Available online at www.sciencedirect.com

SCIENCE DIRECT®

Theoretical Computer Science 325 (2004) 141 – 167

Theoretical Computer Science

www.elsevier.com/locate/tcs

BioAmbients: an abstraction for biological compartments

Aviv Regev[a,*], Ekaterina M. Panina[b], William Silverman[c], Luca Cardelli[d], Ehud Shapiro[c]

[a] *Bauer Center for Genomics Research, Harvard University, Cambridge, MA, USA*
[b] *Molecular Biology Institute, UCLA, Los Angeles, CA, USA*
[c] *Computer Science and Applied Math, Weizmann Institute, Rehovot, Israel*
[d] *Microsoft Research, Cambridge, UK*

Abstract

Biomolecular systems, composed of networks of proteins, underlie the major functions of living cells. Compartments are key to the organization of such systems. We have previously developed an abstraction for biomolecular systems using the π-calculus process algebra, which successfully handled their molecular and biochemical aspects, but provided only a limited solution for representing compartments. In this work, we extend this abstraction to handle compartments. We are motivated by the ambient calculus, a process algebra for the specification of process location and movement through computational domains. We present the BioAmbients calculus, which is suitable for representing various aspects of molecular localization and compartmentalization, including the movement of molecules between compartments, the dynamic rearrangement of cellular compartments, and the interaction between molecules in a compartmentalized setting. Guided by the calculus, we adapt the BioSpi simulation system, to provide an extended modular framework for molecular and cellular compartmentalization, and we use it to model and study a complex multi-cellular system.
© 2004 Elsevier B.V. All rights reserved.

Keywords: Biology; π-Calculus; Ambient calculus; BioAmbients; Modeling; Simulation

1. Introduction

Compartments play an essential role in the functioning of biomolecular systems, by organizing them in a hierarchical and modular way. In order to perform its function,

* Corresponding author.
E-mail address: aregev@cgr.harvard.edu (A. Regev).

0304-3975/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2004.03.061

a molecule must be present in the right location. Localization of molecules to specific compartments is a key regulatory mechanism in diverse biological systems [11].

Despite the critical role of compartments in biology, most existing models of biological systems have focused on chemical reactions and pay little attention to this level of organization. In our previous work [19], we developed an abstraction for biomolecular systems using the π-calculus process algebra [13], and extended the calculus to accurately handle the quantitative aspects of biochemical systems [17]. This abstraction successfully handled the molecular and biochemical aspects of biological systems. However, its treatment of compartments (as private channels) was limited.

Here, we address this problem by broadening our approach and developing an extension of the stochastic π-calculus that provides a better abstraction of compartmentalization. The extended abstraction, motivated by the Ambient calculus [4], allows us to study biological compartments of different granularities, movement of molecules between compartments, and dynamic rearrangement of cellular compartments and molecular complexes. We show how the resulting *BioAmbients Calculus* facilitates the modeling of complex molecular systems in a cellular and multi-cellular context. We implement BioAmbients as part of the BioSpi system to provide a more complete modular framework for molecular interaction, localization and compartmentalization.

1.1. Previous work

Existing work on modeling biological compartments is limited, and can be roughly divided into three categories.

- *Ontologies and data schemas* are based on the “compartment-as-object” abstraction. These are designed for genome and pathway databases and often represent cellular and sub-cellular compartments using a comprehensive hierarchy of objects. For example, the Gene Ontology (GO [1]), employed by most genome databases, has an elaborate hierarchical vocabulary which encompasses both sub-cellular compartments and molecular complexes and machines. Pathway databases such as BIND [2] and TRANSPATH [23] incorporate some compartment hierarchy and localization information. Naturally, however, this information is not dynamic and thus fails to reflect movement of and between compartments.
- *Ad hoc kinetic models with a compartment component*: Most kinetic models of specific systems either neglect this aspect entirely or form ad hoc solutions for the problem, with highly specialized models (e.g. [8]). These may provide an efficient solution to specific problems, but are not easily generalizable, and do not constitute a rigorous framework.
- *Models based on abstract process languages* use the “compartment-as-process” abstraction. Most works based on such languages, such as Petri nets (e.g. [7]) or process algebras (e.g. the core formal molecular biology language of [5] and the Bio-calculus of [15]), do not handle compartments explicitly. Two notable exceptions are Matsuno et al.’s [12] hybrid Petri net model of Notch signaling in *Drosophila* and Kam et al.’s [9] Statecharts models of the immune system and *Caenorhabditis elegans* vulva development. In the former, a stochastic Petri net abstraction is extended with an additional layer, representing cells. In the latter, compartmentalization and

localization are integral parts of a qualitative process model. In both cases, however, abstraction of movement of and between compartments is either limited or completely absent. Finally, Paun's P-systems provide a formal framework with an explicit notion of compartments, grounded in formal grammars (rather than process calculi). While P-systems were inspired by biology, they were neither conceived nor employed as a framework to study real biological systems, and have diverged from the biological example in the abstraction process [16]. Importantly, dynamic membrane P-systems are not well developed, rendering them limited in their applicability to biological systems.

In our previous work, we employ a "compartment as private channel" abstraction. Our underlying assumption there is that compartmentalization can be abstracted by communication scope. Thus, all the processes representing molecules in one compartment or molecular complex share certain exclusive communication capabilities (private channels) that are inaccessible to processes representing molecules outside this compartment. The limited scope of the private channel represents the boundary of the corresponding compartment. Both movement of molecules between compartments and formation of complexes are represented by mobility of private channels. Thus, the "compartment as private channel" approach uniformly treats both sub-cellular compartments and complexes. For example, we represent the limited interactions of a single molecule or a molecular complex using a combination of multiple public and private channels.

While essentially accurate, the private channel-based approach is often impractical. Essential events, such as the movement of a molecule in or out of a compartment or the merger of two compartments, require highly elaborate encodings involving the multi-step propagation of large sets of private channels between many processes. This is due to the fact that "compartments" are only derived from the private channel distribution between different processes all existing at the same level. To rigorously address this problem, we need to extend the mathematical domain of the stochastic π-calculus with additional entities, that will correspond better to our biological notions. In this work we present this extension, called the BioAmbients calculus.

2. Abstracting compartments as ambients: an overview

Building an abstraction consists of three steps: informal organization of the knowledge in the real-world domain, selection (or development) of an appropriate mathematical domain, and designing the abstraction between the two.

2.1. The real-world domain: essential properties of biomolecular compartments

We identify two types of compartmentalization in cellular and molecular systems. *Membrane-bound compartments* include cells, organelles, and vesicles, and have a clearly defined boundary, which insulates the compartment's components from the external environment. Membrane-bound compartments are hierarchically organized, e.g. organelles and vesicles residing within cells.

Molecular compartments include stable or transient multi-molecular complexes and are equally important [10]. The formed complex partially insulates the component molecules from the environment. In addition, single proteins composed of multiple linked parts (termed “domains”) may sometimes be considered compartments in their own right, where the backbone of the protein that links the domains together also partly insulates them from their environment. We can therefore refer to a hierarchy of molecular compartments, where molecules (one level) form complexes (a second, higher level).

Compartments introduce a notion of location. Most entities in the system (i.e. molecules or compartments) may reside either within or outside a given compartment. In addition, *cross-compartment molecules*, such as cross-membrane receptors, channels, and transporters reside across a boundary and belong to two compartments. However, the two compartments are not symmetrical. Since these molecules are membrane-linked, they primarily belong to one of the compartments, and would move together with it.

Entities may also change their location by movement between compartments. We identify two types of movement. *Movement between compartments* occurs when, e.g. molecules move across a membrane, thereby entering or existing a membrane-bound compartment (e.g. the cell, the nucleus or the Golgi apparatus). Similarly, molecules may join a molecular compartment, for example by binding to one of its members.

Compartment movement occurs when an entire compartment moves with respect to the other compartments in the system. The most typical event is the merge of two membrane-bound compartments, in which two separate compartments become one, with their contents shared. In other cases, compartments may enter or exit one another, in events such as phagocytosis (cell “enters” cell) or entry of a complex molecule (a molecular compartment) into an organelle (a membrane-bound compartment).

2.2. The mathematical domain: Bioambients

Compartmentalization and movement across boundaries play a critical role in computational systems as well [4]. In particular, the advent of the World-Wide Web has increased the potential for *mobile computation* that involves mobile devices (e.g. laptops), mobile code that moves between devices, and boundaries (e.g. firewalls).

To describe such organization, Cardelli and Gordon [4] have developed the ambient calculus as a paradigm for mobile computation. Our modified version of the ambient calculus, termed BioAmbients, facilitates the mapping of biological compartments as ambients, as will be illustrated in subsequent sections. Since BioAmbients contains the stochastic π-calculus [17], we discuss only the additional entities and operations. The full syntax, congruence laws and semantics are given in Figs. 1–3. We address the differences between the ambient calculus and BioAmbients in Section 6.

2.2.1. Ambients

An *ambient* is a bounded place where computation happens. The boundary surrounding the ambient defines what is inside and what is outside it. Ambients may have names, but these are used to improve readability only, and have no functionality. Each ambient harbors a collection of processes, that reside and run directly within it.

