Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
Electronic Notes in Theoretical Computer Science 227 (2009) 143–159
Electronic Notes in Theoretical Computer Science
www.elsevier.com/locate/entcs

An Abstract Machine for the Stochastic Bioambient calculus

Andrew Phillips[1]

Microsoft Research
7 JJ Thomson Avenue
Cambridge, UK

Abstract

This paper presents an abstract machine for the stochastic bioambient calculus. The abstract machine is proved sound and complete with respect to a novel stochastic semantics, and is also shown to preserve the reduction probabilities of the calculus. The machine is implemented as an extension to an existing simulator for stochastic pi-calculus.

Keywords: abstract machine, stochastic, bioambient calculus, correctness, implementation.

1 Introduction

Since the Gillespie algorithm was first introduced in the 1970's [5], a vast number of chemical and biological models have been simulated using this approach. Even today, many biological models are published as collections of chemical reactions, and the format is supported by modern modelling standards such as the Systems Biology Markup Language. Many published models consist of hundreds of reactions and, as our knowledge of biological systems continues to increase, models consisting of tens of thousands of reactions will soon be commonplace. Clearly, writing a model of this size as a list of reactions is not a scalable solution, for the same reasons that we would not write a large software program as a single list of thousands of instructions. Rather, we need a way of decomposing a large biological model into a collection of smaller modules, to facilitate model readability, maintenance and extension. A promising approach for achieving this decomposition is through the use of concurrent modelling languages. One such language is the stochastic pi-calculus [11], which allows a system of chemical reactions to be written as a collection of parallel processes that interact with each other by message passing.

[1] Email: andrew.phillips@microsoft.com

doi:10.1016/j.entcs.2008.12.109

In addition to enabling modular decomposition, the stochastic pi-calculus supports parameterised processes that can be used to define generic building blocks such as gene gates, which can then be instantiated to build systems of arbitrary complexity [1]. Furthermore, the calculus allows complexes of arbitrary size to be formed by means of dynamically generated channels, where such complexation primitives have been shown to be more expressive than the language of chemical reactions [4]. While the stochastic pi-calculus affords improved modularity and abstraction of biological models, it is equally important to ensure that these models are accurately simulated. To this end, previous work defined an abstract machine for the stochastic pi-calculus [9,10], which described how a process of the calculus can be exactly simulated in a way that remains faithful to the original Gillespie algorithm [5].

More recently, there has been substantial work on extending the stochastic pi-calculus with constructs that model the complex spatial hierarchy of a biological system. In particular, the bioambient calculus [12] was introduced to model the notion of mobile compartments, and the brane calculus [3] was introduced to model the notion of mobile membranes. As with the stochastic pi-calculus, in order for these abstractions to be useful in practice it is essential that the corresponding models are simulated exactly, in accordance with the principles of the Gillespie algorithm. A simulator of the bioambient calculus has already been developed [12], which continues to be a valuable tool for validating stochastic bioambient models. However, there is a significant gap between the definition of the calculus and the definition and implementation of the underlying simulation algorithm. One could argue that the accuracy of the simulation algorithm is just as important as the calculus itself, and should therefore be treated with the same rigour.

This paper presents an abstract machine for the stochastic bioambient calculus. First we present the stochastic calculus together with a simple example. We then present the bioambient machine, followed by an outline of a proof of correctness.

2 The Stochastic Bioambient Calculus

The bioambient calculus was presented in [12] as a means of modelling mobile compartments in biological processes. This section presents a compact syntax for the calculus, together with a novel stochastic semantics.

2.1 Syntax

The syntax of the stochastic bioambient calculus (SBA) used in this paper is presented in Definition 2.1. A process P can be a choice of actions M, an instance $X(\tilde{n})$ of a definition X with parameters $\tilde{n}$, a parallel composition of processes $P \mid Q$, a process $\nu x\, P$ with a private channel x, or an ambient $\boxed{P}$ consisting of a process P inside a compartment. A choice M consists of a competition between zero or more actions $\pi^i.P$, where π is the action that can be performed, after which process P is executed, and i is an index used for identifying individual actions. An action π can

$E ::=$	$\emptyset$	Empty	$\pi ::=$	τ_r	Delay
	$\mid E, X(\tilde{m}) = P$	Process		$\mid \gamma!x(\tilde{n})$	Send
				$\mid \gamma?x(\tilde{m})$	Receive
$P, Q ::=$	M	Choice		$\mid \mu!x$	Move
	$\mid X(\tilde{n})$	Instance		$\mid \mu?x$	Accept
	$\mid P \mid Q$	Parallel	$\gamma ::=$	$\texttt{local}$	Local
	$\mid \nu x\, P$	Restriction		$\mid \texttt{s2s}$	Sibling
	$\mid \boxed{P}$	Ambient		$\mid \texttt{c2p}$	Parent
				$\mid \texttt{p2c}$	Child
$M, N ::=$	$\mathbf{0}$	Null	$\mu ::=$	$\texttt{in}$	Enter
	$\mid \pi^i.P + M$	Action		$\mid \texttt{out}$	Leave
				$\mid \texttt{merge}$	Merge

Definition 2.1 *Syntax of* SBA. *A system $E \vdash P$ consists of a constant environment E of definitions, together with a process P.*

(1) $$\tau_r^i.P + M \xrightarrow{r,i} P$$

(2) $$!x(\tilde{n})^i.P + M \mid ?x(\tilde{m})^j.P' + M' \xrightarrow{\rho(x),(i,j)} P \mid P'_{\{\tilde{n}/\tilde{m}\}}$$

(3) $$\boxed{Q \mid \texttt{c2p}!x(\tilde{n})^i.P + M} \mid Q' \mid \texttt{c2p}?x(\tilde{m})^j.P' + M' \xrightarrow{\rho(x),(i,j)} \boxed{Q \mid P} \mid Q' \mid P'_{\{\tilde{n}/\tilde{m}\}}$$

(4) $$Q \mid \texttt{p2c}!x(\tilde{n})^i.P + M \mid \boxed{Q' \mid \texttt{p2c}?x(\tilde{m})^j.P' + M'} \xrightarrow{\rho(x),(i,j)} Q \mid P \mid \boxed{Q' \mid P'_{\{\tilde{n}/\tilde{m}\}}}$$

(5) $$\boxed{Q \mid \texttt{s2s}!x(\tilde{n})^i.P + M} \mid \boxed{Q' \mid \texttt{s2s}?x(\tilde{m})^j.P' + M'} \xrightarrow{\rho(x),(i,j)} \boxed{Q \mid P} \mid \boxed{Q' \mid P'_{\{\tilde{n}/\tilde{m}\}}}$$

(6) $$\boxed{Q \mid \texttt{in}!x^i.P + M} \mid \boxed{Q' \mid \texttt{in}?x^j.P' + M'} \xrightarrow{\rho(x),(i,j)} \boxed{\boxed{Q \mid P} \mid Q' \mid P'}$$

