distributed with parameter ∞ then $Pr(X = 0) = 1$.

$$(\text{ELIM}_1)\quad \frac{P \xrightarrow[w]{\infty} Q \qquad n = \sharp\{w' \in \mathbb{N} \cup \mathbb{N}^4 \mid P \xrightarrow[w']{\infty} Q'\}}{P \xrightarrow[w]{\infty(1/n)} Q} \qquad w \in \mathbb{N} \cup \mathbb{N}^4$$

$$(\text{ELIM}_2)\quad \frac{P \xrightarrow[w]{r} Q \qquad Q \xrightarrow[w_1]{\infty(p_1)} \cdots \xrightarrow[w_n]{\infty(p_n)} Q_n \not\xrightarrow{\infty}}{P \underset{ww_1 \ldots w_n}{\overset{rp_1 \ldots p_n}{\Longrightarrow}} Q_n} \qquad r \in \mathbb{R}^+$$

$$(\text{ELIM}^{\text{sum}})\quad \frac{P \equiv P' \qquad r = \sum_{P' \underset{w_1 \ldots w_n}{\overset{r'}{\Longrightarrow}} Q' \equiv Q} r'}{P \overset{r}{\Longrightarrow} Q}$$

Table 6: Elimination of immediate transitions and merging timed transitions

We are now giving an interpretation of the reduction semantics with immediate transitions in terms of CTMCs for processes that can not exhibit infinite sequences of immediate transitions. The Markovian transition system deriving statements $P \overset{r}{\Longrightarrow} Q$ is defined in Table 6. The idea is quite similar to that of [BDG94]: the transitions are obtained by integrating immediate transitions into time consuming transitions. An example for this transformation is as follows:

$$P \begin{cases} \xrightarrow{r_1} & Q_1 \not\xrightarrow{\infty} \\ \xrightarrow{r_2} & Q_2 \begin{cases} \xrightarrow{\infty(p)} & Q_{21} \not\xrightarrow{\infty} \\ \xrightarrow{\infty(1-p)} & Q_{22} \not\xrightarrow{\infty} \end{cases} \end{cases} \quad \text{becomes} \quad P \begin{cases} \overset{r_1}{\Longrightarrow} & Q_1 \\ \overset{r_2 p}{\Longrightarrow} & Q_{21} \\ \overset{r_2(1-p)}{\Longrightarrow} & Q_{22} \end{cases}$$

In general, a sequence of reductions $P \xrightarrow{r} P_1 \xrightarrow{\infty(p_1)} \ldots P_n \xrightarrow{\infty(p_n)} Q \not\xrightarrow{\infty}$ reduces to $P \overset{rp_1 \ldots p_n}{\Longrightarrow} Q$. However, we must beware of merging initially distinct states. Indeed, in the previous example, if $Q_{22} \equiv Q_1$ then the CTMC should have transitions $P \overset{r_1 + r_2(1-p)}{\Longrightarrow} Q_1$ and $P \overset{r_2 p}{\Longrightarrow} Q_{21}$. In order to infer these transitions correctly, the elimination procedure defines labeled transitions $\underset{w}{\overset{r}{\Longrightarrow}}$ with *labels* $w \in (\mathbb{N} \cup \mathbb{N}^4)^\star$ representing *paths* in the labeled derivation trees of $\xrightarrow{r}$.

Lemma 3 *Let P and P' be in prenex normal form. If $P \equiv P'$ and $P \underset{w}{\overset{r}{\Longrightarrow}} Q$ (resp. $P \underset{w}{\overset{\infty(p)}{\Longrightarrow}} Q$), then there exists $Q' \equiv Q$ and w' such that $P' \underset{w'}{\overset{r}{\Longrightarrow}} Q'$ (resp. $P' \underset{w'}{\overset{\infty(p)}{\Longrightarrow}} Q'$).*

Proof.

1. Suppose that $P \underset{w}{\overset{r}{\Longrightarrow}} Q$, by rule ($\text{ELIM}_2$), we have $P \underset{w_0}{\overset{r_0}{\longrightarrow}} Q_0$ and $Q_0 \underset{w_1}{\overset{\infty(p_1)}{\longrightarrow}} \cdots \underset{w_n}{\overset{\infty(p_n)}{\longrightarrow}} Q_n = Q \not\xrightarrow{\infty}$ where $r = r_0 p_1 \ldots p_n$ and $w = w_0 w_1 \ldots w_n$. By Lemma 1, there exists $Q_0' \equiv Q_0$ such that $P' \underset{w_0'}{\overset{r_0}{\longrightarrow}} Q_0'$. Q_0' is necessarily in prenex normal form. Thus by

Lemma 1, there are $Q'_1 \equiv Q_1, \ldots Q'_n = Q' \equiv Q$ all in prenex normal form such that $Q'_0 \xrightarrow[w'_1]{\infty(p_1)} \cdots \xrightarrow[w'_n]{\infty(p_n)} Q'_n \not\xrightarrow{\infty}$. Therefore, by rule (ELIM$_2$), $P' \overset{r}{\underset{w'}{\Longrightarrow}} Q' \equiv Q$ where $w' = w'_0 w'_1 \ldots w'_n$.

2. Suppose that $P \xrightarrow[w]{\infty(p)} Q$, by rule (ELIM$_1$), we have $P \xrightarrow[w]{\infty} Q$ and $p = 1/\sharp\{w_0 \in \mathbb{N} \cup \mathbb{N}^4 \mid P \xrightarrow[w_0]{\infty} R\}$. By Lemma 1, there exist w' and $Q' \equiv Q$ such that $P' \xrightarrow[w']{\infty} Q'$ and $\sharp\{w_0 \in \mathbb{N} \cup \mathbb{N}^4 \mid P \xrightarrow[w_0]{\infty} R\} = \sharp\{w'_0 \in \mathbb{N} \cup \mathbb{N}^4 \mid P' \xrightarrow[w'_0]{\infty} R'\}$. Therefore, by (ELIM$_1$), we have $P' \xrightarrow[w']{\infty(p)} Q'$. □

Lemma 4 *$\equiv \circ \overset{r}{\Longrightarrow} \circ \equiv \subseteq \overset{r}{\Longrightarrow}$ and if $P \overset{r}{\Longrightarrow} Q$ and $P \overset{r'}{\Longrightarrow} Q$ then $r = r'$.*

Proof. Similar to the proof of Lemma 2 using Lemma 3.