Mobility and communication primitives		
n, m, p		names
π	$\overset{\text{def}}{=}$	Actions
	$\$n!\{m\}$	Output action
	$\$n?\{m\}$	Input action
$\$$	$\overset{\text{def}}{=}$	Directions
	local	Intra-ambient
	s2s	Inter-siblings
	p2c	Parent to child
	c2p	Child to parent
M, N	$\overset{\text{def}}{=}$	Capabilities
	enter n	Synch entry
	accept n	Synch accept
	exit n	Synch exit
	expel n	Synch expel
	merge $+$ *n*	Synch merge with
	merge $-$ *n*	Synch merge into
P, Q	$\overset{\text{def}}{=}$	Processes
	$(\text{new } n)P$	Restriction
	$P \mid Q$	Composition
	$!P$	Replication
	$[P]$	Ambient (membrane)
	$\pi.P$	Communication prefix
	$M.P$	Capability prefix
	$\sum_{i \in I} \pi_i.P_i$	Communication Choice
	$\sum_{i \in I} M_i.P_i$	Capability Choice

Fig. 1. BioAmbients: syntax. All capabilities and communications are synchronous, ambients are nameless (but labels can be attached as comments). Communication is allowed within ambients and between sibling and parent–child ambients. We write $\pi_1.P + \pi_2.Q$ for binary communication choice, 0 for empty choice, and $\pi.P + T$ to single out one communication option, and similarly for capability choice. Communication choice and capability choice are kept separate, both to simplify the implementation, and because mixed choices do not appear to be very useful. Note that replication is taken as a primitive instead of recursion: this is commonly done in process calculi since replication is formally simpler to handle, and recursion can be easily derived from it [13]. Recursion will be used in examples.

Ambients can be nested within other ambients, with each ambient harboring a collection of sub-ambients, with their content. An ambient moves as a whole, with its component processes and sub-ambients. The processes inside an ambient control it, by instructing it to move.

An ambient is written $n[P]$, where n is the (optional) name of the ambient, and P is the process running inside the ambient. In $n[P]$, P is actively running and can be the parallel composition of several processes. The ambient tree hierarchy is represented by the nesting of ambient brackets (Fig. 4A). Each node of the tree may contain both non-ambient processes running in parallel and sub-ambients. An ambient with p content processes and q sub-ambients is $n[P_1 | \cdots | P_p | m_1[\cdots] | \cdots | m_q[\cdots]]$ (where P_i is not of the form $n[\cdots]$).

Structural congruence and process identity

$P\|Q \equiv Q\|P$	Struct Par Commut
$(P\|Q)\|R \equiv P\|(Q\|R)$	Struct Par Assoc
$P\|\mathbf{0} \equiv P$	Struct Par Zero
$[\mathbf{0}] \equiv \mathbf{0}$	Struct Amb Zero
$(\mathsf{new}\ n)\mathbf{0} \equiv \mathbf{0}$	Struct Res Zero
$(\mathsf{new}\ n)(\mathsf{new}\ m)P \equiv (\mathsf{new}\ m)(\mathsf{new}\ n)P$	Struct Res Res
$(\mathsf{new}\ n)(P\|Q) \equiv P\|(\mathsf{new}\ n)Q$ if $n \notin fn(P)$	Struct Res Par
$(\mathsf{new}\ n)[P] \equiv [(\mathsf{new}\ n)P]$	Struct Res Amb
$\$n?\{m\}.P \equiv \$n?\{p\}.P\{m \leftarrow p\}$ if $p \notin fn(P)$	Renaming bound names
$(\mathsf{new}\ n)P \equiv (\mathsf{new}\ m)P\{n \leftarrow m\}$ if $m \notin fn(P)$	Renaming bound names
$!\mathbf{0} \equiv \mathbf{0}$	Struct Repl Zero
$!P \equiv P\|!P$	Struct Repl Par

Fig. 2. BioAmbients: structural congruence. $\equiv$ is a congruence relation over the syntax, including reordering of terms in a choice and with the additional properties listed here. $fn(P)$, $fn(\pi)$, and $fn(M)$ are the free names of a process, communication and capability, respectively, i.e. the names not bound by new or input. $P\{n \leftarrow m\}$ is the substitution of m for the free occurrences of n in P.

Reduction rules

$[(T + enter\ n.P) \mid Q] \mid [(T' + accept\ n.R) \mid S] \to [\ [P \mid Q] \mid R \mid S\ ]$	Red In
$[[(T + exit\ n.P) \mid Q] \mid (T' + expel\ n.R) \mid S] \to [P \mid Q] \mid [R \mid S]$	Red Out
$[(T + merge+\ \ n.P) \mid Q] \mid [(T' + merge-\ \ n.R) \mid S] \to [P\|Q\|R\|S]$	Red Merge
$(T + local\ n!\{m\}.P) \mid (local\ n?\{p\}.Q + T') \to P\|Q\{p \leftarrow m\}$	Red Local
$(T + p2c\ n!\{m\}.P) \mid [(c2p\ n?\{p\}.Q + T') \mid R] \to P \mid [Q\{p \leftarrow m\} \mid R]$	Red Parent Output
$[R \mid (T + c2p\ n!\{m\}.P)] \mid (p2c\ n?\{p\}.Q + T') \to [R\|P] \mid Q\{p \leftarrow m\}$	Red Parent Input
$[R \mid (T + s2s\ n!\{m\}.P)] \mid [(s2s\ n?\{p\}.Q + T') \mid S] \to [R\|P] \mid [Q\{p \leftarrow m\}\|S]$	Red Sibling
$P \to Q \Rightarrow (\mathsf{new}\ n)P \to (\mathsf{new}\ n)Q$	Red Res
$P \to Q \Rightarrow [P] \to [Q]$	Red Amb
$P \to Q \Rightarrow P\|R \to Q\|R$	Red Par
$P \equiv P',\ P \to Q,\ Q \equiv Q' \Rightarrow P' \to Q'$	Red $\equiv$ (Struct)

Fig. 3. BioAmbients: operational semantics. The first three reduction rules handle ambient operations. The next four reduction rules handle communication within ambients (similar to the π-calculus) and between neighboring ambients. The remaining rules handle reductions in context and up to structural congruence. Note, that we write $\pi.P + T$ to single out one communication option, and similarly for capability choice. Stochastic semantics follows the same approach as in the biochemical stochastic π-calculus ([17], not shown).

Capabilities can change the ambient hierarchy by allowing ambient entry, exit, or merge. All capabilities are synchronized in pairs, using named channels. There are three pairs of capabilities: the *enter*/*accept* capability pair (RedIn rule and Fig. 4B) is required for one ambient to enter a sibling accepting ambient; the *exit*/*expel* capability pair (RedOut rule and Fig. 4C) is required for an ambient to exit its parent (expelling) ambient; and the *merge+*/*merge−* capability pair (RedMerge rule and Fig. 4D) is required for one ambient to merge with another (sibling) ambient.

Ambient boundaries restrict communication between processes. We distinguish three communication directions using appropriate *labels*: *local* communication (RedLocal rule

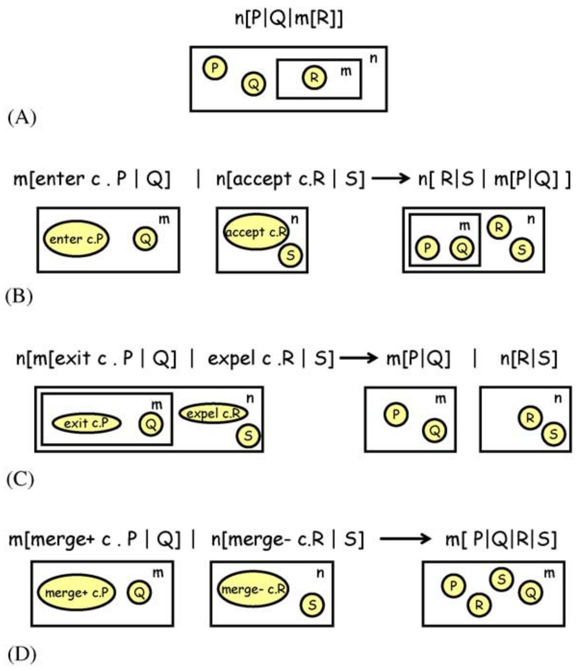

n[P|Q|m[R]]
(A)
m[enter c . P | Q] | n[accept c.R | S] ⟶ n[R|S | m[P|Q]]
enter c.P
accept c.R
(B)
n[m[exit c . P | Q] | expel c .R | S] ⟶ m[P|Q] | n[R|S]
exit c.P
expel c.R
(C)
m[merge+ c . P | Q] | n[merge- c.R | S] ⟶ m[P|Q|R|S]
merge+ c.P
merge- c.R
(D)

Fig. 4. Ambient moves. (A) Ambient *m* (child) inside ambient *n* (parent). (B) Entry of ambient *m* into (sibling) ambient *n*. (C) Exit of ambient *m* out of (parent) ambient *n*. (D) Merge of sibling ambients *m* and *n*. (These are simplified versions of the rules in Fig. 3).

and Fig. 5A) occurs between two processes residing in the same (immediate) ambient; *s*2*s* communication (RedSibling rule and Fig. 5B) occurs between two processes residing in two (immediate) sibling ambients; and *p*2*c*/*c*2*p* communication occurring between a process in a parent ambient and a process in its (immediate) child ambient. This latter communication is asymmetric, with two reduction rules (RedParentOutput and RedParentInput), depending on the location of the sender and the receiver (Fig. 5C and D). Directions are independent of the channel's identity as a public or private channel. These distinctions are kept in combination with the channel's direction. For example, the same private channel *name* may be used for communication between siblings (when both own the same private name) as well as for local communication between processes in the same ambient.

2.2.2. Stochastic semantics

Like the π-calculus, the original ambient calculus is non-deterministic, and all enabled capabilities and communications are equally likely to occur. To better suit biomolecular systems, we adapt the ambient calculus to a stochastic framework by extending our application [17] of the Gillespie [6] algorithm to (movement) capabilities in addition to communications.