(7) $$\boxed{\boxed{Q \mid \texttt{out}!x^i.P + M} \mid Q' \mid \texttt{out}?x^j.P' + M'} \xrightarrow{\rho(x),(i,j)} \boxed{Q \mid P} \mid \boxed{Q' \mid P'}$$

(8) $$\boxed{Q \mid \texttt{merge}!x^i.P + M} \mid \boxed{Q' \mid \texttt{merge}?x^j.P' + M'} \xrightarrow{\rho(x),(i,j)} \boxed{Q \mid P \mid Q' \mid P'}$$

(9) $$P \xrightarrow{r,w} P' \quad \Rightarrow \quad \boxed{P} \xrightarrow{r,w} \boxed{P'}$$

(10) $$P \xrightarrow{r,w} P' \quad \Rightarrow \quad \nu x\, P \xrightarrow{r,w} \nu x\, P'$$

(11) $$P \xrightarrow{r,w} P' \quad \Rightarrow \quad P \mid Q \xrightarrow{r,w} P' \mid Q$$

(12) $$Q \equiv P \xrightarrow{r,w} P' \equiv Q' \quad \Rightarrow \quad Q \xrightarrow{r,w} Q'$$

Definition 2.2 *Reduction rules of* SBA, *where a reaction identifier w can be an index i or a pair of indices (i, j).*

be a delay τ_r of rate r, a send $\gamma!x(\tilde{n})$ of values $\tilde{n}$ on channel x, or a receive $\gamma?x(\tilde{m})$ of values $\tilde{m}$ on channel x, where γ denotes the type of communication. This can be inside the same ambient (`local`), from one sibling ambient to another (`s2s`), from a child ambient to its parent (`c2p`) or from a parent ambient to a child (`p2c`). In addition, an action π can be a move $\mu!x$ on channel x or an accept $\mu?x$ on channel x, where μ denotes the type of movement. This can be an ambient entering one of its siblings (`in`), a child ambient leaving its parent (`out`) or a merge of two sibling ambients (`merge`).

An environment E consists of a set of definitions $X(\tilde{m}) = P$, where X is the name of the definition, $\tilde{m}$ are its parameters and P is the corresponding process. It is assumed that $\text{fn}(P) \subseteq \tilde{m}$, where $\text{fn}(P)$ denotes the set of free names of P, given that $\nu x\, P$ binds the name x in P and $\gamma?x(\tilde{m}).P$ binds the set of names $\tilde{m}$ in P. It is also assumed that recursive definitions in the environment are *guarded*, such that for a given definition $X(\tilde{m}) = P$, any recursive call to X inside P can only occur after an action π. We abbreviate $\texttt{local}!x(\tilde{n})$ to $!x(\tilde{n})$ and $\texttt{local}?x(\tilde{m})$ to $?x(\tilde{m})$, and we also abbreviate π^i to π in cases where the index i is unused. Stochastic behaviour is introduced into the calculus by associating each delay τ_r with a rate r and by associating each channel x with a corresponding rate given by $\rho(x)$. Each rate characterises an exponential distribution, such that the probability of a reaction with rate r occurring within time t is given by $F(t) = 1 - e^{-rt}$. The average duration of the reaction is given by the mean $1/r$ of this distribution.

2.2 Reduction

The reduction rules of the calculus are presented in Definition 2.2. The notation $P \xrightarrow{r,w} P'$ states that the process P can reduce to P' by performing a reaction w at rate r. The reaction identifier w can be an index i denoting a particular delay τ_r^i, or a pair of indices (i, j) denoting an interaction between two actions with indices i and j, respectively. A delay $\tau_r.P$ competing with alternative actions M can evolve to a process P at rate r (1). An output $\gamma!x(\tilde{n}).P$ competing with alternative actions M can interact with a corresponding input $\gamma?x(\tilde{m}).P$ competing with M', after which processes P and P' are executed in parallel, where the parameters $\tilde{m}$ are replaced with the values $\tilde{n}$ in process P, written $P_{\{\tilde{n}/\tilde{m}\}}$. The communication can be local (2), from a child to a parent (3) or from a parent to a child (4). Similarly, a move $\mu!x.P$ competing with alternative actions M can interact with a corresponding accept $\mu?x.P'$ competing with M', after which processes P and P' are executed in parallel. The move can be one sibling entering another (6), a child leaving its parent (7) or two siblings merging (8). In addition, all of these reductions can take place inside an ambient (9), inside a parallel composition (10), inside a restriction (11) and up to re-ordering of terms (12). This re-ordering is defined by structural congruence rules in the standard way, as presented in Definition 2.3.

(13) $P \mid \mathbf{0} \equiv P$

(14) $P \mid Q \equiv Q \mid P$

(15) $P \mid (Q \mid R) \equiv (P \mid Q) \mid R$

(16) $X(\tilde{n}) \equiv P_{\{\tilde{n}/\tilde{m}\}}$ **if** $X(\tilde{m}) = P$

(17) $\boxed{\mathbf{0}} \equiv \mathbf{0}$

(18) $\nu x\, \mathbf{0} \equiv \mathbf{0}$

(19) $\nu x\, \nu y\, P \equiv \nu y\, \nu x\, P$

(20) $\nu x\, (P \mid Q) \equiv P \mid \nu x\, Q$ **if** $x \notin \mathrm{fn}(P)$

(21) $\nu x \boxed{P} \equiv \boxed{\nu x\, P}$

Definition 2.3 *Structural congruence axioms in* SBA. Structural congruence is defined as the least congruence that satisfies these axioms. Processes in SBA are also equal up to renaming of bound names and reordering of terms in a choice.

(22) $$\nu x_1 \ldots \nu x_K\, (M_1 \mid \ldots \mid M_J \mid \boxed{P_1} \mid \ldots \mid \boxed{P_N})$$

Definition 2.4 *A process P is in* standard form *if it is in the form given by (22), where processes $P_1, \ldots P_N$ are also in standard form. A process P is in* standard index form *if it is in standard form and if each unguarded action π^i is associated with a unique index i. We write $P \cong P'$ if the indices of P can be renamed such that $P \equiv P'$.*

(23) $$\rho(P) \triangleq \sum_{P \xrightarrow{r,w}} r$$

(24) $$Pr(P \xrightarrow{r,w} P') \triangleq \frac{r}{\rho(P)}$$

(25) $$Pr(P \longrightarrow P') \triangleq \sum_{P \xrightarrow{r,w} \cong P'} \frac{r}{\rho(P)}$$

Definition 2.5 *Reaction Probabilities for a given process P in standard index form. (23) defines the total rate $\rho(P)$ of P, where $P \xrightarrow{r,w}$ means that P can perform a reaction w with rate r. (24) defines the probability $Pr(P \xrightarrow{r,w} P')$ that the process can perform a particular reaction w with rate r and evolve to P'. (25) defines the probability $Pr(P \longrightarrow P')$ that a process P can reduce to P', up to renaming of indices.*

2.3 Reaction Probability

In order to compute the probability of a given reaction, we need a way of identifying the individual reactions that a process can perform. We do this using a notion of standard index form, given by Definition 2.4.

