A Stochastic Pi Calculus for Concurrent Objects

Céline Kuttler, Cédric Lhoussaine, Joachim Niehren

A Stochastic Pi Calculus for Concurrent Objects

Céline Kuttler[1], Cédric Lhoussaine[2], and Joachim Niehren[3]

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

Abstract. We present SpiCO, a new modeling and simulation language for systems biology. SpiCO is based on the stochastic π-calculus. It 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.

1 Introduction

A central objective of systems biology is the investigation of the dynamics in living cells, that arises from interactions between its molecular components. Modeling and simulation increasingly complement knowledge acquisition through experimentation. Discrete event based approaches are advantageous with respect to detailed cellular control by small numbers of molecular actors. Deterministic approaches offer benefits when modeling large populations.

Regev and co-authors [22] proposed to apply the stochastic π-calculus as a modeling language for systems biology, based on Priami's [20] refinement of the synchronous π-calculus [17] by a notion of time. Expression in the π-calculus then abstract chemical solutions, in which molecules interact concurrently. As for earlier stochastic process algebras [9], stochastic parameters impose exponential distributions of waiting times on reaction. Thus π-calculus expressions give rise to continuous time Markov chains (CTMCs). Their execution yields stochastic simulation, based on Gillespie's algorithm [7].

Both existing simulation engines for the (biochemical) stochastic π-calculus– SPiM [19] and BioSpi [22] – have been applied in case studies of small to medium size [11, 15, 16]. Alternative modeling languages as BioCham [4] or SBML [10] 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 [11], tricky 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*. SpiCO was indeed developed concomitantly with modeling case studies [12, 13]. 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 Core SpiCO, a novel stochastic π-calculus with input patterns, that originate from the distributed programming language TyCO [18, 25] for typed concurrent objects in the asynchronous π-calculus. SpiCO assigns stochastic rates to pairs of channel and function names.
2. We define a *stochastic semantics* assigning CTMCs to SpiCO's process expressions, carefully distinguising 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 [19, 22], or disregard immediate reactions [20] essential for expressiveness and modeling. Experiments with SPiM confirm a correct treatment in implementations nevertheless.
3. We identify multi-profile objects with expressions in Core SpiCO. Each profile comes with its own interface, similarly as TyCO's non-uniform objects [23]. 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 to model mutual exclusion of molecular events, as frequently encountered in cellular regulation. Its essence lies in escaping inconsistent intermediate states by immediate reactions, that are applied before timed ones. This solves tedious *atomicity* problems, without introducing transactions [5].
6. We encode SpiCO back into the biochemical stochastic π-calculus, so that we can run SpiCO programs in SPiM or BioSpi. The main challenge is to encode input patterns, while preserving the stochastic semantics.

In previous work we proposed a first ad hoc abstraction of interacting molecules as objects that switch between discrete states [6]. How to explicitely support objects with multiple profiles in a more conservative language with proper syntax and semantics remained open.
Other languages for systems biology were recently proposed. Beta binders [21] are inspired by the π-calculus, but enable interactions by type coincidence rather than channel name equality. Others [3, 24] adress spatial aspects at membranes.

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. For space considerations, the proofs are not included here but can be found in [14].

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

Table 1. Syntax of Core SpiCO

2 A Stochastic Pi-calculus with Input Patterns

The core of SpiCO (Core SpiCO) consists in a novel stochastic π-calculus with *input patterns*, a linguistic feature introduced by Vasconcelos and Tokoro for typed concurrent objects in the asynchronous π-calculus (TyCO) [18, 25]. Input patterns are motivated by pattern matching in functional programming languages of the ML family. In TyCO, they are closely tied to communication: objects only receive tuples if they provide a matching input pattern.

Core SpiCO's vocabulary consists in an infinite set of *channel names* $\mathcal{N} = \{x, y, z, \ldots\}$, a set of *process names* A, and a set of *function names* $f \in \mathscr{F}$. Process and function names have fixed arities. 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).