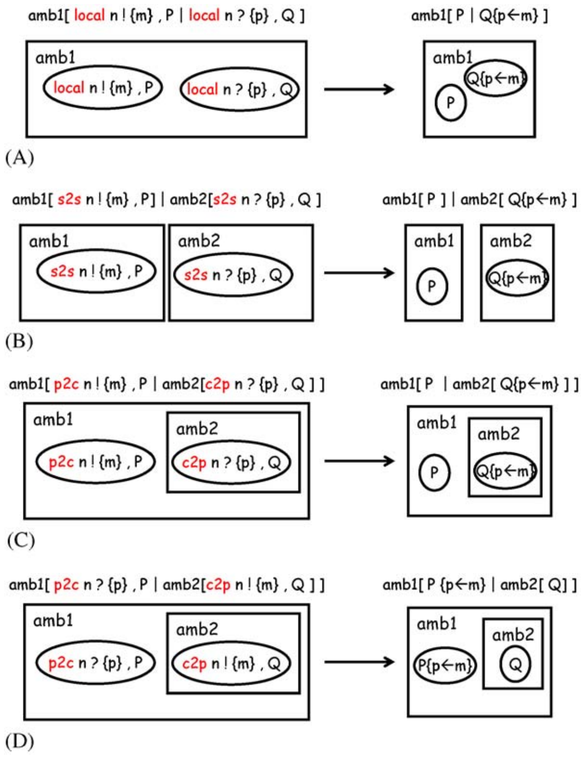

amb1[local n ! {m} , P | local n ? {p} , Q]
amb1[P | Q{p←m}]
amb1
local n ! {m} , P
local n ? {p} , Q
Q{p←m}
P
(A)
amb1[s2s n ! {m} , P] | amb2[s2s n ? {p} , Q]
amb1[P] | amb2[Q{p←m}]
amb1
amb2
s2s n ! {m} , P
s2s n ? {p} , Q
(B)
amb1[p2c n ! {m} , P | amb2[c2p n ? {p} , Q]]
amb1[P | amb2[Q{p←m}]]
p2c n ! {m} , P
c2p n ? {p} , Q
(C)
amb1[p2c n ? {p} , P | amb2[c2p n ! {m} , Q]]
amb1[P {p←m} | amb2[Q]]
p2c n ? {p} , P
c2p n ! {m} , Q
P{p←m}
Q
(D)

Fig. 5. Communication with ambients. (A) *Local* communication between two processes in the same ambient. (B) *s*2*s* communication between processes in two sibling ambients. (C) *p*2*c*/*c*2*p* communication from a process in a parent ambient to a process in a child ambient. (D) *c*2*p*/*p*2*c* communication from a process in a child ambient to a process in its parent ambient.

Briefly, the Gillespie algorithm provides an accurate mechanism for the stochastic time evolution of a biomolecular system. Based on the basal rate of the reactions in the system and the quantities of reactants, the algorithm selects at each step the next reaction to occur and a time step to advance the system's "clock". In our application of the Gillespie algorithm in the biochemical stochastic π-calculus [17], each reaction is represented by a communication channel, assigned with a *Base_rate*, and the quantities of reactant molecules are represented by the quantities of the processes offering to send and receive on the channel.

Since capabilities are represented as analogs of communication channels, the Gillespie algorithm is easily extended to handle BioAmbients, by allowing the algorithm to select any of the enabled events (either a communication or a capability). Note, that each combination of channel, direction, and configuration represents a separate "reaction" in the Gillespie algorithm. This simple extension does not appear to require an essential modification of the semantics presented in [17], but the properties of the resulting system are still to be studied.

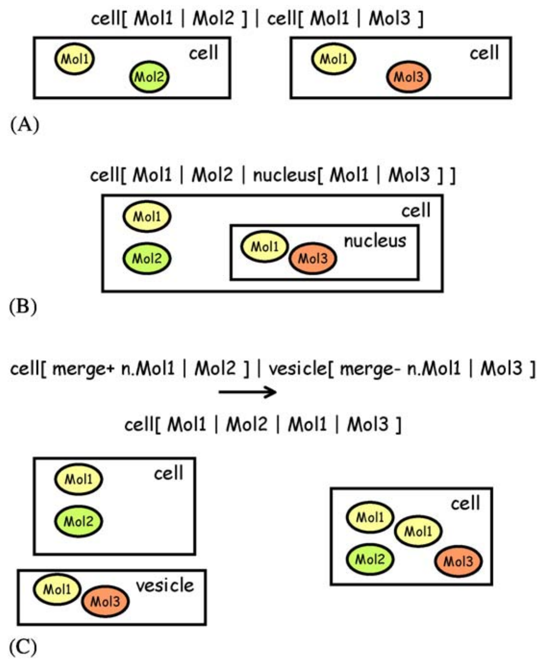

cell[Mol1 | Mol2] | cell[Mol1 | Mol3]
cell
Mol1
Mol2
Mol3
(A)
cell[Mol1 | Mol2 | nucleus[Mol1 | Mol3]]
nucleus
(B)
cell[merge+ n.Mol1 | Mol2] | vesicle[merge- n.Mol1 | Mol3]
cell[Mol1 | Mol2 | Mol1 | Mol3]
vesicle
(C)

Fig. 6. Basic use of BioAmbients: membrane-bound compartments. (A) Membrane-bound compartments as ambients. (B) Nested compartments as nested ambients. (C) Membrane fusion as ambient merger.

2.3. The "ambient as biological compartment" abstraction

To describe the application of the BioAmbients abstraction, we now turn to a series of biological examples, which we will abstract using the formal constructs and rules unique to BioAmbients.

2.3.1. Membrane-bound compartments as ambients

We abstract membrane-bound compartments as ambients. For example, a system with several cells (Fig. 6A), each with several molecules inside, is modeled by $System ::= cell[Mol|\cdots|Mol]|\cdots|cell[Mol|\cdots|Mol]$. Compartments hierarchy is abstracted as ambient nesting. For example, a cell, which in addition to several molecules also has a nested nucleus compartment, is represented by $Cell ::= cell[Mol|\cdots|Mol|nucleus[Mol|\cdots|Mol]]$. Here, the nesting of ambient brackets abstracts the hierarchy of membrane-bound compartments (Fig. 6B).

2.3.2. Membrane fusion as ambient merger

The most common change in membrane-bound compartments is their merger by membrane fusion. This is the case when vesicles fuse into the ER and Golgi apparatus, or when a virus enters a cell. Fusion requires specific interaction between molecules in the two compartments (e.g. receptors) and is abstracted by complementary *merge* capabilities on a specific channel in two processes in two sibling ambients, for example

$$cell[merge+\ n.Mol_1 \mid \cdots \mid Mol_n] \mid ves[merge-\ n.VMol_1 \mid \cdots \mid VMol_n].$$

As a result of compartment fusion, the organization of the system changes. The two compartments are unified and their contents are united together, wrapped by a single membrane. This is reflected by the result of the *merge* operation: $cell[Mol_1 \mid \cdots \mid Mol_n \mid VMol_1 \mid \cdots \mid VMol_n]$ (Fig. 6C). Any internal structure (i.e. sub-ambients) of the merging ambients would be preserved by this operation, abstracting the preservation of sub-compartments. Note, that only “sibling” compartments, not separated by a third membrane, may fuse to one another. Correspondingly, the abstracted *merge* operation is allowed only if the two ambients are siblings. Ambient entry and exit similarly abstracts the entry and exit of membrane-bound compartments.[1]

2.3.3. Molecular compartments as ambients

We abstract molecular compartments, such as multi-domain molecules and molecular complexes, as ambients, too. For example, a protein with three domains is abstracted as a *protein* ambient with three resident processes, one per domain $protein[(\mathsf{new}\ backbone)\ Domain_1 \mid Domain_2 \mid Domain_3]$ (Fig. 7A). Note, that we may concomitantly abstract the molecular compartment using a private *backbone* channel. As we will presently see, this channel represents intra-molecular interaction once the molecule is part of a multi-molecular complex.

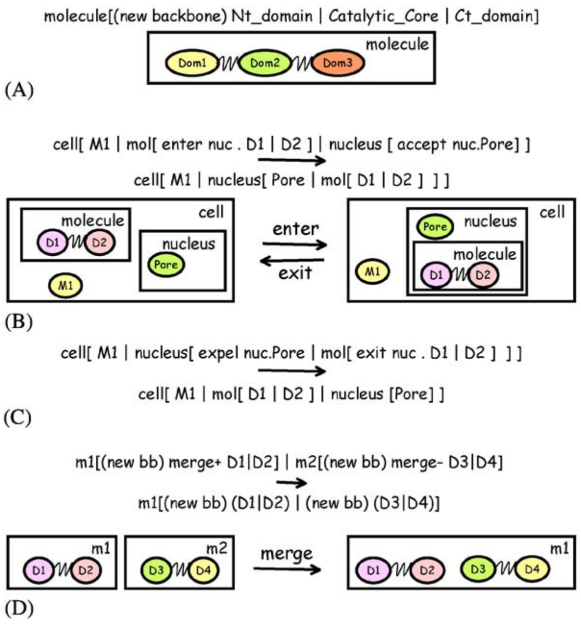
2.3.4. Molecule movement as ambient movement

The movement of molecules across membrane-bound compartments is initiated by specific interactions. It is now easily abstracted as entry or exit of ambients (representing molecules) to and from ambients (representing membrane-bound compartments). For example (Fig. 7B and C), consider a system where a two-domain protein molecule exits the nucleus by interaction between one of the protein’s domains (D1) and a pore protein in the nucleus’ envelope. We abstract the protein molecule as a *prot* ambient with two resident processes D_1 and D_2, and the nucleus as a *nucleus* ambient with a *Pore* process. Ambient exit is initiated by synchronization of an *exit nuc* capability in the D_1 process and an *expel nuc* capability in the *Pore* process (Fig. 7C), written as

$$cell[M_1 \mid nucleus[\ expel\ nuc.Pore \mid prot[exit\ nuc.D_1 \mid D_2]]].$$

While only one domain of the protein may interact directly with the pore, the entire molecule moves *as a whole*. Similarly, while only one process initiates the ambient’s

[1] Such entry and exit events, although rare, are relevant in some systems, such as *phagocytosis* or cell motility.



molecule[(new backbone) Nt_domain | Catalytic_Core | Ct_domain]
molecule
Dom1
Dom2
Dom3
(A)
cell[M1 | mol[enter nuc . D1 | D2] | nucleus [accept nuc.Pore]]
cell[M1 | nucleus[Pore | mol[D1 | D2]]]
molecule
cell
D1
D2
nucleus
Pore
M1
enter
exit
(B)
cell[M1 | nucleus[expel nuc.Pore | mol[exit nuc . D1 | D2]]]
cell[M1 | mol[D1 | D2] | nucleus [Pore]]
(C)
m1[(new bb) merge+ D1|D2] | m2[(new bb) merge- D3|D4]
m1[(new bb) (D1|D2) | (new bb) (D3|D4)]
m1
m2
D3
D4
merge
(D)

Fig. 7. Basic use of BioAmbients: molecular compartments. (A) Molecular compartments as ambients. (B,C) Molecular movement as ambient entry and exit, respectively. (D) Complex formation as ambient merger.

move, the entire ambient moves as a whole, leading to $cell[\ prot[D_1|D_2]\,|\,M_1\,|\,nucleus\ [Pore]\ ]$.