Proposition 2.6 *Every process P in* SBA *is structurally congruent to a process in standard form.*

Proof. The proof is by straightforward application of structural congruence rules (16), (20) and (21). □

A given process can be converted to standard index form by first converting it to standard form and then renaming the indices of the unguarded actions so they are distinct. Note that the sole purpose of the indices is to identify each individual reaction, and renaming these indices has no effect on the calculus semantics, apart from the structural congruence rule (16). Consider the following system:

$$\{X(\tilde{n}) = \pi_1^1.P_1 + \pi_2^2.P_2 + \pi_3^3.P_3\} \vdash X(\tilde{n}) \mid X(\tilde{n})$$

If we convert process $X(\tilde{n}) \mid X(\tilde{n})$ to standard form we obtain the following:

$$\pi_1^1.P_1 + \pi_2^2.P_2 + \pi_3^3.P_3 \mid \pi_1^1.P_1 + \pi_2^2.P_2 + \pi_3^3.P_3$$

If we then convert this to standard index form we obtain a process in which each action π^i has a unique index i:

$$\pi_1^1.P_1 + \pi_2^2.P_2 + \pi_3^3.P_3 \mid \pi_1^4.P_1 + \pi_2^5.P_2 + \pi_3^6.P_3$$

Now, we can no longer apply rule (16) to contract this process to its original form $X(\tilde{n}) \mid X(\tilde{n})$, since the indices in the second process are no longer the same as the indices in the definition of $X(\tilde{n})$, and processes are not structurally congruent up to renaming of indices. This is important, since if we allowed the contraction we would loose the property that each action has a distinct index, and we would no longer be able to identify each individual reaction. Although we loose the ability to contract certain process definitions, this does not limit the set of reductions that the process P can perform. The result is similar to replacing rule (16) with the standard reduction rule $X(\tilde{n}) \longrightarrow P_{\{\tilde{n}/\tilde{m}\}}$ where $X(\tilde{m}) = P$. Since processes are not structurally congruent up to renaming of indices, we define an additional equivalence relation $P \cong P'$, which holds if the indices of P can be renamed such that $P \equiv P'$.

We can use the standard index form of a process to compute the probability of individual reactions, as shown in Definition 2.5. Essentially, the probability of each reaction is proportional to its rate. For a given process P in standard index form, the total rate $\rho(P)$ of the process is defined as the sum of the rates r for each reaction w of P (23). The probability $Pr(P \xrightarrow{r,w} P')$ that the process can perform a particular reaction w with rate r and evolve to P' is given by the rate of the reaction divided by the total rate of the process (24). Similarly, the probability $Pr(P \longrightarrow P')$ that the process can reduce to P' is given by the sum of the probabilities of all the reactions w for which P can reduce to P', up to renaming of indices (25). We allow reduction up to renaming of indices in order to group together processes that would otherwise be structurally congruent if their corresponding indices were the same. We can use these definitions to derive a Continuous Time Markov Chain for process P in a straightforward manner, by placing the process in standard index form after each unfolding of the chain, and computing the rate of transitioning to subsequent processes, up to structural congruence. We omit the details here, but they are analogous to [6].

2.4 Example

The following example uses the stochastic bioambient calculus to model a reversible reaction between an enzyme and a substrate, assuming that the substrate and product can also degrade. The system consists of an enzyme, a substrate and a product. The definitions in the environment are given below, together with a possible evolution of the system:

$$E = \texttt{in}?s^1.ES + \texttt{in}?p^2.ES$$
$$ES = \texttt{out}?d^3.E + \texttt{out}?k^4.E$$
$$S = \texttt{in}!s^5.X + \tau_r^6$$
$$X = \texttt{out}!d^7.S + \texttt{out}!k^8.P$$
$$P = \texttt{in}!p^9.X + \tau_r^{10}$$

$$\boxed{E} \mid \boxed{S} \mid \boxed{P}$$
$$\xrightarrow{\rho(s),(5,1)} \quad \boxed{ES \mid \boxed{X}} \mid \boxed{P}$$
$$\xrightarrow{\rho(k),(7,3)} \quad \boxed{E} \mid \boxed{P} \mid \boxed{P}$$

3 The Stochastic Bioambient Machine

A process of the bioambient calculus is simulated by encoding it to a corresponding term of the bioambient machine. The Gillespie algorithm is performed on a machine term by first choosing an ambient with probability proportional to the rate of the ambient, and then choosing a local reaction inside this ambient with probability proportional to the rate of the reaction.

3.1 Definition

A term A of the bioambient machine consists of a store S of reactions, a heap H of species and a tree T of ambients. The store S consists of a set of mappings from reactions θ to records R_θ, where each record contains information about the activity and rate of the reaction. The heap H consists of a set of mappings from species I to triples (i, U, C), where each triple contains information about the population and the behaviour of the species. The tree T consists of a list of child ambients, where each ambient contains a term, recursively. Thus, a term A is of the following form:

$$A ::= \{\theta \mapsto R_\theta, \ldots, \theta \mapsto R_\theta\}, \{I \mapsto (i, U, C), \ldots, I \mapsto (i, U, C)\}, \boxed{A} :: \ldots :: \boxed{A} :: []$$

The full syntax of the bioambient machine is given in Definition 3.1. The store S records the rates of all the *local reactions* in the term. By definition, a local reaction corresponds to one of the rules (1)-(8) of the bioambient calculus. A reaction θ can be a delay (τ, r) of rate r or an interaction (ι, x) of type ι on channel x. The interaction type is needed to distinguish between different types of interactions on the same channel, such as `in` or `out`. A record R_θ contains the activity and apparent rate of the reaction θ. The activity of a delay of rate r is given by Tot_{τ_r}, which records the total number of local delays τ_r in the term. The apparent rate $a_{(\tau,r)}$ of the delay is defined as $r \times \mathrm{Tot}_{\tau_r}$. The activity of an interaction (ι, x) is given by the triple $\mathrm{Tot}_{\iota?x}, \mathrm{Tot}_{\iota!x}, \mathrm{Mix}_{(\iota,x)}$, which records the total number of local $\iota?x$ and $\iota!x$ actions in the term, together with the number of pairs of $\iota?x$ and $\iota!x$ actions