For any P such that $P \not\xrightarrow{\infty}$, $(\mathscr{P}_{/\equiv}, (\overset{r}{\Longrightarrow})_{r \in \mathbb{R}^+}, P_{/\equiv})$ is a Markovian transition system[2] with sojourn time parameters and transition probabilities:

$$\Downarrow P = \sum_{P \overset{r}{\Longrightarrow} Q} r \qquad \text{and} \qquad Pr(P \Rightarrow Q) = \begin{cases} r / \sum_{P \overset{r'}{\Longrightarrow} Q'} r' & \text{if } P \overset{r}{\Longrightarrow} Q \\ 0 & \text{otherwise} \end{cases}$$

In order to show that this defines a Markovian model for the reduction semantics with immediate transitions, we show that their dynamics coincide, that is: the sojourn time parameters and the transition probabilities with respect to $\overset{r}{\rightarrow}$ are identical to those of $\overset{r}{\Longrightarrow}$. However, transition probabilities can be compared only for processes performing timed transitions (that is processes P such that $P \not\xrightarrow{\infty}$). We thus define a suitable transition probability $Pr(P \twoheadrightarrow Q)$ for $P \not\xrightarrow{\infty}$ and $Q \not\xrightarrow{\infty}$, that is the probability to reach Q from P by a sequence of transitions made of one timed transition and possibly several intermediate immediate transitions. Formally, $Pr(P \twoheadrightarrow Q)$ is the sum of the probabilities of all such sequences:

$$Pr(P \twoheadrightarrow Q) = \sum_{P \overset{r}{\rightarrow} Q_1 \xrightarrow{\infty(p_1)} \ldots Q_n \xrightarrow{\infty(p_n)} Q \not\xrightarrow{\infty}} \left(Pr(P \rightarrow Q_1) \times \prod_{i=1}^{n} p_i \right)$$

Theorem 1 *If $P \not\xrightarrow{\infty}$ and if no infinite sequence of immediate transitions is reachable from P, then*

- *(Timed correctness) $\downarrow P = \Downarrow P$,*

[2] For $P \xrightarrow{\infty}$ it suffices to start with a process $Q = x!f().\mathbf{0} \mid x?f().P$ such that $\varrho(x)(f) = 1$ in order to obtain a set of initial processes together with an initial probability distribution of those processes rather than a single initial process.

- *(Probabilistic correctness)* $Pr(P \twoheadrightarrow Q) = Pr(P \Rightarrow Q)$.

Proof.

- *Timed correctness.* First, note that for any P we have $\sum_{P \xrightarrow[w]{\infty(p)} Q} p = 1$. In order to show that $\downarrow P = \Downarrow P$, it is sufficient, by rules (SUM) and (ELIM^{sum}) to show that

$$\sum_{P \xrightarrow[w]{r} Q} r \quad = \quad \sum_{P \xRightarrow[w]{r} Q} r \tag{1}$$

For any $n \geq 1$, we define the relation $P\big(\xrightarrow[w]{\infty(p)}\big)^n Q$ meaning the longest derivation of immediate actions leading from P to Q and which length is smaller or equal to n. More formally, $P\big(\xrightarrow[w]{\infty(p)}\big)^n Q$ iff

- either if there exists a (unique) sequence $P \xrightarrow[w_1]{\infty(p_1)} Q_1 \ldots \xrightarrow[w_n]{\infty(p_n)} Q_n = Q$ and such that $p = p_i \times \ldots \times p_n$ and $w = w_1 \ldots w_n$, or
- or if $n > 1$ and $P\big(\xrightarrow[w]{\infty(p)}\big)^{(n-1)} Q$.

Given P, let us define n_P as the length of the longest derivation of immediate actions leading from P to some process Q. By the hypothesis of none infinite sequences of immediate actions, such an integer always exists and if $P\big(\xrightarrow[w]{\infty(p)}\big)^{n_P} Q$ then $Q \not\xrightarrow{\infty}$. It is clear that $P \xRightarrow[w]{r} Q$ iff either $P \xrightarrow[w]{r} Q \not\xrightarrow{\infty}$ or, $P \xrightarrow[w_0]{r'} R\big(\xrightarrow[w']{\infty(p)}\big)^{n_R} Q$ for some R and where $r = pr'$ and $w = w_0 w'$. Moreover, one can easily show (by induction on n) that for any P and n, $\sum_{P\left(\xrightarrow[w]{\infty(p)}\right)^n Q} p = 1$. Therefore, given P, we have

$$\begin{aligned}
\sum_{P \xRightarrow[w]{r} Q} r \quad &= \quad \sum_{P \xrightarrow[w_0]{r'} R} \Big(r \times \big(\sum_{R\left(\xrightarrow[w']{\infty(p)}\right)^{n_R} Q} p \big) \Big) \\
&= \quad \sum_{P \xrightarrow[w_0]{r'} R} \big(r \times 1 \big) \\
&= \quad \sum_{P \xrightarrow[w]{r} Q} r
\end{aligned}$$

which proves (1).

- *Probabilistic correctness.* By timed correctness, we have $\sum_{P \stackrel{r'}{\Longrightarrow} Q} r' = \sum_{P \stackrel{r'}{\longrightarrow} Q} r'$ (1).