2.3.5. Complex formation as ambient merger

The content of molecular compartments may change dynamically (e.g. complex formation) due to specific molecular interactions. We abstract this by the *merge* operation (Fig. 7D), synchronized on a specific channel. For example, consider the formation of a complex between two proteins, each with two domains. We abstract the two proteins as the *prot*1 and *prot*2 ambients, each with two sub-processes (D_1 and D_2 in the former and D_3 and D_4 in the latter) with specific *merge* capabilities on the *bind* channel, written as

$$prot1[(\mathsf{new}\,bb)merge+\ \ bind.D_1|D_2]\,|\,prot2[(\mathsf{new}\,bb)merge-\ \ bind.D_3|D_4].$$

In forming a complex, while only one domain participates directly in the interaction, all the domains end up in the same complex. Similarly, upon ambient *merge* the resulting single ambient contains all the contents of the merging ambients, e.g. $prot1[(\mathsf{new}\,bb)(D_1|D_2)|(\mathsf{new}\,bb)(D_3|D_4)]$. Note, that the two private backbone channels (one *bb* per each protein) are maintained throughout the *merge* operation, and abstract individual protein "identity" in the complex. We subsequently use them to abstract complex breakage, as shown in Section 3.2.

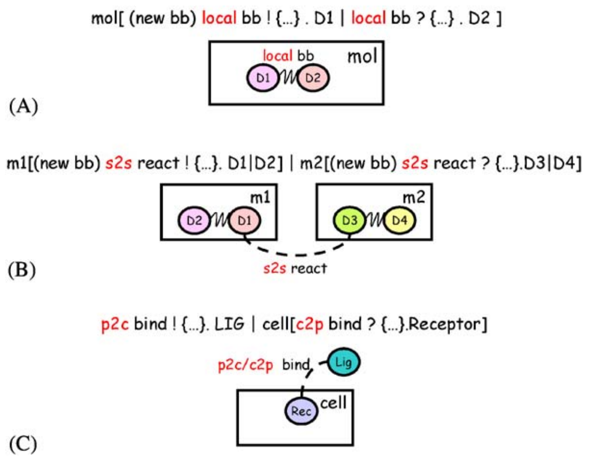

mol[(new bb) local bb ! {...} . D1 | local bb ? {...} . D2]
local bb
mol
D1
D2
(A)
m1[(new bb) s2s react ! {...}. D1|D2] | m2[(new bb) s2s react ? {...}.D3|D4]
m1
m2
D2
D1
D3
D4
s2s react
(B)
p2c bind ! {...}. LIG | cell[c2p bind ? {...}.Receptor]
p2c/c2p bind
Lig
Rec
cell
(C)

Fig. 8. Basic use of BioAmbients: molecular interactions. (A) Intra-molecular interaction as local communication. (B) Inter-molecular interaction as sibling communication. (C) Transmembranal interaction as parent–child communication.

2.3.6. Compartment limitation on interaction as ambient restriction of communication

Membrane-bound and molecular compartments restrict molecular interaction. Molecules that reside within membrane-bound compartments may typically interact only with molecules in the same compartment, while cross-membrane molecules may participate in interactions in two compartments. Molecular compartments also restrict some interactions (e.g. intra-molecular or intra-complex interactions) but not others (e.g. inter-molecular or inter-complex interactions). We abstract the different restrictions on molecular interaction using three types of communication *directions* restricted by ambient boundaries: *local*, *siblings*, and *parent–child*.

Intra-compartment interaction is abstracted by the *local* communication direction, allowing communication only between processes in the same ambient. For example, consider a specific interaction between two domains of the same molecule. If we abstract the molecule as a *mol* ambient with two processes, D_1 and D_2, then the potential intra-molecular interaction is represented by complementary *local* communication actions on a communication channel, e.g. $mol[local\, tyr\ !\ \{\cdots\}.D_1 \mid local\, tyr\ ?\ \{\cdots\}.D_2]$ (Fig. 8A).

Interaction between molecular compartments is abstracted by the $s2s$ communication direction, allowing communication between processes in immediate sibling ambients. When we abstract complexes and multi-domain molecules by ambients, all inter-molecular interaction is abstracted by $s2s$ communication,[2] e.g. $prot1[s2s\, react\ !\ \{\cdots\}.D_1|D_2] \mid prot2[s2s\, react\ ?\ \{\cdots\}.D_3|D_4]$ (Fig. 8B).

[2] In practice, we take a hybrid approach when more complex molecules are abstracted as ambients while others are abstracted as naked processes. The parent–child communication ensures this approach's feasibility, as described below.

Interaction of a membrane-embedded molecule (e.g. receptor) with a molecule outside its compartment (e.g. ligand) is abstracted by the *p*2*c* and *c*2*p* complementary directions. These allow interaction between a process in a child ambient and a process in a parent ambient (note the inherent asymmetry of these directions). For example, for the binding of a ligand to a receptor (Fig. 8C), we write *p*2*c bind* ! $\{\cdots\}$.*Lig* | *cell*[*c*2*p bind* ? $\{\cdots\}$.*Receptor*]. Note, that since we allow communication to cross at most one ambient boundary (no grandchild to grandparent communication), the receptor must be abstracted as a "naked" process and not as a molecular ambient.

3. Simple examples: transport, enzymes and complexes

To illustrate the utility of the "compartment as ambient" abstraction, we next follow a few simple BioAmbients programs representing biological systems.

3.1. Transport

Our first example is a membranal pore, which allows bi-directional passage of molecules across a membrane. Passage depends on specific interaction between the molecule and the pore.

We abstract the cell and each of the molecules as ambients (*cell*[$\cdots$] and *molecule* [$\cdots$], respectively). To represent the molecule's ability to pass through the pore, we equip each *molecule* ambient with a process (*Mol*), with *choice* between an *enter* and an *exit* capability, synchronized on the *cell*1 and *cell*2 channels, respectively. The complementary ability of the pore to let the molecule through is represented by the complementary choice in a *Porin* process to *accept* and *expel* on the corresponding channels. Note, that the membrane-embedded porin is abstracted as a naked process residing within the *cell* ambient, while other molecules (that are either properly in or out of the cell) are abstracted as processes encapsulated in ambients. The resulting code is shown in Fig. 9.

An illustration of the operation of a toy system representing one molecule, one cell and one pore is shown in Fig. 10. We start when the *molecule* ambient is outside the *cell* ambient. The *molecule* enters the *cell* by a synchronized *enter–accept* on *cell*1 between the *Mol* and *Porin* in the sibling ambients (the *exit–expel* option is irrelevant at this point, as it requires one ambient to reside within the other). As a result, the system now consists of the *molecule* ambient within the *cell* ambient and the *enter–accept* option is no longer relevant. Rather, the *exit–expel* capability pair on *cell*2

```
System::= molecule[Mol] | ... | molecule[Mol] | cell[Porin]
     Mol::= enter cell1 . Mol +
            exit  cell2 . Mol
     Porin::= accept cell1 . Porin +
              expel  cell2 . Porin
```

Fig. 9. BioAmbients code for the porin example.

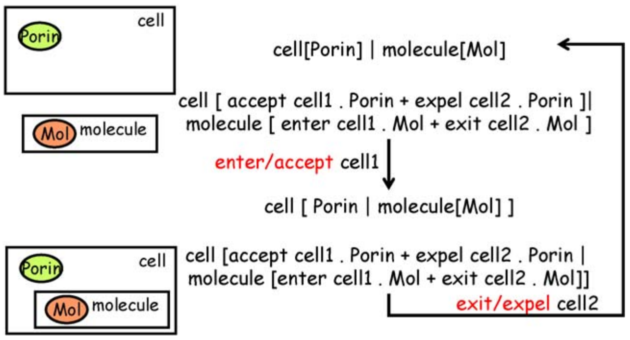

cell
Porin
Mol
molecule
cell[Porin] | molecule[Mol]
cell [accept cell1 . Porin + expel cell2 . Porin]|
molecule [enter cell1 . Mol + exit cell2 . Mol]
enter/accept cell1
cell [Porin | molecule[Mol]]
cell [accept cell1 . Porin + expel cell2 . Porin |
molecule [enter cell1 . Mol + exit cell2 . Mol]]
exit/expel cell2

Fig. 10. Porin example. An illustration of the BioAmbients reduction steps representing entry and exit of a molecule to and from a cell via a pore.

```
System::= molecule[ProteinA] | ... | molecule[ProteinA] |
          molecule[ProteinB] | ... | molecule[ProteinB]
ProteinA::= D3
  D3::= merge- complexAB . BoundD3
  BoundD3::= local breakAB ? {...} . expel breakAB1 . D3
ProteinB::= (new bb) D1 | D2
  D1::= merge+ complexAB . BD1
  BD1::= local breakAB ! {...} . local bb ! {...} .
              molecule[merge+ bb . exit breakAB1 . D1]
  D2::= local bb ? {...} . molecule[merge- bb . D2]
```

Fig. 11. BioAmbients code for a two-protein complex.

can be used by *Porin* in the parent *cell* and *Mol* in the child *molecule*, to pass the *molecule* outside the *cell*. Note, that this process can iterate forever.