$A ::=$	S, H, T	Term	$P, Q ::=$	$\mathbf{0}$	Null
$S ::=$	$\{\theta \mapsto R_\theta, \ldots, \theta \mapsto R_\theta\}$	Store	$\mid$	$X(\tilde{n})$	Instance
$H ::=$	$\{I \mapsto (i, U, C), \ldots, I \mapsto (i, U, C)\}$	Heap	$\mid$	$P \mid Q$	Parallel
$U ::=$	$\{\theta \mapsto K_\theta, \ldots, \theta \mapsto K_\theta\}$	Substore	$\mid$	$\nu x\, P$	Restriction
$T ::=$	$\boxed{A} :: \ldots :: \boxed{A} :: []$	Tree	$\mid$	$\boxed{P}$	Ambient
$\theta ::=$	(τ, r)	Delay			
$\mid$	(ι, x)	Interaction	$M ::=$	$\mathbf{0}$	Null
$R_{(\iota,x)} ::=$	$\mathrm{Tot}_{\iota?x}, \mathrm{Tot}_{\iota!x}, \mathrm{Mix}_{(\iota,x)}, a_{(\iota,x)}$	Activity	$\mid$	$\pi.P + M$	Action
$R_{(\tau,r)} ::=$	$\mathrm{Tot}_{\tau_r}, a_r$	Activity	$C ::=$	$\nu\tilde{n}\, M$	Choice
$K_{(\iota,x)} ::=$	$\mathrm{Tot}_{\iota?x}, \mathrm{Tot}_{\iota!x}$	Totals	$E ::=$	$\emptyset$	Empty
$K_{(\tau,r)} ::=$	Tot_{τ_r}	Totals		$\mid E, X(\tilde{m}) = P$	Process
				$\mid E, X(\tilde{m}) = C$	Choice

Definition 3.1 *Bioambient Machine Syntax.* An interaction type $\iota ::= \gamma \mid \mu$. The notation $S(\theta)$ returns the corresponding values associated with θ in S, as usual. By definition, $a_{(\iota,x)} = \rho(x) \times (\mathrm{Tot}_{\iota?x} \times \mathrm{Tot}_{\iota!x} - \mathrm{Mix}_{(\iota,x)})$ and $a_{(\tau,r)} = r \times \mathrm{Tot}_{\tau_r}$. Assume a global constant environment E, together with a global set of names Z.

that cannot interact, respectively. The apparent rate $a_{(\iota,x)}$ of an interaction (ι, x) is defined as $\rho(x) \times (\mathrm{Tot}_{\iota?x} \times \mathrm{Tot}_{\iota!x} - \mathrm{Mix}_{(\iota,x)})$. For example, $\mathrm{Tot}_{\mathtt{in}!x}$ records the number of $\mathtt{in}!x$ actions inside a child ambient of the term, $\mathrm{Tot}_{\mathtt{in}?x}$ records the number of $\mathtt{in}?x$ actions inside a child ambient of the term, and $\mathrm{Mix}_{(\mathtt{in},x)}$ records the number of pairs of $\mathtt{in}!x$ and $\mathtt{in}?x$ actions inside the same child ambient, since an ambient cannot enter itself.

The heap H consists of a set of mappings from species I to triples (i, U, C). The counter i denotes the population of the species, the choice C records the set of actions that the species can perform, and the substore U counts the number of each type of action in the choice.

The tree T consists of a list of all the child ambients in the term. Each term is also associated with a global set of names Z and a global constant environment E, which contains a set of process definitions. The syntax of processes P is constrained so that each choice C is associated with a corresponding species definition in the environment. This is necessary in order to group populations of identical species.

Definition 3.2 creates a substore U from a choice C by counting the unguarded actions inside the choice. The definition relies on a matching relation $\pi \simeq \pi'$, which checks whether an action π matches an action π'. Two actions match if they are equal (31), if they are both an input $\iota?x$ on the same channel x (32), or if they are both an output $\iota!x$ on the same channel x (33). The matching ignores any values that are sent or received over a channel.

Definition 3.3 creates substores to keep track of all the actions that can participate in a local reaction inside a given ambient. $\mathrm{Sub}(U)$ counts the relevant actions

(26) $\mathrm{Tot}_\pi(\mathbf{0}) \triangleq 0$

(27) $\mathrm{Tot}_\pi(\pi'.P + M) \triangleq 1 + \mathrm{Tot}_\pi(M)$ **if** $\pi \simeq \pi'$

(28) $\mathrm{Tot}_\pi(\pi'.P + M) \triangleq \mathrm{Tot}_\pi(M)$ **if** $\pi \not\simeq \pi'$

(29) $\mathrm{Tot}_\pi(\nu\tilde{z}\, M) \triangleq \mathrm{Tot}_\pi(M)$ **if** $\tilde{z} \cap \mathrm{sn}(\pi) = \emptyset$

(30) $\mathrm{Tot}_\pi(\nu\tilde{z}\, M) \triangleq 0$ **if** $\tilde{z} \cap \mathrm{sn}(\pi) \neq \emptyset$

(31) $\pi \simeq \pi$

(32) $\iota?x(\tilde{m}) \simeq \iota?x$

(33) $\iota!x(\tilde{n}) \simeq \iota!x$

(34) $\mathrm{Sub}(C) \triangleq \{(\tau, x) \mapsto d \quad | \quad d = \mathrm{Tot}_{\tau_x}(C) \wedge d \neq 0\}$

(35) $\cup\, \{(\iota, x) \mapsto (i, o) \quad | \quad i = \mathrm{Tot}_{\iota?x}(C) \wedge o = \mathrm{Tot}_{\iota!x}(C) \wedge (i, o) \neq (0, 0)$

Definition 3.2 *Creating a substore from a choice C, where* $\mathrm{sn}(\pi)$ *denotes the set of channels in π that are used for input or output.* The notation $\{x \mid \mathit{Condition}\}$ denotes the set of all elements x that satisfy the given *Condition*, as usual.

(36) $\mathrm{Sub}(\emptyset) \triangleq \emptyset$

(37) $\mathrm{Sub}(I \mapsto (i, U, C), H) \triangleq \mathrm{Sub}_{\mathrm{H}}(U) \oplus_i \mathrm{Sub}(H)$

(38) $\mathrm{Sub}([]) \triangleq \emptyset$

(39) $\mathrm{Sub}(\boxed{S', \emptyset, T'} :: T) \triangleq \mathrm{Sub}(T)$

(40) $\emptyset \oplus U \triangleq U$

(41) $\{U', \theta \mapsto K_\theta\} \oplus U \triangleq U' \oplus U\{\theta \mapsto (K_\theta \oplus U(\theta)\}$

(42) $(i', o') \oplus (i, o) \triangleq (i + i', o + o')$

(43) $d' \oplus d \triangleq d + d'$

(44) $\mathrm{Sub}(\boxed{S', (I \mapsto (i, U, C), H'), T'} :: T) \triangleq \mathrm{Sub}_{\mathrm{T}}(U) \oplus_i \mathrm{Sub}(\boxed{S', H', T'} :: T)$

