π@: A pi-based process calculus for the implementation of compartmentalised bio-inspired calculi *

Cristian Versari and Roberto Gorrieri

Università di Bologna, Dipartimento di Scienze dell'Informazione
Mura Anteo Zamboni 7, 40127 Bologna, Italy
{versari,gorrieri}@cs.unibo.it

Abstract. The modelling of biological systems led to the explicit introduction of compartments in several bio-oriented process calculi. In this tutorial we show how different compartment semantics can be obtained by means of a simple and conservative extension of the standard pi-calculus, the pi@ calculus. Significant examples are given through the encoding of two well known bio-inspired process calculi: BioAmbients and Brane Calculi.

Key words: pi-calculus, priority, polyadic synchronisation, BioAmbients, Brane Calculi

1 Introduction

There is an increasing awareness that complex biological systems can be better understood and studied at a system level, where a network of biochemical cells can be seen as a computing machinery, made of processing agents which interact and cooperate to achieve a common goal. This informal description applies to concurrent system as well, hence it is natural to use techniques from the global computing field to study the behaviour of biological cells. Particularly promising is the use of process calculi, which are formalisms used to describe concurrent and mobile systems. Process calculi are equipped with a formal semantics describing their behaviour, and plenty of tools for the static and dynamic analysis of systems have been produced, in particular simulators. These tools can be therefore used in the field of biological organisms, as well, to analyse and possibly predict their behavior. The pioneering work on modelling biochemical systems with a calculus is the work by Fontana and Buss [1] where a version of the lambda-calculus is used. A better account of pathways descriptions is proposed in [2] by Regev, Silverman and Shapiro, who use the π-calculus [3], a well-known formalism for mobility.

* This work has been partially sponsored by the PRIN 2006 Project BISCA – Sistemi e calcoli di ispirazione biologica e loro applicazioni.

This first case-study on the use of π-calculus for the modeling of biological processes raised an enormous interest for the potential application of process calculi techniques to Systems Biology. The direct employment of π-Calculus allowed the formalisation of several biological mechanisms, and even more by means of its variants and extensions [4–6] that permitted the representation or analysis *in silico* of complex cellular processes [7, 8]. Nonetheless, π-calculus is not the ultimate answer: it is difficult to model some other aspects of biological systems that cannot be ignored, if one wishes to obtain descriptions at a higher abstraction level and with higher biological faithfulness. A basic missing abstraction is the notion of *compartment*: a cell is composed of different materials that are confined in different areas of the cell (the compartments), usually separated by suitable *membranes* (i.e., semi-permeable barriers), ensuring that certain substances always stay into the compartment while other substances stay out of it. Similarly, any non-trivial biological system is a construct in which various components in different compartments execute different "computations", and the results of these computations can be communicated among the components through the compartments.

To this aim, more complex calculi have been proposed, such as Brane Calculi [9] and its variants [10], Bioambients [11], Beta binders [12, 13], the K-calculus [14], BioK [15] which are based on or get inspiration from π-calculus, enriched with some powerful mechanisms for compartmentalised computation. Even if they present many common features, each calculus focuses its attention on particular biological entities or mechanisms. Their similarity induces the interest for a parallel analysis, but their specialisation does not allow a direct comparison. Moreover, there is a lack of techniques and tool support for these calculi, because the added features are often radically new and the existing theories and tools cannot be adapted easily.

In this paper, we propose a novel calculus, called $\pi@$ (to be pronounced as the French word "paillette"), designed specifically to overcome the problems singled out above. It is a simple, conservative and powerful extension to π-calculus, with two new basic mechanisms: polyadic synchronisation and prioritised communication. In particular, polyadic synchronisation [16] allows for the modeling of compartments in a natural way, but still in the classic message-passing flavour typical of π-calculus.; priority instead is extremely useful for implementing transactional mechanisms that are essential when dealing with compartment operations (e.g., dissolution of a membrane) that involve many components that are to be updated consequently. Its simple syntax and semantics, very close to π-calculus, allow a natural extension of many properties and results already stated for standard π-calculus, thus facilitating $\pi@$ theoretical analysis. On the other hand, it is so powerful that it can be used with a pivotal role for comparing the various compartment-based formalisms. In this sense, we claim that $\pi@$ is the right compromise between the simple and elegant theory of π-calculus and the biological needs for modeling compartmentalised behavior. In order to match these expectations even more strikingly, we define a simpler sublanguage, called *core-$\pi@$*, where polyadic synchronisation is limited to its simplest form (two names at most are used in a channel name) and priority levels can only be two, and we show its great flexibility in modelling biological systems.

In this paper we show π@ and core-π@ at work by encoding two well-known compartment-based formalisms: BioAmbients and Brane Calculi. Their straightforward embedding in the same language allows to understand clearly their structural/semantical common points and differences and provides their ready-to-run implementation on top of a common platform. For the second language we show both encodings in π@ and core-π@, the former being more easily understandable, but the latter justifying our claim that, for systems biology modelling, core-π@ is enough.