3.2. Protein complexes

As shown above, the formation of a multi-protein complex is easily abstracted as ambient merger. As a result, a single ambient forms where the resident processes represent the domains of different proteins. This allows us to abstract intra- and inter-complex interactions as local and sibling communication, respectively. However, it also introduces a difficulty when we wish to abstract the reverse event of complex breakage: how can we identify the individual domains of one protein in the abstracted representation when the processes that represent them are no longer encapsulated by an exclusive ambient boundary?

There are two solutions to this problem. First, we can use private channels to sustain a specific link between the processes representing independent domains of a single protein, similar to the approach we applied prior to introducing ambients. For example (Figs. 11 and 12) consider a complex formed between one protein molecule

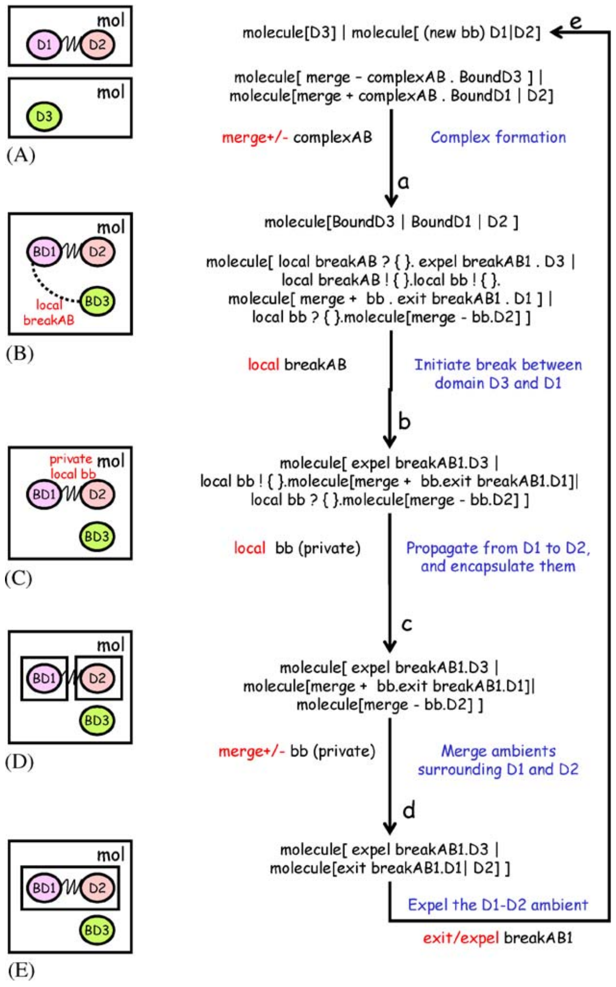

mol
D1
D2
D3
(A)
molecule[D3] | molecule[(new bb) D1|D2]
e
molecule[merge - complexAB . BoundD3] |
molecule[merge + complexAB . BoundD1 | D2]
merge+/- complexAB
Complex formation
a
BD1
BD3
local breakAB
(B)
molecule[BoundD3 | BoundD1 | D2]
molecule[local breakAB ? { }. expel breakAB1 . D3 |
local breakAB ! { }.local bb ! { }.
molecule[merge + bb . exit breakAB1 . D1] |
local bb ? { }.molecule[merge - bb.D2]]
local breakAB
Initiate break between domain D3 and D1
b
private local bb
(C)
molecule[expel breakAB1.D3 |
local bb ! { }.molecule[merge + bb.exit breakAB1.D1]|
local bb ? { }.molecule[merge - bb.D2]]
local bb (private)
Propagate from D1 to D2, and encapsulate them
c
(D)
molecule[expel breakAB1.D3 |
molecule[merge + bb.exit breakAB1.D1]|
molecule[merge - bb.D2]]
merge+/- bb (private)
Merge ambients surrounding D1 and D2
d
(E)
molecule[expel breakAB1.D3 |
molecule[exit breakAB1.D1| D2]]
Expel the D1-D2 ambient
exit/expel breakAB1

Fig. 12. Protein complex example. An illustration of the BioAmbients reduction steps representing formation and breakage of a two-protein complex. While complex formation is represented by a single step (A), breakage requires several steps, from initiation of the break (B), through inter-process interactions (C,D) and the eventual separation of the two ambients (E).

(*ProteinA* process) with a single domain (*D*3 process) and a second molecule (*ProteinB*) comprised of two domains (*D*1 and *D*2). First, we represent complex formation by exercising complementary *merge* capabilities in *D*3 and *D*1 on the *complexAB* channel (Fig. 12A). As a result, *D*1, *D*2 and *D*3 all reside within the same ambient, representing the complex. In order to distinguish the processes according to their origin, we employ a private backbone channel *bb*. This private channel is declared when *ProteinB* is originally created, and is thus known only to its two sub-processes, *D*1 and *D*2, but not to *ProteinA*'s *D*3.

The private channel will be used when abstracting complex breakage. The event is initiated by a *local communication* on *breakAB* between *BD*3 and *BD*1 representing the directly bound domains (Fig. 12B). In the next three steps, *BD*1 uses the private *bb* channel to coordinate the simultaneous exit of itself and *D*2 as a single ambient. First, the event is propagated from *BD*1 to its companion *BD*2, by a local communication on *bb*. This is followed by separate encapsulation of each of the two processes (Fig. 12C). The two encapsulated processes then use the same private *bb* channel name to merge and form a single ambient, embedded within the original complex ambient (Fig. 12D). Finally, an *exit–expel* capability pair on the *breakAB*1 channel between *BD*1 and *BD*3 completes complex breakage resulting in two sibling ambients, one per protein (Fig. 12E).

Note, that as we are using a multi-step scenario to model an "atomic" event, we must ensure its biochemical accuracy by assigning appropriate channel rates. With the exception of the *breakAB* channel (used to initiate the break), all the "intermediate" channels (*bb*, *breakAB*1) are *instantaneous*. Instantaneous channels [17] have an *infinite* rate, such that communications on them are executed immediately when enabled, are not queued by the Gillespie algorithm and do not affect the time evolution of the system. Rather, the rate of the entire scenario is determined by the rate associated with the *breakAB* channel.

3.3. Enzymes

An alternative way to abstract complexes is to replace ambient merger by entry of one ambient into another. This representation, albeit simpler, is inherently asymmetric, and creates a double barrier between the nested ambient and the general environment (its own ambient boundary and that of its including parent). Thus, we use this approach mostly to abstract transient complexes, where one expects only limited interaction between the complexed proteins and the external environment.

A prime example for such a scenario is the enzyme–substrate complex. The general framework for representing enzymatic reactions as the movement of molecular ambients is shown in Fig. 13. We abstract the enzyme and each of its substrates as a separate ambient (with a process inside) and enzyme–substrate binding is modeled as entry of the *substrate* ambient to the *enzyme* ambient. For a reversible single-substrate reaction (Fig. 13A), both forward reaction (or product release) and substrate unbinding (or reverse reaction) are abstracted as ambient exit, highlighting the symmetry of both directions. For a bi-substrate reaction (Fig. 13B), the model is more complex, with a sibling to sibling communication between the two *substrate* ambients inside the *enzyme* ambient representing the reaction (preceded and followed by *enter* and *exit* events).

A toy example of a reversible single substrate reaction is shown in Figs. 14 and 15. The enzyme, substrate, and product are represented by three ambients harboring the *E*, *S*, and *P* processes. *enter–accept* capabilities on *e_s_bind* and on *e_p_bind* represent enzyme–substrate and enzyme–product binding, respectively. The resulting *ES* complex is abstracted as a nested structure with the *substrate* ambient inside the *enzyme* one. Next, an *exit–expel* event occurs on either the *unbind* channel (resulting in release of the ambient with an intact *S*), or the *react* channel (releasing the ambient with a

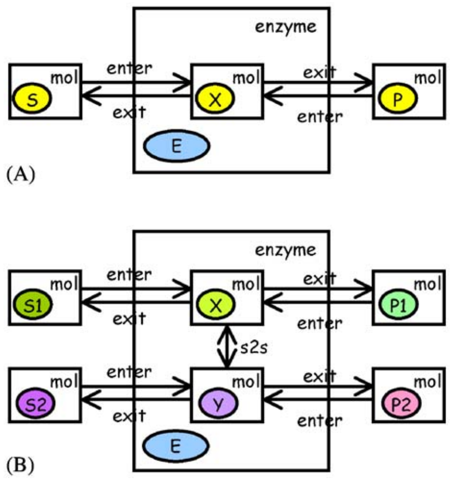


Fig. 13. Enzymes as molecular ambients. Schematic representation of reversible single substrate (A) and bi-substrate (B) reactions with BioAmbients.

```
System::= enzyme[E] | ... | enzyme[E] |
          molecule[S] | ... | molecule[S]
E::= accept e_s_bind . ES + accept e_p_bind . ES
ES::= expel unbind . E + expel react . E
S::= enter e_s_bind . X
X::= exit unbind . S + exit react . P
P::= enter e_p_bind . X
```

Fig. 14. BioAmbients code for a single substrate enzymatic reaction.

product P). Note that the channel names (*unbind*, *react*) were given with the forward reaction in mind, but in fact represent the same type of molecular event. Thus, both sides of the reaction are seamlessly and symmetrically represented.