Table 1 defines the syntax of Core SpiCO. We write $\tilde{x}$ for finite, possibly empty sequences of channels $x_1, \ldots, x_n$ where $n \geq 0$. When using tuples $f(\tilde{x})$ or terms $A(\tilde{x})$ the number of arguments (the length of $\tilde{x}$) is assumed equal to the respective arity of f or A. Process expressions are ranged over by P. The only atomic expression (not decomposable 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** $x{:}\rho.\ P$ introduces 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* over x a tuple constructed by f. The channels $\tilde{y}$ in input guards serve as pattern variables; these bound variables 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 a 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. For modeling convenience, we permit free channel names in P besides the

$(P_1 \mid P_2) \mid P_3 \equiv P_1 \mid (P_2 \mid P_3)$	$P_1 \mid P_2 \equiv P_2 \mid P_1$
$\ldots + C_1 + C_2 + \ldots \equiv \ldots + C_2 + C_1 + \ldots$	$P \mid \mathbf{0} \equiv P$
$\mathbf{new}\ x{:}\rho.\ (P_1 \mid P_2) \equiv P_1 \mid \mathbf{new}\ x{:}\rho.\ P_2$ if $x \notin fv(P_1)$	$P_1 \equiv P_2$ if $P_1 \equiv_\alpha P_2$
$\mathbf{new}\ x_1{:}\rho_1.\ \mathbf{new}\ x_2{:}\rho_2.\ P \equiv \mathbf{new}\ x_2{:}\rho_2.\ \mathbf{new}\ x_1{:}\rho_1.\ P$	if $x_1 \neq x_2$

Table 2. Axioms of the structural congruence

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:

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

Table 3. Reduction relation for a finite set of definitions Δ

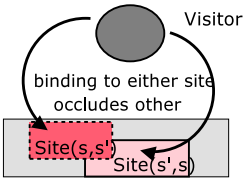
parameters in $\tilde{x}$. The set of *free channel names* for processes P and guarded processes C are denoted by $fv(P)$ and $fv(C)$ respectively. There are three scope baring constructs: new binder $\mathbf{new}\ x{:}\rho.\ P$, input patterns $\_?f(\tilde{x}).P$, and definitions $A(\tilde{x}) \triangleq P$.

We define an (non-stochastic) operational semantics for the π-calculus in terms of a binary relation between expressions, the so called (one step) reduction. We will later refine it to a ternary relation adding stochastic labels. The reduction relation is closed under the usual structural congruence (Table 2) between expressions.

Table 3 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 [1].

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 $\tilde{x}$.



(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***

Fig. 1. Overlapping sites located at s and s'

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.

The syntax of the biochemical stochastic π-calculus is the same as ours except for function names, and our more flexible assignment of stochastic rates. 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:

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

3 Molecular Binding at Overlapping Sites

We illustrate the modeling power of SpiCO by a frequent control mechanism between molecular interactions, as binding of molecules: mutual exclusion [11, 12]. Consider overlapping sites allowing for a unique visitor at a time - i.e. overlapping *semaphores*.

Figure 1(a) illustrates two such overlapping sites s and s'. Each can be either free, bound, or blocked. Only a free site can become bound by a visitor - while blocking the peer. We model sites as multi-profile object with three profiles Site_free, Site_bound, and Site_blocked. Figure 2 presents their definitions 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, 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 tran-

```
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
```

Fig. 2. Overlapping sites module

sitions 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 4). This way, we solve a tedious atomicity problems while avoiding heavier extensions of the π-calculus by transactions [5].