(45) $\mathrm{Sub}(U) \triangleq \{(\mathtt{local}, x) \mapsto (i, o) \mid i, o = U(\mathtt{local}, x)\}$
$\cup\, \{(\mathtt{p2c}, x) \mapsto (0, o) \mid i, o = U(\mathtt{p2c}, x)\}$
$\cup\, \{(\mathtt{c2p}, x) \mapsto (i, 0) \mid i, o = U(\mathtt{c2p}, x)\}$
$\cup\, \{(\tau, r) \mapsto (d) \mid d = U(\tau, r)\}$

(46) $\mathrm{Sub}_{\mathrm{H}}(U) \triangleq \{(\iota, x) \mapsto (i, o) \mid i, o = U(\iota, x) \wedge \iota \in \{\mathtt{in}, \mathtt{merge}, \mathtt{s2s}\}\}$
$\cup\, \{(\mathtt{c2p}, x) \mapsto (0, o) \mid i, o = U(\mathtt{c2p}, x)\}$
$\cup\, \{(\iota, x) \mapsto (i, 0) \mid i, o = U(\iota, x) \wedge \iota \in \{\mathtt{p2c}, \mathtt{out}\}\}$

(47) $\mathrm{Sub}_{\mathrm{T}}(U) \triangleq \{(\mathtt{out}, x) \mapsto (0, o) \mid i, o = U(\mathtt{out}, x)\}$

Definition 3.3 *Creating a substore from a heap H and a tree T inside a child ambient.* The function $U \oplus_i U'$ adds i copies of the substore U to U'.

inside the heap that are contained in a substore U. $\mathrm{Sub}(H)$ counts the relevant actions inside the heap H of a child ambient, and $\mathrm{Sub}(T)$ counts the relevant actions inside the tree T of a child ambient. The function $U \oplus U'$ adds the substores U and U' by adding the counters for the corresponding elements in both substores.

Definition 3.5 adds a species to a term. The function $(I, C) \oplus (S, H, T)$ adds a species I with body C to a term (S, H, T). If a binding (i, U, C) for I is already present in the heap then the population i of the species is incremented (53). Otherwise, a new binding $(1, U, C)$ for I is created, where the substore U counts the

$$\emptyset \oplus S \triangleq S \tag{48}$$

$$\{U, \theta \mapsto K_\theta\} \oplus S \triangleq U \oplus S\{\theta \mapsto ((\theta, K_\theta) \oplus S(\theta)\} \tag{49}$$

$$((\iota, x), (i', o')) \oplus (i, o, m, a) \triangleq (i + i', o + o', m + i' \cdot o', \rho(x) \cdot ((i + i') \cdot (o + o') - (m + i' \cdot o')) \tag{50}$$

$$((\mathtt{out}, x), (i', o')) \oplus (i, o, m, a) \triangleq (i + i', o + o', m + i \cdot o' + i' \cdot o, a + \rho(x) \cdot i' \cdot o') \tag{51}$$

$$((\tau, r), d') \oplus (d, a) \triangleq (d + d', r \cdot (d + d')) \tag{52}$$

Definition 3.4 *Adding a substore to a store. Note that* `out` *is treated as a special case for rule (61), since a child cannot leave a sibling of its parent.*

$$(I, C) \oplus (S, H, T) \triangleq (\mathrm{Sub}(U) \oplus S), H\{I \mapsto (i + 1, U, C)\}, T \textbf{ if } H(I) = (i, U, C) \tag{53}$$

$$(I, C) \oplus (S, H, T) \triangleq (\mathrm{Sub}(U) \oplus S), H\{I \mapsto (1, U, C)\}, T \textbf{ if } H(I) = \emptyset, U = \mathrm{Sub}(C) \tag{54}$$

Definition 3.5 *Adding a species to a term*

$$\mathbf{0} \oplus A \triangleq A \tag{55}$$

$$X(\tilde{n}) \oplus A \triangleq P_{\{\tilde{n}/\tilde{m}\}} \oplus A \textbf{ if } X(\tilde{m}) = P \tag{56}$$

$$X(\tilde{n}) \oplus A \triangleq (X(\tilde{n}), C_{\{\tilde{n}/\tilde{m}\}}) \oplus A \textbf{ if } X(\tilde{m}) = C \tag{57}$$

$$(P \mid Q) \oplus A \triangleq P \oplus Q \oplus A \tag{58}$$

$$(\nu x\, P) \oplus A \triangleq P_{\{y/x\}} \oplus A \textbf{ if } y \text{ fresh } \textbf{and } Z \leftarrow Z \cup \{y\} \tag{59}$$

$$\boxed{P} \oplus A \triangleq \boxed{P \oplus (\emptyset, \emptyset, [])} \oplus A \tag{60}$$

$$\boxed{S', H', T'} \oplus (S, H, T) \triangleq (\mathrm{Sub}(H') \oplus \mathrm{Sub}(T')) \oplus S, H, \boxed{S', H', T'} :: T \tag{61}$$

Definition 3.6 *Adding a process and an ambient to a term. Assume a global constant environment E, and a global set of names Z that can be updated.*

$$(S, H, T) \oplus (S', H', T') \triangleq (S \oplus S'), (H \oplus H'), (T @ T') \tag{62}$$

Definition 3.7 *Merging two terms.* The function $T@T'$ concatenates the lists T and T' in the standard manner. The definitions $S \oplus S'$ and $H \oplus H'$ are omitted.

$$\rho(S) \triangleq \sum_{i=1}^{N} a_i \textbf{ if } k_i, a_i = S(\theta_i) \tag{63}$$

$$\rho(S, H, T) \triangleq \rho(S) + \rho(T) \tag{64}$$

$$\rho([]) \triangleq 0 \tag{65}$$

$$\rho(\boxed{A} :: T) \triangleq \rho(A) + \rho(T) \tag{66}$$

Definition 3.8 *Computing the rate of a term.*

actions in the species, given by $\mathrm{Sub}(C)$, and the population of the species is set to 1 (54). Note that whenever a new species is added to a term, the local actions inside the substore U, given by $\mathrm{Sub}(U)$, are added to the store S.

Definition 3.6 adds a process or an ambient to a machine term. The function $P \oplus A$ adds process P to term A. The null process $\mathbf{0}$ is discarded (55). If a species

$X(\tilde{n})$ is defined as a process P then the process is added to the term, where the parameters $\tilde{m}$ are instantiated with the values $\tilde{n}$ (56). If a species $X(\tilde{n})$ is defined as a choice C then the species is added to the term, where the parameters $\tilde{m}$ are instantiated with the values $\tilde{n}$ in the choice (57). A parallel composition $P \mid Q$ is split so that each process is added separately (58). A restriction $\nu x\, P$ is added to a term by replacing x with a fresh channel y and adding this to the global set of channels Z (59). An ambient $\boxed{P}$ is added to a term by creating a new machine ambient with an empty term, and adding the process P to this term. The function $\boxed{B} \oplus A$ adds a machine ambient $\boxed{B}$ to a term A. When a child ambient $\boxed{S', H', T'}$ is added to a term (S, H, T), the function $\mathrm{Sub}(H')$ counts the actions in the heap of the child ambient that can be involved in a local reaction, and the function $\mathrm{Sub}(T')$ counts the actions in the tree of the child ambient that can be involved in a local reaction. These actions are added to the store S (61). For improved efficiency, $\mathrm{Sub}(H)$ and $\mathrm{Sub}(T)$ can be cached locally inside each ambient.