The paper is organised as follows. Section 2 is a gentle introduction to the basic features of the π-calculus, in particular of its syntax and reduction semantics via a structural congruence. It also comprises a non-trivial example of a biological system modelled in π-calculus: the insuline secretion process of a pancreatic β cell in response to a rise of glucose in the blood. We discuss the limitations of this representation, in particular for the difficulties in handling properly the surrogate of compartments that π-calculus is able to express and for the lack of transactional mechanisms. For these reasons, Section 2.3 introduces polyadic synchronisation and Section 2.4 a form of prioritised communication, which constitute the basic ingredients added to π-calculus to obtain the language π@. This language is described in Section 3, with its syntax and reduction semantics. Moreover, the same biological case-study of the insuline secretion process is now modeled in π@, showing that the new representation is much more faithful. Section 3 ends with the presentation of *core*-π@, the minimal subcalculus of π@ that we claim is powerful enough to model complex biological systems. Section 4 is devoted to present the encodings of BioAmbents and Brane Calculi into π@ (and for the second language also in *core*-π@). Finally, Section 5 reports some related work, further research and conclusive remarks.

2 Setting the context

2.1 The π-Calculus

The π-Calculus [17, 18] is a derivation of CCS [19] where parallel processes interact through synchronisation over named channels, with the capability of receiving new channels and subsequently using them for the interaction with other processes, in order to model mobility.

Names constitute the basic entities of the calculus. Each name represents a channel which can be used for synchronisation by parallel processes. For example, the system

$$a(x).P \;\big|\; \overline{a}\langle z\rangle.Q \tag{1}$$

represents two parallel processes $a(x).P$ and $\overline{a}\langle z\rangle.Q$, the first one ready to receive some datum (whose local name is x) over the channel a, the second one ready to send some datum z over the same channel a. The datum z represents in turn another channel, which can be used by the first process for subsequent communications.

If $\overline{a}\langle z\rangle.Q$ sends z to $a(x).P$, then the subsequent behaviour of the two processes is specified by the expressions Q and P respectively. More precisely, we write that the system of Expr. (1) may evolve in the following way:

$$a(x).P \;\big|\; \overline{a}\langle z\rangle.Q \quad \rightarrow \quad P\{z/x\} \;\big|\; Q \tag{2}$$

where $P\{z/x\}$ represents the process P where all the occurrences of the placeholder x have been replaced by z. Here, x is said to be a *bound name*, in opposition to a which is *free*.

The *transition* of the system $a(x).P \;\big|\; \overline{a}\langle z\rangle.Q$ to the system $P\{z/x\} \;\big|\; Q$ is governed by to *reduction relation* "$\rightarrow$", which states that two processes may exchange data if they are ready to perform input/output respectively over the same channel.

The *nondeterministic choice* between two (or more) possible transitions is denoted by the *choice operator "+"*. For example, in the system

$$a(x).P' + b(y).P'' \;\big|\; \overline{a}\langle z\rangle.Q \;\big|\; \overline{b}\langle w\rangle.R$$

the first process may undergo two different, equally possible transitions, caused by a synchronisation with the second process or the third one, respectively. The first transition can be written as

$$a(x).P' + b(y).P'' \;\big|\; \overline{a}\langle z\rangle.Q \;\big|\; \overline{b}\langle w\rangle.R \quad \rightarrow \quad P'\{z/x\} \;\big|\; Q \;\big|\; \overline{b}\langle w\rangle.R$$

while the second as

$$a(x).P' + b(y).P'' \;\big|\; \overline{a}\langle z\rangle.Q \;\big|\; \overline{b}\langle w\rangle.R \quad \rightarrow \quad P''\{w/y\} \;\big|\; \overline{a}\langle z\rangle.Q \;\big|\; R$$

Depending on the occurring transition, the future behaviour of the first process is denoted by $P'\{z/x\}$ or $P''\{w/y\}$ respectively.

Since the order used for enumerating the possible choices is meaningless, i.e. the choice operator is commutative (and associative), we write that

$$a(x).P' + b(y).P'' \quad \equiv \quad b(y).P'' + a(x).P'$$

where "$\equiv$" represents a *congruence relation* between processes that are meant to be characterised by the same behaviour.

Anyway, the choice operator is not the first cause of nondeterminism. As usual for concurrent calculi, the parallelism of the system can produce nondeterministic behaviour. For example, the system

$$a(x).P \;\big|\; \overline{a}\langle z\rangle.Q \;\big|\; \overline{a}\langle w\rangle.R$$

is subjected to two transitions, depending on the process that will actually perform the output operation on channel a:

$$a(x).P \;\big|\; \overline{a}\langle z\rangle.Q \;\big|\; \overline{a}\langle w\rangle.R \quad \rightarrow \quad P\{z/x\} \;\big|\; Q \;\big|\; \overline{a}\langle w\rangle.R$$
$$a(x).P \;\big|\; \overline{a}\langle z\rangle.Q \;\big|\; \overline{a}\langle w\rangle.R \quad \rightarrow \quad P\{w/x\} \;\big|\; \overline{a}\langle z\rangle.Q \;\big|\; R$$