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:

$$\begin{array}{rl} & \mathsf{Site\_free(s,s')} \mid \mathsf{Site\_free(s',s)} \mid \mathsf{Visitor\_free} \mid \mathsf{Visitor\_free} \\ \rightarrow & \mathsf{Site\_bound(s,s')} \mid \mathsf{Site\_free(s',s)} \mid \mathsf{Visitor\_free} \mid \mathsf{Visitor\_at(s)} \\ \not\rightarrow & \mathsf{Site\_bound(s,s')} \mid \mathsf{Site\_bound(s',s)} \mid \mathsf{Visitor\_at(s)} \mid \mathsf{Visitor\_at(s')} \end{array}$$

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 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*

```
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()
```

Fig. 3. Visitors module for sites s and s'

```
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)
```

Fig. 4. Promoters inherit from overlapping sites

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 biological systems is model extensibility by object inheritance. Numerous examples are elaborated in [12]. 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:

$$\begin{aligned} &\mathsf{Obj2}\ \textbf{extends}\ \mathsf{Obj} \\ &\mathsf{Obj2\_p}_1(\tilde{z_1})\ \textbf{extended by}\ \mathsf{C}^1_{k_1+1} + \ldots + \mathsf{C}^1_{l_1} \\ &\quad \ldots \\ &\mathsf{Obj2\_p}_n(\tilde{z_n})\ \textbf{extended by}\ \mathsf{C}^n_{k_n+1} + \ldots + \mathsf{C}^n_{l_n} \end{aligned}$$

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

$$\begin{aligned} &\mathsf{Obj2\_p}_1(\tilde{z_1}) \triangleq \mathsf{C}^1_1 + \ldots + \mathsf{C}^1_{l_1}\ [\mathsf{Obj} \mapsto \mathsf{Obj2}] \\ &\quad \ldots \\ &\mathsf{Obj2\_p}_n(\tilde{z_n}) \triangleq \mathsf{C}^n_1 + \ldots + \mathsf{C}^n_{l_n}\ [\mathsf{Obj} \mapsto \mathsf{Obj2}] \end{aligned}$$

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 (CTSP), with memoryless sojourn times for all states. More formally, a CTMC over S is a CTSP $\{X_t \mid t \in \mathbb{R}^+\}$ 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" [8]. 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) [7] 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 [7]. 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:

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

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

6 Stochastic Semantics of Core SpiCO

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 [20] 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 [20], the author proposes a *labeled 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 style for defining 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 [20], our semantics takes into account immediate transitions of 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 this 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

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
$1 \leq i_1 \neq i_2 \leq n,\ 1 \leq j_1 \leq m_{i_1},\ 1 \leq j_2 \leq m_{i_2}$

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

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

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

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 4. 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.

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 $P = \mathbf{new}\ \widetilde{x{:}\rho}.\ (P_1 \mid \ldots \mid P_m)$ 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 pnfs after removing top-level **new**-binders are multisets of sums and applications. All applications must have been reduced before time consuming transitions can apply, so we have a multiset of sums. Each sum is like a molecule, except that each of its choices offers its own interactions.

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:

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

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 pnf.

We merge labeled immediate transitions with rule (COUNT). Although being immediate we want to associate probabilities, which characterize the number

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

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

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

Table 5. Elimination of immediate transitions and merging timed transitions

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:

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

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[4]:

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

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 5. The idea is quite similar to that of [2]: the transitions are obtained by integrating immediate transitions into time consuming transitions. An example for this transformation is as follows:

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

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

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 $\xrightarrow{r}$.

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[5] 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 $\xrightarrow{r}$ are identical to those of $\overset{r}{\Longrightarrow}$. However, transition probabilities can be compared only for processes performing timed transitions. 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:

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

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

- *(Timed correctness) $\downarrow P = \Downarrow P$,*
- *(Probabilistic correctness) $Pr(P \twoheadrightarrow Q) = Pr(P \Rightarrow Q)$.*

7 Encoding Input Patterns

We now encode SPICO 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): $\mathscr{F}' = \{\text{UNIT}_i \mid 0 \leq i \leq max\}$. In what follows, we write UNIT instead of UNIT_i.

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

[5] 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.

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

Table 6. Encoding of input patterns

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 6.

The following theorem states the correctness of our encoding. It allows us to run simulations of models expressed in SpiCO, via an implementation of the original stochastic π-calculus, as implemented in the SPiM system [19].

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. $\varrho'(x_f) = \varrho'(x_f)(\text{UNIT})$.