Definition 3.4 add a substore to a store. The function $U \oplus S$ adds a substore U to a store S by adding the corresponding totals for each reaction θ in U, and re-computing the apparent rate a_θ accordingly. Note that the number of mixed reactions for a given reaction in the substore is computed by multiplying the number of inputs and outputs. Conversely, the function $S \ominus U$ subtracts the substore U from the store S by subtracting the corresponding totals for each reaction in U, and re-computing the apparent rate accordingly (definition not shown).

Definition 3.7 merges two terms. The function $A \oplus B$ merges terms A and B by merging their stores, concatenating their trees and merging their heaps. The function $H \oplus H'$ merges two heaps by taking the disjoint union of the heaps and summing the populations for entries I that occur in both heaps. The function $S \oplus S'$ merges two stores by adding the corresponding totals for each reaction θ in S and S' and re-computing the apparent rate a_θ accordingly (definitions not shown).

Definition 3.8 computes the rate of a term A. The function $\rho(A)$ computes the sum of the rates of all the reactions in term A. The function $\rho(S)$ computes the sum of the rates of all the reactions in S. For a given term (S, H, T) the function $\rho(S)$ corresponds to the sum of the rates of all the local reactions in the term. This allows the reactions of a term to be divided up between the different ambients in a hierarchical fashion.

Definition 3.9 executes a single reaction in a given machine term. The relation $A \xrightarrow{r,w} A'$ executes a reaction inside A with rate r and identifier w according to the Gillespie algorithm, producing a modified term A'. Rule (77) computes the total rate a_0 of the term A and then chooses a reaction index a between 0 and a_0 in accordance with the Gillespie algorithm [5], where the function $\mathrm{Rand}(n)$ returns a number in the interval $[0, n[$ following a uniform probability distribution. The global time of the simulation is updated as a side effect in accordance with [5].

The relation $A \xrightarrow{r,w,a} A'$ executes the a^{th} reaction inside A with rate r and identifier w, producing a modified term A'. Rule (76) uses the reaction index a to

(67) $$(I, \nu\tilde{z}\,(\tau_r.P + M)) \oplus B \quad \xrightarrow{(\tau,r)} \quad P \oplus B$$

(68) $$\begin{array}{l}(I, \nu\tilde{z}\,(!x(\tilde{n}).P + M))\oplus \\ (I', \nu\tilde{z}'\,(?x(\tilde{m}).P' + M')) \oplus B\end{array} \quad \xrightarrow{(\texttt{local},x)} \quad P \oplus P'_{\{\tilde{n}/\tilde{m}\}} \oplus B$$

(69) $$\begin{array}{l}\boxed{(I, \nu\tilde{z}\,(\texttt{in}!x.P + M)) \oplus A} \oplus \\ \boxed{(I', \nu\tilde{z}'\,(\texttt{in}?x.P' + M')) \oplus A'} \oplus B\end{array} \quad \xrightarrow{(\texttt{in},x)} \quad \boxed{P' \oplus (\boxed{P \oplus A} \oplus A')} \oplus B$$

(70) $$\boxed{\begin{array}{l}(I', \nu\tilde{z}'\,(\texttt{out}?x.P' + M'))\oplus \\ \boxed{(I, \nu\tilde{z}\,(\texttt{out}!x.P + M)) \oplus A} \oplus A'\end{array}} \oplus B \quad \xrightarrow{(\texttt{out},x)} \quad \boxed{P \oplus A} \oplus \boxed{P' \oplus A'} \oplus B$$

(71) $$\begin{array}{l}\boxed{(I, \nu\tilde{z}\,(\texttt{merge}!x.P + M)) \oplus A} \oplus \\ \boxed{(I', \nu\tilde{z}'\,(\texttt{merge}?x.P' + M')) \oplus A'} \oplus B\end{array} \quad \xrightarrow{(\texttt{merge},x)} \quad \boxed{P \oplus P' \oplus (A \oplus A')} \oplus B$$

(72) $$\begin{array}{l}(I', \nu\tilde{z}'\,(\texttt{c2p}?x(m).P' + M'))\oplus \\ \boxed{(I, \nu\tilde{z}\,(\texttt{c2p}!x(n).P + M)) \oplus A} \oplus B\end{array} \quad \xrightarrow{(\texttt{c2p},x)} \quad \boxed{P \oplus A} \oplus P'_{\{\tilde{n}/\tilde{m}\}} \oplus B$$

(73) $$\begin{array}{l}(I, \nu\tilde{z}\,(\texttt{p2c}!x(n).P + M))\oplus \\ \boxed{(I', \nu\tilde{z}'\,(\texttt{p2c}?x(m).P' + M')) \oplus A'} \oplus B\end{array} \quad \xrightarrow{(\texttt{p2c},x)} \quad P \oplus \boxed{P'_{\{\tilde{n}/\tilde{m}\}} \oplus A'} \oplus B$$

(74) $$\begin{array}{l}\boxed{(I, \nu\tilde{z}\,(\texttt{s2s}!x(n).P + M)) \oplus A} \oplus \\ \boxed{(I', \nu\tilde{z}'\,(\texttt{s2s}?x(m).P' + M')) \oplus A'} \oplus B\end{array} \quad \xrightarrow{(\texttt{s2s},x)} \quad \boxed{P \oplus A} \oplus \boxed{P'_{\{\tilde{n}/\tilde{m}\}} \oplus A'} \oplus B$$

(75) $$\frac{\begin{array}{c}\text{Choose}(j, \theta_\mu, (S, H, T)) \xrightarrow{\theta_\mu} A' \\ j = \lfloor \frac{a-a'}{r} \rfloor \quad a' < a \leq \sum_{i=1}^{\mu} a_i \quad k_i, a_i = S(\theta_i) \quad a' = \sum_{i=1}^{\mu-1} a_i \quad r = \rho(\theta_\mu)\end{array}}{S, H, T \xrightarrow{r, a'+j\cdot r, a} A'}$$

(76) $$\frac{S', H', T' \xrightarrow{r,w,a} A' \quad a' = \rho(S) + \rho(T)}{S, H, T@\boxed{S', H', T'} :: T'' \xrightarrow{r, a'+w, a'+a} \boxed{A'} \oplus ((S \ominus \text{Sub}(H') \ominus \text{Sub}(T')), H, T@T'')}$$