4. BioSpi 3.0: extending BioSpi to simulate compartmentalized systems

Guided by BioAmbients, we extended the BioSpi simulation system to handle compartments.[3] The BioSpi 2.0 system [17,19] receives as input stochastic π-calculus code and executes it. In the simulation, each instance of a π-calculus process is realized as a running computational one. Processes run concurrently and interact using channel objects, following the reaction rules of the calculus. The simulation follows the

[3] Note that the properties of a stochastic version of BioAmbients require further study (see Section 2.2.2). Thus, while BioSpi 3.0 is inspired and guided by (the non-deterministic) BioAmbients, it is not equivalent.

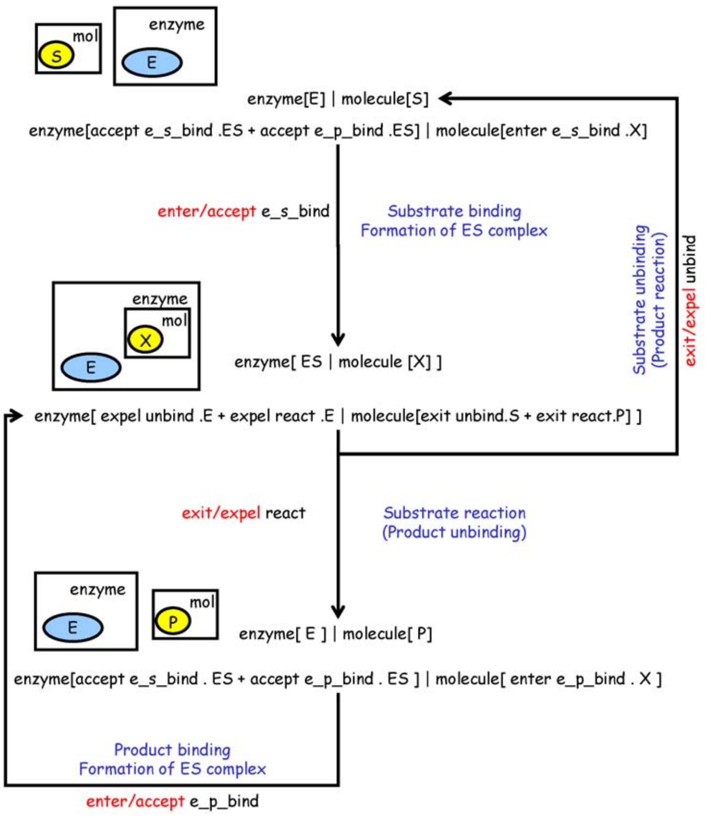

mol
S
enzyme
E
enzyme[E] | molecule[S]
enzyme[accept e_s_bind .ES + accept e_p_bind .ES] | molecule[enter e_s_bind .X]
enter/accept e_s_bind
Substrate binding
Formation of ES complex
Substrate unbinding
(Product reaction)
exit/expel unbind
enzyme
mol
X
E
enzyme[ES | molecule [X]]
enzyme[expel unbind .E + expel react .E | molecule[exit unbind.S + exit react.P]]
exit/expel react
Substrate reaction
(Product unbinding)
enzyme
E
mol
P
enzyme[E] | molecule[P]
enzyme[accept e_s_bind . ES + accept e_p_bind . ES] | molecule[enter e_p_bind . X]
Product binding
Formation of ES complex
enter/accept e_p_bind

Fig. 15. Single-substrate enzymatic reaction. An illustration of the BioAmbients reduction steps representing a reversible single-substrate enzymatic reaction.

time evolution of the system by following the Gillespie algorithm. The implementation of the algorithm is coordinated by a central general process (termed “monitor”), which at each step, selects a single communication (reaction) to be realized in one atomic operation, and advances the clock. BioSpi is based on the Logix system [22], an implementation of Flat Concurrent Prolog (FCP [21]). FCP provides us with unique mechanisms to implement both mobility (passing logical variables in messages) as well as full synchronized communication and choice with both output and input guards (using guarded atomic unification). Note, that previous implementations of the π-calculus or of related formalisms (e.g. [13] and references therein) do not provide such full guarded synchronous communication.

The adaptation of BioSpi to accommodate BioAmbients entailed three major changes: incorporation of a hierarchical ambient tree, handling ambient capabilities (entry, exit, merge), and scoping communication by ambient organization, distinguishing between local, sibling, and parent–child directions.

4.1. The ambient tree hierarchy

Ambients are transformed to FCP [22] procedures, and organized as a tree, permitting inter-ambient communication. Each node in the tree may include active processes and channels and is identified by a unique name. Within a given run, a communication or capability offer from a process is first interpreted by the FCP procedure representing the ambient in which the process resides, before being forwarded to the central monitor. The stochastic simulation is carried out by the monitor as before, and a single communication or capability out of the entire system is chosen at a time for execution. The ambient tree structure may change during the course of a given run as a result of exercising capabilities (exit, enter, merge) or due to a declaration of a sub-ambient within a process. Both cases are detailed next.

4.2. Channels, communication and capabilities

Since channels are used in BioAmbients for six different types of synchronous actions (three for communication and three for capabilities), BioSpi's channel scheme is extended to handle the different kinds of interaction. Importantly, we want a channel name to be declared only once, and allow its use for the different actions in a context-sensitive manner. To this end, we distinguish between two kinds of channels: internal and external. An internal channel may be used for intra-ambient communication (*local*), while an external channel may be used either for inter-ambient communication ($s2s$, $p2c$–$c2p$) or to assert a capability (*merge*, *enter*, *exit*). At first, all channels are created as internal channels. Then, an external channel of a particular type is automatically derived from an internal channel when an action of that type is first declared on that channel in a given ambient. Thus, channels which share names but have different types or reside in different ambients are separate channels and when an internal channel's *name* is sent to another ambient, the receiving ambient creates or associates a local internal version of it. As before, a channel may be either public or private. Public channel names are globally defined, while private channel names are declared locally by a specific process.

A special case is introduced when the ambient tree structure changes. When exercising capabilities, the channel sets associated with the moving or merging ambients or their new parents may require modification. When an ambient *exit*s from its parent or *enter*s a sibling ambient, all of its references to non-*local* channels of its parent are transferred to the corresponding channels of the new parent. This may require creation of non-*local* channels in the new parent. When an ambient *merge*s into a sibling ambient, all of its channels and references to them are transferred to the merged ambient. When a new ambient is declared, we consider all the local channels in the ambient of the declaring process. All such channels which are needed by the transitive closure of the new ambient and its continuations are inherited by the new ambient. A fresh internal version of each channel is created, and resides in the new ambient.

4.3. Tracing and recording BioAmbients simulations

Like previous versions, BioSpi 3.0 maintains a full *record* of the time evolution of a simulation session. The record specifies each communication in the system, the time at which it occurred, the communicating processes, the channel on which the communication occurred and the resulting processes. Importantly, the BioSpi 3.0 record also allows us to either count processes per ambient, across all ambients, or across a specific type of ambient, as specified by its name. In addition, the ambient tree of a computation can be displayed at different levels of detail, showing node hierarchy and names as well as processes and channels associated with each node, and providing us with a comprehensive snapshot of system organization. Specific filters may be used to show only some of the channels, specific details on the channel, or a partial sub-tree rooted at a specific ambient, a single specific node, or a class of nodes.

5. Multi-level models

We conclude this work with an example of how the BioAmbients calculus can be used to model and study a highly complex multi-cellular, multi-level system.

5.1. The hypothalamic weight regulation system

The example we have chosen is the hypothalamic weight regulation system, which involves several levels of biological organization: molecular, cellular and anatomical. First, we briefly review the system. The balance between energy intake, expenditure and storage is determined by a tightly controlled feedback system for body weight regulation, involving complex physiological and molecular mechanisms. The central controller in this system resides in the hypothalamus region of the brain. This region receives input signals in the form of hormone molecules secreted by various tissues in the body and utilizes a complex neuronal and biochemical circuitry to integrate them and elicit output signals, which will affect the various physiological functions related to energy homeostasis. The response to these hormones is "computed" by the balance between an orexigenic biomolecular system, which induces energy accumulation and an anorexigenic one, which induces energy expenditure. The hypothalamus is organized into distinct compartments (termed "nuclei") each populated by one or more different types of neuron cells. We focus on the ARC nucleus and limit ourselves to a relatively simplified view of the events there (Fig. 16), composed of two large steps. In the first step (the first-order response), we distinguish between NPY/AgRP neurons that produce the orexigenic Neuropeptide Y (NPY) and AgRP hormones, and POMC/CART neurons that produce the anorexigenic neuropeptides αMSH and CART. Both neuron populations harbor receptors to the leptin and insulin hormone molecules. A rise in leptin and insulin levels induces expression and secretion of the anorexigenic peptides (αMSH, CART) and reduces expression and secretion of the orexigenic ones (NPY and AgRP). Fall in leptin levels has an opposite effect. The ARC neurons innervate several additional sites, and the released peptides elicit a second-order response and further

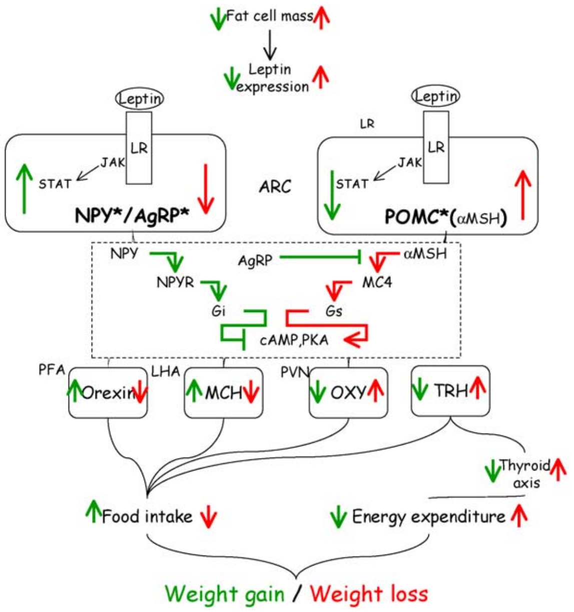


Fig. 16. Hypothalamic pathways for weight regulation. A partial view of molecular pathways, neurons and nuclei involved in weight control. Orexigenic (weight gaining) signals are in green, anorexigenic (weight loss) ones are in red. For further details see the main text. Adapted from [20].

coordinated synthesis of appropriate signaling neuropeptides (including the anorexigenic TRH, CRH, and oxytocin, and the orexigenic orexin and MCH). This involves binding of the peptides (synthesized in the first-order response) to specific receptors, as well as negative interference of orexigenic peptides (AgRP) with anorexigenic signaling (by αMSH). Many of the second-order neuropeptides may have a direct effect on the production of output signals, which affect peripheral systems, such as fat or muscle tissues. For example, CRH and TRH affect the hypothalamic–pituitary–adrenal axis and the thyroid axis, respectively. Both axes play a key role in the control of food intake and energy expenditure.