8 Conclusion and Future Work

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

References

1. M. Baldamus, J. Parrow, and B. Victor. A fully abstract encoding of the π-calculus with data terms. In *ICALP. LNCS* 3580:1202–1213. 2005.

2. M. Bernardo, L. Donatiello, and R. Gorrieri. MPA: A stochastic process algebra. Technical Report UBLCS-94-10, University Bologna, 1994.
3. L. Cardelli. Brane calculi: interactions of biological membranes. In *CMSB 2004*, 3082 of *LNBI*, 257–278, 2005.
4. N. Chabrier-Rivier, F. Fages, and S. Soliman. The biochemical abstract machine BioCham. In *CMSB 2004*, volume 3082 of *LNBI*, 172–191, 2005.
5. F. Ciocchetta and C. Priami. Biological transactions for quantitative models. In *MeCBIC. ENTCS*. 2006. to appear.
6. D. Duchier and C. Kuttler. Biomolecular agents as multi-behavioural concurrent objects. In *Proc. MTCoord*, vol 150 of *ENTCS*, 31–49, 2005.
7. D. T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J Comp Phys*, 22:403–434, 1976.
8. H. Hermanns. *Interactive Markov Chains: The Quest for Quantified Quality*, *LNCS* 2428. Springer, 2002.
9. J. Hillston. *A Compositional Approach to Performance Modelling*. PhD thesis, University of Edinburgh, 1995. Cambridge University Press, 1996.
10. M. Hucka et. al. The systems biology markup language (SBML). *Bioinformatics*, 19:524–531, 2003.
11. C. Kuttler and J. Niehren. Gene regulation in the pi calculus: Simulating cooperativity at the lambda switch. In *Trans Comp Syst Bio*, *LNBI* 4230:24–55. 2006.
12. C. Kuttler. Bacterial transcription and translation in the pi calculus. *Trans Comp Syst Biol VI*, *LNBI* 4220:113–149, 2006.
13. C. Kuttler. *Modeling Bacterial Gene Expression in a Stochastic Pi Calculus with Concurrent Objects*. PhD thesis, University Lille 1, 2006.
14. C. Kuttler, C. Lhoussaine, and J. Niehren. A stochastic pi-calculus for concurrent objects. *INRIA technical report* 6076, 2006.
15. M. Kwiatkowska, G. Norman, D. Parker, O. Tymchyshyn, J. Heath, and E. Gaffney. Simulation and verification for computational modelling of signalling pathways. *Winter Simulation Conference*, 2006. To appear.
16. P. Lecca, C. Priami, P. 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.
17. R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes (I and II). *Information and Computation*, 100:1–77, 1992.
18. H. Paulino, P. Marques, L. Lopes, V. T. Vasconcelos, and F. Silva. A multithreaded asynchronous language. In *PaCT*, *LNCS* 2763:316–323. 2003.
19. A. Phillips and L. Cardelli. A correct abstract machine for the stochastic pi-calculus. *Proc. Workshop on Concurrent Models in Molecular Biology*, 2004.
20. C. Priami. Stochastic π-calculus. *Computer Journal*, 6:578–589, 1995.
21. C. Priami and P. Quaglia. Beta binders for biological interactions. In *Proc. of CMSB 2004*, volume 3082 of *LNBI*, 20–33, 2005.
22. C. Priami, A. Regev, E. Shapiro, and W. Silverman. Application of a stochastic name-passing calculus to representation and simulation of molecular processes. *Information Processing Letters*, 80:25–31, 2001.
23. A. Ravara and V. T. Vasconcelos. Typing non-uniform concurrent objects. In *CONCUR*, *LNCS* 1877:474–488. 2000.
24. A. Regev, E. M. Panina, W. Silverman, L. Cardelli, and E. Shapiro. BioAmbients: An abstraction for biological compartments. *TCS*, 325(1):141–167, 2004.
25. V. T. Vasconcelos and M. Tokoro. A typing system for a calculus of objects. In *ISOTAS*, *LNCS* 472:460–474. 1993.