  Moreover, one can easily show that $P \xrightarrow{\infty(p)} Q$, iff $\sum_{P \xrightarrow[w]{\infty(p')} R \equiv Q} p' = p$ (2). Given P, Q and length derivation n, we have:

$$
\begin{aligned}
& Pr(P \xrightarrow{r} Q_1 \xrightarrow{\infty(p_1)} \ldots Q_n \xrightarrow{\infty(p_n)} Q_{n+1} \equiv Q \not\xrightarrow{\infty}) \\
&= (\textstyle\sum_{P \stackrel{r'}{\longrightarrow} Q'} r')^{-1} (r \times \prod_{1 \le i \le n} p_i) \\
&= (\textstyle\sum_{P \stackrel{r'}{\Longrightarrow} Q'} r')^{-1} (r \times \prod_{1 \le i \le n} p_i) && \text{by (1)} \\
&= (\textstyle\sum_{P \stackrel{r'}{\Longrightarrow} Q'} r')^{-1} \Big((\sum_{P \xrightarrow[w]{r'} Q' \equiv Q_1} r') \times \prod_{1 \le i \le n} \big(\sum_{Q_i \xrightarrow[w]{\infty(p'_i)} Q' \equiv Q_{i+1}} p'_i \big) \Big) && \text{by (2)} \\
&= (\textstyle\sum_{P \stackrel{r'}{\Longrightarrow} Q'} r')^{-1} \big(\sum_{P \xrightarrow[w_0]{r_0} Q'_1 \xrightarrow[w_1]{\infty(p'_1)} \ldots \xrightarrow[w_n]{\infty(p'_n)} Q'_{n+1} \equiv Q} (r_0 \times \textstyle\prod_{1 \le i \le n} p'_i) \big)
\end{aligned}
$$

  Therefore, summing over all length derivations n, we conclude that $Pr(P \twoheadrightarrow Q) = Pr(P \Rightarrow Q)$. □

7 Encoding Input Patterns

We now encode our π-calculus with input patterns back into the stochastic π-calculus. The latter can be identified as the special case with a unique function name per arity (we assume arities bounded by some max): $\mathscr{F}' = \{\text{UNIT}_i \mid 0 \le i \le max\}$. In what follows, we write UNIT instead of UNIT_i.

7.1 Translation

We assume a total ordering $<$ on a finite set of function names $\mathscr{F}$. This means that $\mathscr{F}$ has a unique representation $\mathscr{F} = \{f_1, \ldots, f_n\}$ with $f_1 < \ldots < f_n$. Our encoding uses channel names from the set $\mathscr{N} \times \mathscr{F}$. We denote elements (x, f) of this set by x_f. For each channel x we define a sequence of n channels $x_{\mathscr{F}}$ as follows: $x_{\mathscr{F}} =_{\text{def}} x_{f_1}, \ldots, x_{f_n}$. Channels in the target language are associated a rate (that may be infinite) by means of the encoding of ϱ defined as $[\![\varrho]\!](x_f) = \varrho(x)(f)$. We write $\tilde{x}, \tilde{y}$ for the concatenation of two sequences $\tilde{x}$ and $\tilde{y}$. If $\tilde{x} = x_1, \ldots, x_n$ then we let $\tilde{x}_{\mathscr{F}} =_{\text{def}} x_{1\mathscr{F}}, \ldots, x_{n\mathscr{F}}$. The encoding is given in Table 7.

Lemma 5 *For all processes P, P' with functions in $\mathscr{F}$ and variable sequences $\tilde{y}, \tilde{z}$ of the same length:*

1. *$\tilde{y}$ is free for $\tilde{z}$ in P if and only if $\tilde{y}_{\mathscr{F}}$ is free for $\tilde{z}_{\mathscr{F}}$ in $[\![P]\!]$.*
2. *$[\![P[\tilde{y} \mapsto \tilde{z}]]\!] = [\![P]\!]\ [\tilde{y}_{\mathscr{F}} \mapsto \tilde{z}_{\mathscr{F}}]$*

$[\![\mathbf{new}\ x{:}\rho.\ P]\!] =_{\text{def}} \mathbf{new}\ x_{f_1}{:}\rho(f_1).\ \cdots\ .\mathbf{new}\ x_{f_n}{:}\rho(f_n).\ [\![P]\!]$	
$[\![P_1 \mid P_2]\!] =_{\text{def}} [\![P_1]\!] \mid [\![P_2]\!]$	$[\![A(\tilde{y})]\!] =_{\text{def}} A(\tilde{y}_{\mathscr{F}})$
$[\![C_1 + \cdots + C_n]\!] =_{\text{def}} [\![C_1]\!] + \cdots + [\![C_n]\!]$	$[\![A(\tilde{x}) \triangleq P]\!] =_{\text{def}} A(\tilde{x}_{\mathscr{F}}) \triangleq [\![P]\!]$
$[\![x?f(\tilde{y}).P]\!] =_{\text{def}} x_f?(\tilde{y}_{\mathscr{F}}).[\![P]\!]$	$[\![x!f(\tilde{y}).P]\!] =_{\text{def}} x_f!(\tilde{y}_{\mathscr{F}}).[\![P]\!]$

Table 7: Encoding of input patterns

3. *If* $[\![P]\!] \equiv Q$ *then there exists* $P' \in [\![Q]\!]^{-1}$ *such that* $P \equiv P'$.
4. $P \equiv P'$ *if and only if* $[\![P]\!] \equiv [\![P']\!]$.
5. P *is in prenex normal form iff* $[\![P]\!]$ *is.*

Proof.

1. By induction on the definition of freeness conditions.
2. By induction on the structure of P. Note that the lemma may fail for processes P with function symbols outside $\mathscr{F}$. As a counter example let $\mathscr{F} = \emptyset$ and consider $x?f().\mathbf{0}[x \mapsto y]$.
3. By induction on the structure of P.
4. By induction on derivations of $P \equiv P'$ resp. $[\![P]\!] \equiv [\![P']\!]$.
5. By induction on the structure of P.

The following theorem shows the correctness of our encoding. It allows us to run simulations of models expressed in our π-calculus with objects, via an implementation of the original stochastic π-calculus, as implemented in the SPiM system [PC06].

Theorem 2 *The encoding defines a stochastic bisimulation: for all processes* P, Q *and finite sets of definitions* Δ*, and all rates* $s \in \mathbb{R}^+ \cup \{\infty(p) \mid p \in]0,1]\}$ *it holds that* $P \xrightarrow{s} Q$ *relative to* Δ *if and only if* $[\![P]\!] \xrightarrow{s} [\![Q]\!]$ *relative to* $[\![\Delta]\!]$.