(77) $$\frac{A \xrightarrow{r,w,a} A' \quad a = \text{Rand}(a_0) \quad t = (1/a_0) \cdot \ln(\frac{1}{\text{Rand}(1)}) \quad a_0 = \rho(A)}{A \xrightarrow{r,w} A'} time = time + t$$

Definition 3.9 *Bioambient Machine Reduction. Assume a global constant environment E, together with a global set of names Z, which contains all of the restricted names in the system. Also let $\rho(\tau, r) = r$ and $\rho(\iota, x) = \rho(x)$. For each rule it is assumed that $(\tilde{z} \cup \tilde{z}') \cap Z = \emptyset$ on the left hand side, and $Z \leftarrow Z \cup (\tilde{z} \cup \tilde{z}')$ on the right hand side.*

choose an appropriate ambient inside the term. Since each ambient in the term effectively owns a subset of the reactions, the choice of an ambient can be made by

subtracting the total rate of each ambient from the index a until the index is less than zero, at which point we know that the index denotes a reaction somewhere inside the current ambient. Rule (75) then uses the remaining index a to choose a reaction type inside the chosen term, by summing the rates of the reaction types until the index is exceeded. Once a reaction type θ_μ is chosen, we use the remaining index j to choose the j^{th} reaction of that type, where the function Choose(j, θ, A) re-arranges the term A in order to choose the j^{th} species or pair of species that can perform a reaction of type θ (definition not shown). We compute the index j so that it corresponds to an integer number of reactions of rate $\rho(\theta_\mu)$. This allows us to assign a unique number to each individual reaction in the term, given by the sum of the rates of all the reactions that precede it.

Thus, a single index a chosen at the top level is used to chose an ambient, then a reaction type, then a specific reaction. The end result is that the probability of an individual reaction being chosen is proportional to the reaction rate, in accordance with the Gillespie algorithm. Note that the initial index a is used to pinpoint a single reaction inside the term, while the number w corresponds to a unique reaction identifier, given by the sum of the rates of all the reactions that precede the chosen reaction. Once we have identified each individual reaction in this way using a suitable index w, we define the reaction rate and reaction probabilities as in Definition 2.5, without the need for a standard index form.

3.2 Example

The example from the Sec. 2 is encoded to the machine term in Fig. 1, assuming that the definitions in the environment remain unchanged. Each term keeps track of the activity and apparent rate of the local reactions that it can perform. Initially, the top-level term can perform an enter reaction on channel s, written $(\texttt{in}, s)$. The activity of the reaction is stored as the tuple $(1, 1, 0, \rho(s))$, which records the number of accept actions $\texttt{in}?s$ inside a child ambient, the number of enter actions $\texttt{in}!s$ inside a child ambient, the number of pairs of enter and accept actions on s that occur inside the same child ambient, and the apparent rate of the reaction, respectively. The top-level term can also perform an enter reaction on p, written $(\texttt{in}, p)$, whose activity is recorded in a similar way. In addition, the substrate ambient can perform a local delay reaction of rate r, written (τ, r). The activity of the reaction is stored as the tuple $(1, r)$, which records the number of delay actions τ_r and the apparent rate of the reaction, respectively. The product ambient can also perform a local delay reaction, whose activity is recorded in a similar way, while the enzyme ambient cannot perform any local reactions. The machine picks one of the four terms in the system with probability proportional to the rate of the term, where the rate of a term is defined as the sum of the rates of all the local reactions in the term. Initially the enzyme term has a rate of 0, since it cannot perform any local reactions, while the top-level term has a rate of $\rho(s) + \rho(p)$, and the substrate and product terms both have a rate of r. For the first reaction, the top-level term is chosen to execute the reaction $(\texttt{in}, s)$. Subsequently, the top-level term is chosen to execute the reaction $(\texttt{out}, k)$. This example was deliberately chosen for its simplicity, in

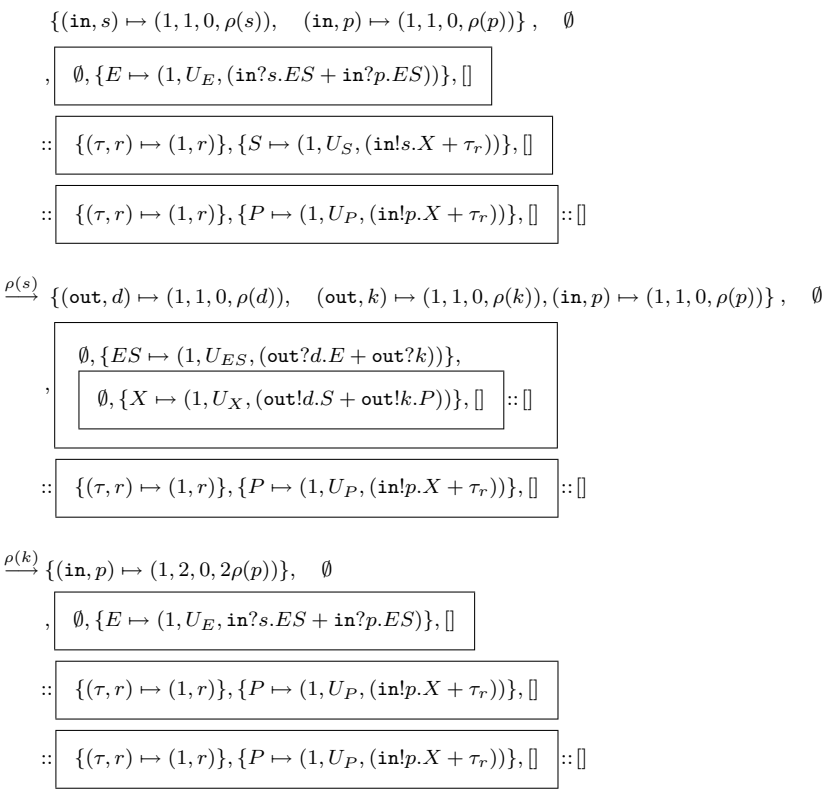

Fig. 1. Enzymatic Example in the Bioambient Machine

order to illustrate the primitives of the abstract machine. We are currently using the bioambient machine to simulate a more complex immune system pathway, in which immune cells move in and out of the thymus, where they interact with T-Cells to trigger an immune response.

3.3 Correctness

We briefly outline a proof of correctness for the stochastic bioambient machine (SBAM). The function $(|E \vdash P|)$ encodes a system $E \vdash P$ in SBA to a corresponding system in SBAM, as described in Definition 3.10. The encoding assumes that each choice of actions in the system $E \vdash P$ is defined as a separate species, which is straightforward to enforce along the lines of [10]. A corresponding decoding from SBAM to SBA is described in Definition 3.11. Theorem 3.12 and Theorem 3.13 ensure that the bioambient calculus and the bioambient machine are reduction equivalent. In order to preserve the correspondence, we define a notion of structural congruence for machine terms, where terms are structurally congruent up to re-ordering of ambients and actions.