5.2. An ambient model for weight regulation

The fragment of the hypothalamic system for body weight regulation shown in Fig. 16 offers an interesting abstraction challenge as it requires us to simultaneously handle molecular events (receptors, signaling pathways and gene expression) for which variable degrees of knowledge exist, within a heterogeneous cell population (different types of neurons), which are further sequestered to distinct anatomical compartments.

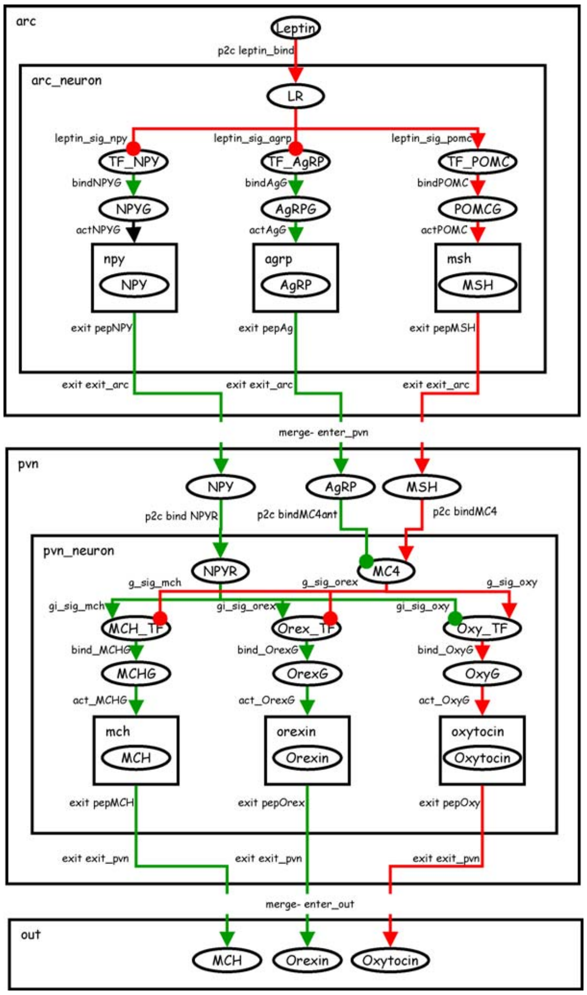


Fig. 17. A scheme for an ambient calculus model for hypothalamic weight regulation. The ambient model is depicted graphically, with ambients as rectangles and molecular processes as ovals. Channel names used in communications or capabilities are shown as labeled arrows, green and red for orexigenic and anorexigenic signals, respectively. Pointed arrowheads represent activatory events, round heads for inhibitory events.

We abstracted the system using BioAmbients (Fig. 17, and see [18]). First, we abstracted each hypothalamic nucleus as an ambient (*arc*, *pvn*), harboring ambients representing individual neurons (*arc_neuron*, *pvn_neuron*). For simplicity, we assumed a single neuron type per nucleus. Second, we represented the molecular components as processes. In addition, several general processes abstract the machineries for transcription, degradation and hormone export. The processes representing hormones (*Leptin* and *NPY*) reside outside the *neuron* ambients, and communicate with their cognate receptors (*LR* and *NPYR*, respectively) using *p2c/c2p* communication on specific channels. *MC*4 has a *choice* construct allowing it to communicate with one of two processes representing hormones: *AgRP* and *MSH*. The fact that only the latter is an agonist that would result in signaling is represented by the communication offered following the corresponding choice. The fact that the former blocks the receptor until unbound, is reflected by lack of such an offer. In all cases a private *unbind* channel is exchanged and is used for specific *p2c/c2p* communication, representing unbinding.

We abstract away some of the molecular details and allow direct communication between the processes representing receptors and those representing transcription factors via specific *s2s* channels. The TF-representing processes have two different states (*TF* and *TF_Active*) and switch between them according to communication from the processes representing activated (bound) receptors. In the *arc_neuron* ambient, the processes representing orexigenic TFs (*TF_NPY*, *TF_AgRP*) are switched to an "inhibited" state by communication from *LR* (leptin receptor), while the one representing anorexigenic TFs (*TF_POMC*) is switched to an "activated" state. A counter-active communication (representing a constitutive negative signal) is supplied by *ARC_PHOSPH* which communicates with the *TF*s, resetting them in the opposite direction. In the *pvn_neuron* ambient, there are two receptors. *MC*4 communicates with both the orexigenic *TF*s (*MCH_TF*, and *Orex_TF*) that switch to an inactive state, as well as with the anorexigenic ones (*Oxy_TF*) which switch to an active state. *NPYR* has the opposite effect. The processes representing active TFs can communicate with their cognate *Gene* processes and "bind them", switching the *Gene*s to an active state, that can result in release of new *Hormone*s (transcription and translation are abstracted in a single step). TF unbinding is abstracted as before by communication on private *unbind* channels, albeit *s2s* ones. *Hormone*s are first encapsulated in specific ambients (*npy*, *agrp*, *msh*, *mch*, *orexin*, *oxytocin*). The ambient membrane is essential for the export of the *Hormone*s to another *nucleus*. The abstraction of the export process involves three steps: exit of the *hormone* ambient from the *neuron* ambient, exit from the *nucleus*, and merge into another *nucleus*, thereby "spilling" the resident *Hormone* outside the *neuron* ambient. *Export* (in *neuron*) and *Boundary* (in *nucleus*) processes provide the complementary, synchronizing, capability. *Hormone_clearance* processes "remove" the generated *Hormone*s, by sending them an alert, resulting in the nullification of *Hormone* to an empty process.

To obtain a proof of principle of the operation of this model, we executed our BioAmbients program in BioSpi 3.0. Our choice of parameters here (initial number of cells and molecules, reaction rates) is relatively arbitrary. Reactions have a uniform relative rate of 1 (except hormone unbinding and clearance rates which are 100-times slower) and we included 10 Leptin receptor processes (*LR*) and 100 *NPYR* and *MC*4 receptor processes per cell. For this initial evaluation only a single neuron ambient was included per nucleus. Our major test of this preliminary model's operation was to examine the levels of the six output *Hormone*s under four different levels of leptin in the ARC. When operating properly, the modeled system should generate high levels of anorexigenic hormone processes and low levels of orexigenic ones when leptin is high, and vice versa when leptin is low. Indeed, as shown in Fig. 18, the system has generally behaved as expected, lending support to this model.

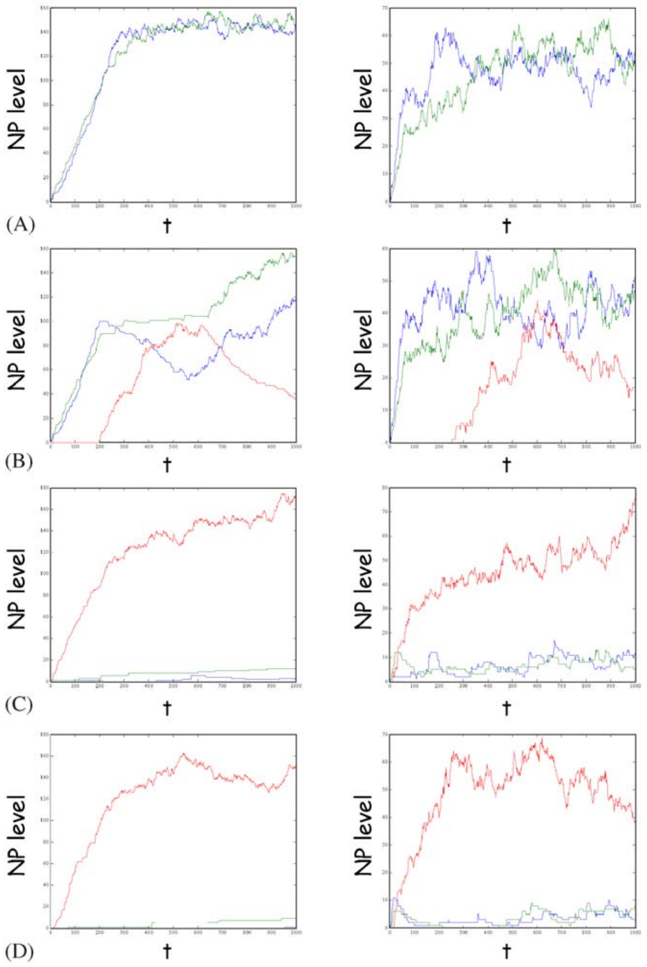


Fig. 18. Neuropeptide (NP) profiles under different levels of Leptin. Simulation results of the time evolution neuropeptide levels under various Leptin creation rates (A) 0.0001 (B) 0.01 (C) 1 (D) 100. In each panel, first-order hormones, AgRP (blue), NPY (green), and MSH (red) are shown on the right, and second-order hormones, MCH (blue), Orexin (green) and Oxytocin (red) are shown on the left. The anorexigenic hormones (red) are high when leptin levels are high, while the orexigenic ones (green,blue) are high when Leptin is low, as expected.