The statement $P \xrightarrow{s} Q$ relative to Δ means that there exists some function $\varrho : \mathscr{N} \to \mathscr{F} \to (\mathbb{R}^+ \cup \{\infty\})$ such that $P \xrightarrow{s} Q$ relative to Δ and ϱ. The values $\varrho(x)$ will be the rate ρ assigned to x in the declaration $\mathbf{new}\ x{:}\rho$. It holds for all ρ and x that $\varrho(x) = \rho$ iff $[\![\varrho]\!](x_f) = \rho(f)$ for all $f \in \mathscr{F}$.

The statement $[\![P]\!] \xrightarrow{s} [\![Q]\!]$ relative to $[\![\Delta]\!]$ means that there exists some function $\varrho' : \{x_f \mid f \in \mathscr{F}, x \in \mathscr{N}\} \to (\mathbb{R}^+ \cup \{\infty\})$ such that $[\![P]\!] \xrightarrow{s} [\![Q]\!]$ relative to $[\![\Delta]\!]$ and ϱ'. The situation differs in that there exists only a single function UNIT for all arities. We are a little sloppy in identifying a constant function with its constant value, i.e. $\varrho'(x_f) = \varrho'(x_f)(\text{UNIT})$.

7.2 Correctness proof

We prove a slightly stronger proposition than Theorem 2. We define a translation of functions $\varrho : \mathscr{N} \to \mathscr{F} \to (\mathbb{R}^+ \cup \{\infty\})$ to functions $[\![\varrho]\!] : \{x_f \mid f \in \mathscr{F}, x \in \mathscr{N}\}$ such that for all $x \in \mathscr{N}$ and $f \in \mathscr{F}$:

$$[\![\varrho]\!](x_f) =_{\text{def}} \varrho(x)(f)$$

The translation is onto, i.e for all $\varrho' : \{x_f \mid f \in \mathscr{F}, x \in \mathscr{N}\} \to (\mathbb{R}^+ \cup \{\infty\})$ there exists some $\varrho : \mathscr{N} \to \mathscr{F} \to (\mathbb{R}^+ \cup \{\infty\})$ such that $\varrho' = [\![\varrho]\!]$. Hence, the theorem follows from the following proposition:

Proposition 1 *$P \xrightarrow{s} Q$ with respect to Δ and ϱ iff $[\![P]\!] \xrightarrow{s} [\![Q]\!]$ with respect to $[\![\Delta]\!]$ and $[\![\varrho]\!]$.*

We need some auxiliary lemmas, before we can show the proposition for infinite rates in Lemma 9 and for finite rates in Lemma 10. All these Lemmas hold for all processes P, Q, Q' and definitions Δ with functions in $\mathscr{F}$, rates $s \in \mathbb{R}^+ \cup \{\infty\}$ or $r \in \mathbb{R}^+$, functions ϱ, labels $w \in \mathbb{N} \cup \mathbb{N}^4$, and probabilities $p \in]0, 1]$. For convenience, we define

$$[\![\textbf{new } y_1{:}\rho_1.\ \cdots\ .\ \textbf{new } y_n{:}\rho_n]\!] =_{\text{def}} [\![\textbf{new } y_1{:}\rho_1]\!].\ \cdots\ .\ [\![\textbf{new } y_n{:}\rho_n]\!]$$
$$[\![\textbf{new } y{:}\rho]\!] =_{\text{def}} \textbf{new } y_{f_1}{:}\rho(f_1).\ \cdots\ .\ \textbf{new } y_{f_n}{:}\rho(f_n)$$

Lemma 6 *If $P \xrightarrow[w]{s} Q$ with respect to Δ and ϱ, then $[\![P]\!] \xrightarrow[w]{s} [\![Q]\!]$ with respect to $[\![\Delta]\!]$ and $[\![\varrho]\!]$.*

Proof. By rule induction. We need to consider the three rules (NEW), (APP), and (COM) by which to infer $P \xrightarrow[w]{s} Q$.

- Rule (NEW). Suppose that $\textbf{new } x{:}\rho.\ P \xrightarrow[w]{s} \textbf{new } x{:}\rho.\ Q$ is inferred as follows:

$$\frac{P \xrightarrow[w]{s} Q \qquad \varrho(x) = \rho}{\textbf{new } x{:}\rho.\ P \xrightarrow[w]{s} \textbf{new } x{:}\rho.\ Q}$$

  The induction hypothesis yields that $[\![P]\!] \xrightarrow[w]{s} [\![Q]\!]$. As argued above, $\varrho(x) = \rho$ is equivalent to that $[\![\varrho]\!](x_f) = \rho(f)$ for all $f \in \mathscr{F}$. We can thus infer $[\![\textbf{new } x{:}\rho.\ P]\!] \xrightarrow[w]{r} [\![\textbf{new } x{:}\rho.\ Q]\!]$ by applying the (NEW) rule in an iterative manner:

$$\frac{[\![P]\!] \xrightarrow[w]{s} [\![Q]\!] \qquad \varrho(x_{f_1}) = [\![\rho]\!](f_1) \qquad \cdots \qquad [\![\varrho]\!](x_{f_n}) = \rho(f_n)}{\begin{array}{l}\textbf{new } x_{f_1}{:}\rho(f_1).\ \cdots\ .\ \textbf{new } x_{f_n}{:}\rho(f_n).\ [\![P]\!] \xrightarrow[w]{s} \\ \textbf{new } x_{f_1}{:}\rho(f_1).\ \cdots\ .\ \textbf{new } x_{f_n}{:}\rho(f_n).\ [\![Q]\!]\end{array}}$$

- Rule (APP). In this case, the judgment has been derived as follows:

$$\frac{P_{i_1} = A(\tilde{y}) \qquad A(\tilde{x}) \triangleq \mathbf{new}\ \widetilde{z{:}\rho}.\ Q \text{ in } \Delta}{\Pi_{i=1}^n P_i \xrightarrow[i_1]{\infty} \mathbf{new}\ \widetilde{z{:}\rho}.\ (Q[\tilde{x} \mapsto \tilde{y}] \mid \Pi_{i=1, i\neq i_1}^n P_i)}$$