Theorem 3.12 $\forall E, A \in \text{SBAM}.\ E \vdash A \xrightarrow{r,w'} E \vdash A' \Rightarrow [|E \vdash A|] \xrightarrow{r,w} [|E \vdash A'|]$

Theorem 3.13 $\forall E, P \in \text{SBA}.\ E \vdash P \xrightarrow{r,w} E \vdash P' \Rightarrow (|E \vdash P|) \xrightarrow{r,w'} \equiv (|E \vdash P'|)$

(78) $(|E \vdash P|) \triangleq P \oplus (\emptyset, \emptyset, [])$

Definition 3.10 *Encoding a system from* SBA *to* SBAM.

(79) $$[|E \vdash A|] \triangleq E \vdash [|A|]$$

(80) $$[|S, H, T|] \triangleq [|H|] \mid [|T|]$$

(81) $$[|\emptyset|] \triangleq \mathbf{0}$$

(82) $$[|H, X(\tilde{n}) \mapsto (i, U, C)|] \triangleq \underbrace{X(\tilde{n}) \mid \ldots \mid X(\tilde{n})}_{i} \mid [|H|]$$

(83) $$[|[]|] \triangleq \mathbf{0}$$

(84) $$[|\boxed{A} \mid T|] \triangleq \boxed{[|A|]} \mid [|T|]$$

Definition 3.11 *Decoding a system from* SBAM *to* SBA. The environment E is unchanged (79), and for each mapping $X(\tilde{n}) \mapsto (i, U, C)$ in the heap, i copies of the instance are executed in parallel (82). A tree of ambients T is decoded by applying the decoding function to the terms inside the ambient, recursively (84).

Theorem 3.14 and Theorem 3.15 ensure that the reaction probabilities of the bioambient machine correspond to those of the bioambient calculus, and vice-versa. The proofs rely on the fact that each reaction in SBA corresponds to a single reaction in SBAM, and vice-versa.

Theorem 3.14 $\forall P \in \text{SBA}. Pr(P \longrightarrow P') = Pr((|P|) \longrightarrow\equiv (|P'|))$

Theorem 3.15 $\forall A \in \text{SBAM}. Pr(A \longrightarrow\equiv A') = Pr([|A|] \longrightarrow [|A'|])$

4 Discussion

The abstract machine is an extension of the abstract machine for the stochastic pi-calculus presented in [10], where species populations are grouped together for improved efficiency. The definition of the heap H and store S are similar, but the store is extended with many more interaction types, which represent the different interactions between compartments. In order to handle nested compartments, an additional tree T is also defined, where each node in the tree can be viewed as a separate abstract machine. Additional functions are defined for counting the different interactions between compartments in the tree. There are also significantly more reduction rules to implement, which require functions for removing and updating compartments, and for choosing a compartment with the correct probability. When an ambient moves, the reactions of its parent and grandparent ambients need to be updated accordingly, requiring careful tracking of the various dependencies between ambients. The resulting implementation will be made available at [8] as a public resource, and a prototype stochastic simulator has already been released [2] based

[2] http://aesop.doc.ic.ac.uk/tools/bam

on the abstract machine presented in this paper. A description of the prototype is presented in [7] together with a biological example, but the definition of the abstract machine has not previously been published.

Section 2 presents a concise syntax and reduction semantics for the stochastic bioambient calculus, in which a unique index is associated with each action in order to identify and count the individual reactions of a process. In [14] a stochastic semantics for the ambient calculus is defined using a labelled transition system, while in [2] a stochastic semantics for the bioambient calculus is defined in terms of a reduction relation, in which the identification of a reaction is determined by placing a number of constraints on the syntax of the choice operator and by keeping track of the total counts of each type of reaction, along the lines of [10].

To the best of our knowledge, this paper presents the first formally defined abstract machine for the stochastic bioambient calculus. In [13], a variant of the stochastic pi-calculus with polyadic synchronisation is used as a basis for encoding this calculus. The encoding requires a number of locks and broadcast mechanisms in order to propagate changes in the topology of the system, which leads to a significant computational overhead. In contrast, this paper presents an abstract machine for the direct implementation of the bioambient calculus, based on principles of efficient simulation previously defined in [10]. The abstract machine has also been adapted for implementing the stochastic brane calculus [3] and the resulting platform provides a flexible means of experimenting with different synchronisation primitives for calculi with mobile compartments.

References

[1] Ralf Blossey, Luca Cardelli, and Andrew Phillips. A compositional approach to the stochastic dynamics of gene networks. *Transactions in Computational Systems Biology*, 3939:99–122, January 2006.

[2] Linda Brodo, Pierpaolo Degano, and Corrado Priami. A stochastic semantics for bioambients. In *PaCT*, pages 22–34, 2007.

[3] Luca Cardelli. Brane calculi. In *CMSB'04*, pages 257–278, 2004.

[4] Luca Cardelli and Gianluigi Zavattaro. On the computational power of biochemistry. In *Algebraic Biology*, 2008. To Appear.

[5] Daniel T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.*, 81(25):2340–2361, 1977.

[6] Céline Kuttler, Cédric Lhoussaine, and Joachim Niehren. A stochastic pi calculus for concurrent objects. In *AB*, pages 232–246, 2007.

[7] Vinod Mugathan, Andrew Phillips, and Maria Vigliotti. Bam: Bioambient machine. In *ACSD'08*, 2008. To Appear.

[8] Andrew Phillips. *The Stochastic Pi-Machine*, 2007. Available from http://research.microsoft.com/~{}aphillip/spim/.

[9] Andrew Phillips and Luca Cardelli. A correct abstract machine for the stochastic pi-calculus. In *Concurrent Models in Molecular Biology*, August 2004.

[10] Andrew Phillips and Luca Cardelli. Efficient, correct simulation of biological processes in the stochastic pi-calculus. In *Computational Methods in Systems Biology*, volume 4695 of *LNCS*, pages 184–199. Springer, September 2007.

[3] Extended version available at http://research.microsoft.com/~aphillip

[11] Corrado Priami. Stochastic π-calculus. *The Computer Journal*, 38(6):578–589, 1995.

[12] Aviv Regev, Ekaterina M. Panina, William Silverman, Luca Cardelli, and Ehud Y. Shapiro. Bioambients: an abstraction for biological compartments. *Theor. Comput. Sci.*, 325(1):141–167, 2004.

[13] Cristian Versari. A core calculus for a comparative analysis of bio-inspired calculi. In *ESOP*, pages 411–425, 2007.

[14] Maria Grazia Vigliotti and Peter G. Harrison. Stochastic ambient calculus. *Electr. Notes Theor. Comput. Sci.*, 164(3):169–186, 2006.