6. Perspective: the compartment as ambient extension

The original ambient calculus contains the critical component necessary to abstract biological compartments—the bounded ambient—but is not accurate enough to model biology and required several modifications.

- In biology, both movement of compartments and of molecules across compartments requires some interaction between the moving entity and the compartment which it attempts to “cross”. In the original ambient calculus, movement was asynchronous and could be initiated in a one-sided manner by a process in the moving entity. Furthermore, movement in the original ambient calculus required the moving entity to “know” a single ambient name, which allowed all the exit and entry operations from and to a given ambient. Biological compartments (both membrane-bound and molecular), however, may be entered, exited or joined by molecules via multiple routes, with different specificities and rates. To handle these two limitations in BioAmbients, we essentially abolished ambient names and replaced the unilateral movement capabilities with bi-lateral ones that are synchronized on specific channels. Thus, movement now requires synchronization between two ambients, and may be done in different ways by using different channel names.

- Interaction in biology is synchronous, and may occur both within the compartment boundary and across it, albeit only in specific configurations. In the original ambient calculus, communication events are asynchronous, and all communication is purely local. To allow cross-boundary communication, we equipped BioAmbients with two additional communication directions: sibling (to handle molecular compartments) and parent–child (to handle cross-membrane molecules).
- The semantics of both moves and communication in the original ambient calculus is non-deterministic, while an accurate abstraction must be quantitative [17]. To this end, we supplied BioAmbients with stochastic semantics, along the lines of our biochemical stochastic π-calculus.
- The original ambient calculus did not provide a primitive capability for ambient merger. In biology, both the merge of molecular compartments and that of membrane-bound compartments is prevalent. We therefore added in BioAmbients the “*merge*+ /*merge*−” primitive pair.
- Some of the components of previous variants of the ambient calculus are actually superfluous for the abstraction of biological systems. These included the movement of “naked” processes across boundaries and ambient names. We removed these from the BioAmbients variant.

The “compartment as ambient” abstraction extends the “molecule as computation” abstraction based on the π-calculus in a significant but seamless way. It provides an easy mechanism to abstract biological compartments, without changes to the existing mathematical domain and without violating the semantic and pragmatic guidelines of the original abstraction. We have shown the utility of this extension to handle a variety of small, but essential, biological examples, as well as a complex realistic one, covering a wide spectrum of entities and events related to biological compartmentalization.

Our approach can be further improved in several ways. First, several useful extensions can be added to the current list of primitives to allow the direct modeling of unique biological events. Such extensions include a *kill* capability, to eliminate an ambient and all its contents in one primitive operation; an *acid* capability, to remove an ambient membrane and merge its contents with the parent ambient; a *duplicate* capability, to create two identical sibling ambients from a single one; and a *divide* capability in which the contents of the ambient are randomly split into two siblings. While the former three are relatively straightforward to define and implement, the latter one is more challenging. Note, that in all cases these capabilities will not require synchronization, and thus deviate from our current communication and capability model. Other extensions could simplify some of the current multi-step modeling. In particular, complex breakage is now relatively complicated and a *split–break* capability would be highly useful, but cannot be rigorously defined within the current framework. One possibility is to add a "transparent boundary", which is formed around a protein process during a merge (replacing the old usual ambient boundary), does not create any limitation on communication during the merged state, but serves to identify all of that protein's "pieces" necessary to exercise a *split–break* capability. Transparent boundaries may also facilitate better representation of multi-domain, cross-compartment molecules, which we currently represent as "naked" processes offering child to parent communication.

Second, the hierarchical organization of ambients calls for a graphical representation. We used graphics informally throughout this work. However, these did not have well-defined syntax and semantics, and did not include dynamic information. The lack of a graphical component is a major disadvantage of our approach compared to graphical languages such as Statecharts [9]. We note that Bigraphs, recently developed by Milner [14] as a graphical formalism for concurrency, have striking resemblance to our informal models and have been shown adequate to represent both the π-calculus and the original ambient calculus. Recent work [13] explores its adaptation to *bio-graphs* suitable for our biological variant of the ambient calculus.

Finally, in handling compartmentalization, ambients provide us with a coarse-grained, albeit flexible, way to handle some aspects of the heterogeneous organization of biomolecular systems. A longer-term challenge is to fully handle three-dimensional space. A first step in this direction would be to handle space as a lattice of ambients, and diffusion of molecules as movement across them. This would ultimately require a new kinetic semantics as well.

References

[1] M. Ashburner, C.A. Ball, J.A. Blake, D. Botstein, H. Butler, J.M. Cherry, A.P. Davis, K. Dolinski, S.S. Dwight, J.T. Eppig, M.A. Harris, D.P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J.C. Matese, J.E. Richardson, M. Ringwald, G.M. Rubin, G. Sherlock, Gene ontology: tool for the unification of biology. The gene ontology consortium, Natur. Genetics 25 (1) (2000) 25–29.

[2] G.D. Bader, I. Donaldson, C. Wolting, B.F. Ouellette, T. Pawson, C.W. Hogue, Bind-the biomolecular interaction network database, Nucleic Acids Res. 29 (1) (2001) 242–245.

[3] L. Cardelli, Bioware languages, in: A. Herbert, K. Spärck Jones (Eds.), Computer Systems: Theory, Technology, and Applications — A Tribute to Rodger Needham, Springer, 2003.

[4] L. Cardelli, A.D. Gordon, Mobile ambients, in: Foundations of Software Science and Computation Structures: First Internat. Conf., FOSSACS '98, Springer, Berlin, 1998.

[5] V. Danos, C. Laneve, Graphs for core molecular biology, in: C. Priami (Ed.), Proceedings of the First Internat. Workshop on Computational Methods in Systems Biology, Lecture Notes in Computer Science, Vol. 2602, Springer, Berlin, 2003, pp. 34–46.

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

[7] P.J.E. Goss, J. Peccoud, Quantitative modeling of stochastic systems in molecular biology by using stochastic petri nets, Proc. Nat. Acad. Sci. USA 95 (12) (1998) 6750–6755.

[8] J.M. Haugh, D.A. Lauffenburger, Analysis of receptor internalization as a mechanism for modulating signal transduction, J. Theoret. Biol. 195 (2) (1998) 187–218.

[9] N. Kam, I.R. Cohen, D. Harel, The immune system as a reactive system: modeling T cell activation with statecharts, Bull. Math. Biol., 2002, to appear (an extended abstract of this paper appeared in the Proceeding of the Symposia on Human-Centric Computing Languages and Environments, Stresa, Italy, September 2001, pp. 15–22).
[10] A. Levchenko, J. Bruck, P.W. Sternberg, Scaffold proteins may biphasically affect the levels of mitogen-activated protein kinase signaling and reduce its threshold properties, Proc. Nat. Acad. Sci. USA 97 (2000) 5818–5823.
[11] H. Lodish, A. Berk, S.L. Zipursky, P. Matsudaira, D. Baltimore, J.E. Darnell, Molecular Cell Biology, 4th Edition, W.H. Freeman, New York, 2000.
[12] H. Matsuno, R. Murakani, R. Yamane, N. Yamasaki, S. Fujita, H. Yoshimori, S. Miyano, Boundary formation by notch signaling in Drosophila multicellular systems: experimental observations and gene network modeling by genomic object net, in: R.B. Altman, A.K. Dunker, L. Hunter, T.E. Klein (Eds.), Pacific Symp. on Biocomputing, World Scientific Press, Singapore, 2003, pp. 152–163.
[13] R. Milner, Communicating and Mobile Systems: the π-Calculus, Cambridge University Press, Cambridge, 1999.
[14] R. Milner, Bigraphical reactive systems, in: Proc. of the 12th Internat. Conf. on Concurrency Theory (CONCUR 2001), Lecture Notes in Computer Science, Vol. 2154, Springer, Berlin, 2001, pp. 16–35.
[15] M. Nagasaki, S. Onami, S. Miyano, H. Kitano, Bio-calculus: its concept and molecular interaction, in: Genome Informatics, Vol. 10, Universal Academy Press, Tokyo, 1999, pp. 133–143.
[16] G. Paun, Membrane Computing: An Introduction, Springer, Berlin, 2002.
[17] C. Priami, A. Regev, E. Shapiro, W. Silverman, Application of a stochastic name-passing calculus to representation and simulation of molecular processes, Inform. Process. Lett. 80 (2001) 25–31.
[18] A. Regev, The full code for this model is available from ftp://www.cgr.harvard.edu/public/ambients.
[19] A. Regev, W. Silverman, E. Shapiro, Representation and simulation of biochemical processes using the pi-calculus process algebra, in: R.B. Altman, A.K. Dunker, L. Hunter, T.E. Klein (Eds.), Pacific Symp. on Biocomputing, Vol. 6, World Scientific Press, Singapore, 2001, pp. 459–470.
[20] M.W. Schwartz, S.C. Woods, D. Porte, R.J. Seeley, D.G. Baskin, Central nervous system control of food intake, Nature 404 (2000) 661–671.
[21] E. Shapiro, Concurrent prolog: a progress report, in: E. Shapiro (Ed.), Concurrent Prolog, Vol. I, MIT Press, Cambridge, MA, 1987, pp. 157–187.
[22] W. Silverman, M. Hirsch, A. Houri, E. Shapiro, The Logix system user manual, version 1.21, in: E. Shapiro (Ed.), Concurrent Prolog, Vol. II, MIT Press, Cambridge, MA, 1987, pp. 46–78.
[23] E. Wingender, X. Chen, E. Fricke, R. Geffers, R. Hehl, I. Liebich, M. Krull, V. Matys, H. Michael, R. Ohnhauser, M. Pruss, F. Schacherer, S. Thiele, S. Urbach, The transfac system on gene expression regulation, Nucleic Acids Res. 29 (1) (2001) 281–283.