  By translation the following rule instance applies too:

$$\frac{[\![P_{i_1}]\!] = A(\tilde{y}_{\mathscr{F}}) \qquad A(\tilde{x}_{\mathscr{F}}) \triangleq [\![\mathbf{new}\ \widetilde{z{:}\rho}.\ Q]\!] \text{ in } [\![\Delta]\!]}{\Pi_{i=1}^n [\![P_i]\!] \xrightarrow[i_1]{\infty} [\![\mathbf{new}\ \widetilde{z{:}\rho}]\!]\ ([\![Q]\!][\tilde{x}_{\mathscr{F}} \mapsto \tilde{y}_{\mathscr{F}}] \mid \Pi_{i=1, i\neq i_1}^n [\![P_i]\!])}$$

  By Lemma 5, this is $[\![\Pi_{i=1}^n P_i]\!] \xrightarrow[i_1]{\infty} [\![\mathbf{new}\ \widetilde{z{:}\rho}.\ (Q[\tilde{x} \mapsto \tilde{y}] \mid \Pi_{i=1, i\neq i_1}^n P_i)]\!]$.

- Rule (COM). The judgment has thus been inferred by an application of the communication rule:

$$\frac{C_{i_1}^{j_1} = x?f(\tilde{z}).\mathbf{new}\ \widetilde{x_1{:}\rho_1}.\ Q_1 \qquad C_{i_2}^{j_2} = x!f(\tilde{y}).\mathbf{new}\ \widetilde{x_2{:}\rho_2}.\ Q_2}{\Pi_{i=1}^n \sum_{j=1}^{m_i} C_i^j \xrightarrow[i_1,j_1,i_2,j_2]{\varrho(x)(f)} \left\{ \begin{array}{l} \mathbf{new}\ \widetilde{x_1{:}\rho_1}.\ \mathbf{new}\ \widetilde{x_2{:}\rho_2}. \\ (Q_1[\tilde{z} \mapsto \tilde{y}] \mid Q_2 \mid \Pi_{i=1, i\neq i_1, i_2}^n \sum_{j=1}^{m_i} C_i^j) \end{array} \right.}$$

  We can then apply the communication rule as follows too:

$$\frac{[\![C_{i_1}^{j_1}]\!] = x_f?(\tilde{z}_{\mathscr{F}}).[\![\mathbf{new}\ \widetilde{x_1{:}\rho_1}.\ Q_1]\!] \qquad [\![C_{i_2}^{j_2}]\!] = x_f!(\tilde{y}_{\mathscr{F}}).[\![\mathbf{new}\ \widetilde{x_2{:}\rho_2}.\ Q_2]\!]}{\Pi_{i=1}^n \sum_{j=1}^{m_i} [\![C_i^j]\!] \xrightarrow[i_1,j_1,i_2,j_2]{\varrho(x_f)} \left\{ \begin{array}{l} [\![\mathbf{new}\ \widetilde{x_1{:}\rho_1}.\ \mathbf{new}\ \widetilde{x_2{:}\rho_2}. \\ (Q_1[\tilde{z} \mapsto \tilde{y}] \mid Q_2 \mid \Pi_{i=1, i\neq i_1, i_2}^n \sum_{j=1}^{m_i} C_i^j)]\!] \end{array} \right.}$$

Lemma 7 *If* $[\![P]\!] \xrightarrow[w]{s} Q'$ *with respect to* $[\![\Delta]\!]$ *and* $[\![\varrho]\!]$ *then there exists* $Q \in [\![Q']\!]^{-1}$ *such that* $P \xrightarrow[w]{s} Q$ *with respect to* Δ *and* ϱ.

Proof. By rule induction. We need to consider the three rules (NEW), (APP), and (COM) by which to infer $[\![P]\!] \xrightarrow[w]{s} Q'$ with respect to $[\![\Delta]\!]$ and $[\![\varrho]\!]$.

- Rule (NEW). In this case, the above judgment was inferred as follows:

$$\frac{[\![P_1]\!] \xrightarrow[w]{s} Q_1' \qquad [\![\varrho]\!](x_{f_1}) = \rho(f_1) \qquad \ldots \qquad [\![\varrho]\!](x_{f_n}) = \rho(f_n)}{\begin{array}{l} [\![P]\!] = \mathbf{new}\ x_{f_1}{:}\rho(f_1).\ \cdots.\ \mathbf{new}\ x_{f_n}{:}\rho(f_n).\ [\![P_1]\!] \xrightarrow[w]{s} \\ Q' = \mathbf{new}\ x_{f_1}{:}\rho(f_1).\ \cdots.\ \mathbf{new}\ x_{f_n}{:}\rho(f_n).\ Q_1' \end{array}}$$

  By induction hypothesis, there exists $Q_1 \in [\![Q_1']\!]^{-1}$ such that $P_1 \xrightarrow[w]{s} Q_1$ with respect to Δ and ϱ. The hypotheses of the rule yield $\varrho(x) = \rho$. We can thus apply the (NEW)

rule as follows:

$$\frac{P_1 \xrightarrow[w]{s} Q_1 \qquad \varrho(x) = \rho}{\mathbf{new}\ x{:}\rho.\ P_1 \xrightarrow[w]{s} \mathbf{new}\ x{:}\rho.\ Q_1}$$

Lemma 6 shows that $[\![\mathbf{new}\ x{:}\rho.\ P_1]\!] \xrightarrow[w]{s} [\![\mathbf{new}\ x{:}\rho.\ Q_1]\!]$. This is equivalent to:

$$[\![P]\!] \xrightarrow[w]{s} \mathbf{new}\ x_{f_1}{:}\rho(f_1).\ \cdots .\ \mathbf{new}\ x_{f_n}{:}\rho(f_n).\ [\![Q_1]\!] = Q'$$

- Rule (APP). The judgment has in this case been inferred as follows:

$$\frac{[\![P_{i_1}]\!] = A(\tilde{y}_{\mathscr{F}}) \qquad A(\tilde{x}_{\mathscr{F}}) \triangleq [\![\mathbf{new}\ \widetilde{z{:}\rho}.\ Q_1]\!] \text{ in } [\![\Delta]\!]}{[\![P]\!] = \Pi_{i=1}^n [\![P_i]\!] \xrightarrow[i_1]{\infty} Q' = [\![\mathbf{new}\ \widetilde{z{:}\rho}]\!]\ ([\![Q_1]\!][\tilde{x}_{\mathscr{F}} \mapsto \tilde{y}_{\mathscr{F}}] \mid \Pi_{i=1, i\neq i_1}^n [\![P_i]\!])}$$

By Lemma 5, we have $[\![Q_1]\!][\tilde{x}_{\mathscr{F}} \mapsto \tilde{y}_{\mathscr{F}}] = [\![Q_1[\tilde{x} \mapsto \tilde{y}]]\!]$. Hence, $Q' = [\![Q]\!]$ where $Q = \mathbf{new}\ \widetilde{y{:}\rho}.\ (Q_1[\tilde{x} \mapsto \tilde{y}] \mid \Pi_{i=1, i\neq i_1}^n P_i)$. Furthermore, we can infer $P \xrightarrow[i_1]{\infty} Q$ as follows:

$$\frac{P_{i_1} = A(\tilde{y}) \qquad A(\tilde{x}) \triangleq \mathbf{new}\ \widetilde{y{:}\rho}.\ Q_1 \text{ in } \Delta}{P = \Pi_{i=1}^n P_i \xrightarrow[i_1]{\infty} Q = \mathbf{new}\ \widetilde{y{:}\rho}.\ (Q_1[\tilde{x} \mapsto \tilde{y}] \mid \Pi_{i=1, i\neq i_1}^n P_i)}$$

- Rule (COM). The judgment is now inferred as follows:

$$\frac{[\![C_{i_1}^{j_1}]\!] = x_f?(\tilde{z}_{\mathscr{F}}).[\![\mathbf{new}\ \widetilde{x_1{:}\rho_1}.\ Q_1]\!] \qquad [\![C_{i_2}^{j_2}]\!] = x_f!(\tilde{y}_{\mathscr{F}}).[\![\mathbf{new}\ \widetilde{x_2{:}\rho_2}.\ Q_2]\!]}{[\![P]\!] = \Pi_{i=1}^n \textstyle\sum_{j=1}^{m_i} [\![C_i^j]\!] \xrightarrow[i_1, j_1, i_2, j_2]{\varrho(x_f)} Q'}$$

where $Q' = [\![\mathbf{new}\ \widetilde{x_1{:}\rho_1}]\!]\ [\![\mathbf{new}\ \widetilde{x_2{:}\rho_2}]\!]\ ([\![Q_1]\!][\tilde{z}_{\mathscr{F}} \mapsto \tilde{y}_{\mathscr{F}}] \mid [\![Q_2]\!] \mid \Pi_{i=1, i\neq i_1, i_2}^n \sum_{j=1}^{m_i} [\![C_i^j]\!])$. The substitution Lemma 5 yields equality between $[\![Q_1]\!][\tilde{z}_{\mathscr{F}} \mapsto \tilde{y}_{\mathscr{F}}]$ and $[\![Q_1[\tilde{z} \mapsto \tilde{y}]]\!]$. Thus, $Q' = [\![Q]\!]$ where $Q = \mathbf{new}\ \widetilde{x_1{:}\rho_1}.\ \mathbf{new}\ \widetilde{x_2{:}\rho_2}.\ (Q_1[\tilde{z} \mapsto \tilde{y}] \mid Q_2 \mid \Pi_{i=1, i\neq i_1, i_2}^n \sum_{j=1}^{m_i} C_i^j)$. Hence, we can infer $P \xrightarrow[i_1, j_1, i_2, j_2]{\varrho(x)(f)} Q$ as follows:

$$\frac{C_{i_1}^{j_1} = x?f(\tilde{z}).\mathbf{new}\ \widetilde{x_1{:}\rho_1}.\ Q_1 \qquad C_{i_2}^{j_2} = x!f(\tilde{y}).\mathbf{new}\ \widetilde{x_2{:}\rho_2}.\ Q_2}{\Pi_{i=1}^n \textstyle\sum_{j=1}^{m_i} C_i^j \xrightarrow[i_1, j_1, i_2, j_2]{\varrho(x)(f)} \left\{ \begin{array}{l} \mathbf{new}\ \widetilde{x_1{:}\rho_1}.\ \mathbf{new}\ \widetilde{x_2{:}\rho_2}. \\ (Q_1[\tilde{z} \mapsto \tilde{y}] \mid Q_2 \mid \Pi_{i=1, i\neq i_1, i_2}^n \sum_{j=1}^{m_i} C_i^j) \end{array} \right.}$$

Lemma 8 *$P \xrightarrow[w]{s} Q$ with respect to Δ and ϱ iff $[\![P]\!] \xrightarrow[w]{s} [\![Q]\!]$ with respect to $[\![\Delta]\!]$ and $[\![\varrho]\!]$.*

Proof. The implication from the left to the right is shown by Lemma 6. For the converse assume $[\![P]\!] \xrightarrow[w]{s} [\![Q]\!]$ with respect to $[\![\Delta]\!]$ and $[\![\varrho]\!]$. By Lemma 7 there exists $R \in [\![[\![Q]\!]]\!]^{-1}$ such that $P \xrightarrow[w]{s} R$. Since $[\![\cdot]\!]$ is injective, it follows that $R = Q$ so that $P \xrightarrow[w]{s} Q$.

Lemma 9 *$P \xrightarrow{\infty(p)} Q$ with respect to Δ and ϱ iff $[\![P]\!] \xrightarrow{\infty(p)} [\![Q]\!]$ with respect to $[\![\Delta]\!]$ and $[\![\varrho]\!]$.*

Proof.

'$\Rightarrow$' Judgments $P \xrightarrow{\infty(p)} Q$ with respect to Δ and ϱ are derived by rule (COUNT):

$$\frac{P \equiv P' \qquad \begin{array}{l} n = \sharp\{w \in \mathbb{N} \cup \mathbb{N}^4 \mid P' \xrightarrow[w]{\infty} R' \equiv Q\} \neq 0 \\ m = \sharp\{w \in \mathbb{N} \cup \mathbb{N}^4 \mid P' \xrightarrow[w]{\infty} R'\} \end{array}}{P \xrightarrow{\infty(n/m)} Q} \tag{2}$$

The corresponding judgment can be derived as follows by rule (COUNT):

$$\frac{[\![P]\!] \equiv [\![P']\!] \qquad \begin{array}{l} n = \sharp\{w \in \mathbb{N} \cup \mathbb{N}^4 \mid [\![P']\!] \xrightarrow[w]{\infty} R \equiv [\![Q]\!]\} \neq 0 \\ m = \sharp\{w \in \mathbb{N} \cup \mathbb{N}^4 \mid [\![P']\!] \xrightarrow[w]{\infty} R\} \end{array}}{[\![P]\!] \xrightarrow{\infty(n/m)} [\![Q]\!]} \tag{3}$$

To see this, we must show that $\exists R'.P' \xrightarrow[w]{\infty} R' \equiv Q$ is equivalent to $\exists R.[\![P']\!] \xrightarrow[w]{\infty} R \equiv [\![Q]\!]$. This follows from Lemmas 5, 6, and 7.

'$\Leftarrow$' The judgment $[\![P]\!] \xrightarrow{\infty(n/m)} [\![Q]\!]$ must be inferred as follows:

$$\frac{[\![P]\!] \equiv P_1 \qquad \begin{array}{l} n = \sharp\{w \in \mathbb{N} \cup \mathbb{N}^4 \mid P_1 \xrightarrow[w]{\infty} R \equiv [\![Q]\!]\} \neq 0 \\ m = \sharp\{w \in \mathbb{N} \cup \mathbb{N}^4 \mid P_1 \xrightarrow[w]{\infty} R\} \end{array}}{[\![P]\!] \xrightarrow{\infty(n/m)} [\![Q]\!]}$$

Since $[\![P]\!] \equiv P_1$ we can apply Lemma 5 which yields the existence of $P' \in [\![P_1]\!]^{-1}$ such that $P \equiv P'$. With this P' the inference step (3) becomes valid, so that we can infer $P \xrightarrow{\infty(n/m)} Q$ with respect to Δ and ϱ as in (2).

Lemma 10 *$P \xrightarrow{r} Q$ with respect to Δ and ϱ iff $[\![P]\!] \xrightarrow{r} [\![Q]\!]$ with respect to $[\![\Delta]\!]$ and $[\![\varrho]\!]$.*

Proof.

'$\Rightarrow$' The judgment $P \xrightarrow{r} Q$ with respect to Δ and ϱ is inferred by rule (SUM).

$$\frac{P \equiv P' \qquad r = \sum_{P' \xrightarrow[w]{r'} R' \equiv Q} r' \neq 0 \qquad \neg\exists R''\exists w'.\ P' \xrightarrow[w']{\infty} R''}{P \xrightarrow{r} Q} \tag{4}$$

The corresponding judgment can be inferred as follows:

$$\frac{[\![P]\!] \equiv [\![P']\!] \qquad r = \sum_{[\![P']\!] \xrightarrow[w]{r'} R \equiv [\![Q]\!]} r' \neq 0 \qquad \neg\exists R''\exists w'.\ [\![P']\!] \xrightarrow[w']{\infty} R''}{[\![P]\!] \xrightarrow{r} [\![Q]\!]} \tag{5}$$

To see this, it is sufficient to show two equivalences, both following from Lemmas 5, 6, and 7:

1. $\exists R'.P' \xrightarrow[w]{r'} R' \equiv Q$ iff $\exists R.[\![P']\!] \xrightarrow[w]{r'} R \equiv [\![Q]\!]$.
2. $\exists R'.P' \xrightarrow[w]{\infty} R'$ iff $\exists R.[\![P']\!] \xrightarrow[w]{\infty} R$.

'$\Leftarrow$' The judgment $[\![P]\!] \xrightarrow{r} [\![Q]\!]$ must be inferred as follows:

$$\frac{[\![P]\!] \equiv P_1 \qquad r = \sum_{P_1 \xrightarrow[w]{r'} R \equiv [\![Q]\!]} r' \neq 0 \qquad \neg\exists R''\exists w'.\ P_1 \xrightarrow[w']{\infty} R''}{[\![P]\!] \xrightarrow{r} [\![Q]\!]}$$

Since $[\![P]\!] \equiv P_1$, Lemma 5 yields the existence of $P' \in [\![P_1]\!]^{-1}$ such that $P \equiv P'$. With this P' the inference step (5) becomes valid, so that we can infer $P \xrightarrow{r} Q$ with respect to Δ and ϱ as in (4).

Conclusion and Future Work

We have presented SPICO, a new modeling language for system biology on higher level on abstraction. SPICO provides multi-profile objects with static inheritance. It supports the paradigm of modeling "molecules as concurrent objects". The core of SPICO is a stochastic π-calculus with input patterns. We have presented the stochastic semantics of this language in terms of CTMCs and shown how to compile it into the biochemical stochastic π-calculus, so that the semantics is preserved. In future work, we hope to finalize SPICO's language specification and to provide an implementation.

References

[BDG94] M. Bernardo, L. Donatiello, and R. Gorrieri. MPA: A stochastic process algebra. Technical Report UBLCS-94-10, University of Bologna, Computer Science Laboratory, 1994.

[BPV05] Michael Baldamus, Joachim Parrow, and Björn Victor. A fully abstract encoding of the π-calculus with data terms. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *Automata, Languages and Programming, 32nd International Colloquium, ICALP 2005, Lisbon, Portugal, July 11-15, 2005, Proceedings*, volume 3580 of *Lecture Notes in Computer Science*, pages 1202–1213. Springer, 2005.

[Car05] Luca Cardelli. Brane calculi: interactions of biological membranes. In *Proceedings of CMSB 2004*, volume 3082 of *Lecture Notes in Bioinformatics*, pages 257–278, 2005.

[CP06] Federica Ciocchetta and Corrado Priami. Biological transactions for quantitative models. In *Proceedings of MeCBIC 2006*, Electronic Notes in Theoretical Computer Science. Elsevier, 2006. to appear.

[CRFS05] Nathalie Chabrier-Rivier, Francois Fages, and Sylvain Soliman. The biochemical abstract machine BioCham. In *Proceedings of CMSB 2004*, volume 3082 of *Lecture Notes in Bioinformatics*, pages 172–191, 2005.

[DK05] Denys Duchier and Céline Kuttler. Biomolecular agents as multi-behavioural concurrent objects. In *Proceedings of the First International Workshop on Methods and Tools for Coordinating Concurrent, Distributed and Mobile Systems*, volume 150 of *Electronical notes in theoretical computer science*, pages 31–49. Elsevier, 2005.

[Gil76] Daniel T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics*, 22:403–434, 1976.

[Her02] Holger Hermanns. *Interactive Markov Chains: The Quest for Quantified Quality*, volume 2428 of *Lecture Notes in Computer Science*. Springer, 2002.

[HFS+03] M Hucka, A Finney, HM Sauro, H Bolouri, JC Doyle, H Kitano, AP Arkin, BJ Bornstein, D Bray, A Cornish-Bowden, AA Cuellar, S Dronov, ED Gilles, M Ginkel, V Gor, II Goryanin, WJ Hedley, TC Hodgman, JH Hofmeyr, PJ Hunter, NS Juty, JL Kasberger, A Kremling, U Kummer, N Le Novere, LM Loew, D Lucio, P Mendes, E Minch, ED Mjolsness, Y Nakayama, MR Nelson, PF Nielsen, T Sakurada, JC Schaff, BE Shapiro, TS Shimizu, HD Spence andJ Stelling, K Takahashi, M Tomita, J Wagner, and J Wang. The systems

biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, 19:524–531, 2003.

[Hil95] Jane Hillston. *A Compositional Approach to Performance Modelling*. PhD thesis, University of Edinburgh, 1995. Distinguished Dissertations Series. Cambridge University Press, 1996.

[Hon92] Kohei Honda. Two bisimilarities in ν-calculus. Technical Report 92-002, Keio University, Department of Computer Science, 1992.

[KN06] Céline Kuttler and Joachim Niehren. Gene regulation in the pi calculus: Simulating cooperativity at the lambda switch. In Corrado Priami, Anna Ingólfsdóttir, Hanne Riis Nielson, and Bud Mishra, editors, *Transactions on Computational Systems Biology VII*, volume 4230 of *Lecture Notes in Bioinformatics*, pages 24–55. Springer, 2006. A preliminary version was presented at the Second International Workshop on Concurrent Models in Molecular Biology (BioConcur 2004).

[KNP+06] M. Kwiatkowska, G. Norman, D. Parker, O. Tymchyshyn, J. Heath, and E. Gaffney. Simulation and verification for computational modelling of signalling pathways. In L. Perrone, F. Wieland, J. Liu, B. Lawson, D. Nicol, and R. Fujimoto, editors, *Proc. 2006 Winter Simulation Conference*, 2006. To appear.

[Kut06a] Céline Kuttler. Bacterial transcription and translation in the pi calculus. *Transactions on Computational Systems Biology*, 4220(VI):113–149, 2006.

[Kut06b] Céline Kuttler. *Modeling Bacterial Gene Expression in a Stochastic Pi Calculus with Concurrent Objects*. PhD thesis, Université des Sciences et Technologies de Lille - Lille 1, 2006.

[LPQ+04] Paola Lecca, Corrado Priami, Paola Quaglia, B. Rossi, C. Laudanna, and G. Constantin. A stochastic process algebra approach to simulation of autoreactive lymphocyte recruitment. *SCS Simulation*, 80(6):273–288, June 2004.

[MPW92] Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes (I and II). *Information and Computation*, 100:1–77, 1992.

[PC06] Andrew Phillips and Luca Cardelli. A correct abstract machine for the stochastic pi-calculus. In Corrado Priami, Anna Ingolfsdottir, Hanne Riis Nielson, and Bud Mishra, editors, *Transactions on Computational Systems Biology VII*, volume 4230. Springer, 2006.

[PML+03] Hervé Paulino, Pedro Marques, Luis Lopes, Vasco T. Vasconcelos, and Fernando Silva. A multi-threaded asynchronous language. In *7th International Conference on Parallel Computing Technologies*, volume 2763 of *Lecture Notes in Computer Science*, pages 316–323. Springer, 2003.

[PQ05] Corrado Priami and Paola Quaglia. Beta binders for biological interactions. In *Proceedings of CMSB 2004*, volume 3082 of *Lecture Notes in Bioinformatics*, pages 20–33, 2005.

[Pri95] Corrado Priami. Stochastic π-calculus. *Computer Journal*, 6:578–589, 1995.

[PRSS01] Corrado Priami, Aviv Regev, Ehud Shapiro, and William Silverman. Application of a stochastic name-passing calculus to representation and simulation of molecular processes. *Information Processing Letters*, 80:25–31, 2001.

[RPS+04] Aviv Regev, E. M. Panina, William Silverman, Luca Cardelli, and Ehud Shapiro. BioAmbients: An abstraction for biological compartments. *Theoretical Computer Science*, 325(1):141–167, 2004.

[RS02] Aviv Regev and Ehud Shapiro. Cells as computation. *Nature*, 419:343, 2002.

[RV00] António Ravara and Vasco T. Vasconcelos. Typing non-uniform concurrent objects. In *CONCUR'00*, volume 1877 of *Lecture Notes in Computer Science*, pages 474–488. Springer, 2000.

[VT93] Vasco T. Vasconcelos and Mario Tokoro. A typing system for a calculus of objects. In *1st International Symposium on Object Technologies for Advanced Software*, volume 472 of *Lecture Notes in Computer Science*, pages 460–474. Springer, 1993.

Unité de recherche INRIA Futurs
Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)

Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399