It is possible to prevent unwanted interactions between processes by limiting the scope of a name:

$$(\nu\ a)\big(a(x).P \ \big|\ \overline{a}\langle z\rangle.Q\big) \ \big|\ \overline{a}\langle w\rangle.R \tag{3}$$

Thanks to the restriction operator "ν ", the name a in the first two processes represents a private channel between them. Even if the same name a occurs also in the third process, it constitutes a completely different communication channel. Any renaming of the restricted channels (as well as bound names, by a procedure called alpha-conversion) has no effect on the behaviour of the systems. In fact the following expression

$$(\nu\ b)\big(b(x).P \ \big|\ \overline{b}\langle z\rangle.Q\big) \ \big|\ \overline{a}\langle w\rangle.R$$

is equivalent to Expr. 3, since $b(x).P$ and $\overline{b}\langle z\rangle.Q$ are able to exchange data exactly as before.

In order to model recursive behaviour, an operator of *replication* is introduced in the language. A process P preceded by "!" is thought as being replicated an unlimited number of times. That is

$$!\,P \equiv P \ \big|\ P \ \big|\ \cdots$$

The formal definition of π-Calculus grammar follows.

Definition 1. *Let*

$\mathcal{N}$ be a set of names on a finite alphabet, $x, y, z, \ldots \in \mathcal{N}$;
$\overline{\mathcal{N}} = \{\overline{x} \mid x \in \mathcal{N}\}$

The syntax of π-Calculus is defined in terms of the following grammar:

$$\begin{array}{lcl} P & ::= & \mathbf{0} \ \Big|\ \sum_{i\in I} \pi_i.P_i \ \Big|\ P \mid Q \ \Big|\ !\,P \ \Big|\ (\nu\ x)P \\ \pi & ::= & \tau \ \Big|\ x(y) \ \Big|\ \overline{x}\langle y\rangle \end{array}$$

where

- $\mathbf{0}$ represents the null process;
- $x(y)$ expresses the capability of performing an input on the channel x and receiving a datum which is then bound to the name y;
- $\overline{x}\langle y\rangle$ expresses the capability of sending the name y on the channel x;
- τ is the invisible, uncontrollable action;
- $P \ \big|\ Q$ represents the parallel composition of processes;
- $!\,P$ stands for the unlimited replication of process P;
- $\sum_{i\in I} \pi_i.P_i$ represents the nondeterministic choice between several input/output communication capabilities, denoted also as $\pi_1.P_1 + \pi_2.P_2 + \ldots$;
- $(\nu\ x)P$ represents the scope restriction of the name x to process P.

The full definition of the congruence relation $\equiv$ follows. It depends in turn on the function $\mathrm{fn}(P)$ which returns the set of free names occurring in P.

Definition 2. *The congruence relation $\equiv$ is defined as the least congruence satisfying alpha conversion, the commutative monoidal laws with respect to both ($|$,$\mathbf{0}$) and (+,$\mathbf{0}$) and the following axioms:*

$$\begin{aligned} (\nu\, x)P \mid Q &\equiv (\nu\, x)(P \mid Q) && \text{if } x \notin fn(Q) \\ (\nu\, x)P &\equiv P && \text{if } x \notin fn(P) \\ !\,P &\equiv\, !\,P \mid P \end{aligned}$$

where the function fn *is defined as*

$$\begin{aligned} \mathrm{fn}(\tau) &\stackrel{def}{=} \emptyset & \mathrm{fn}(x(y)) &\stackrel{def}{=} \{x\} \\ \mathrm{fn}(\overline{x}\langle y\rangle) &\stackrel{def}{=} \{x, y\} & \mathrm{fn}(\mathbf{0}) &\stackrel{def}{=} \emptyset \\ \mathrm{fn}(\pi.P) &\stackrel{def}{=} \mathrm{fn}(\pi) \cup \mathrm{fn}(P) & \mathrm{fn}(\textstyle\sum_{i\in I} \pi_i.P_i) &\stackrel{def}{=} \textstyle\bigcup_i \mathrm{fn}(\pi_i.P_i) \\ \mathrm{fn}(P \mid Q) &\stackrel{def}{=} \mathrm{fn}(P) \cup \mathrm{fn}(Q) & \mathrm{fn}(!\,P) &\stackrel{def}{=} \mathrm{fn}(P) \\ \mathrm{fn}((\nu\, x)P) &\stackrel{def}{=} \mathrm{fn}(P) \setminus \{x\} \end{aligned}$$

The relation describing the possible transitions of a process is defined in terms of few simple reduction rules, which exploit the congruence relation previously given.

Definition 3. *π-Calculus semantics is given in terms of the reduction system described by the following rules:*

$$\textit{TAU:}\ \frac{}{\tau.P \;\rightarrow\; P} \qquad \textit{COMM:}\ \frac{}{(\mu(y).P + M) \mid (\overline{\mu}\langle z\rangle.Q + N) \;\rightarrow\; P\{z/y\} \mid Q}$$

$$\textit{PAR:}\ \frac{P \;\rightarrow\; P'}{P \mid Q \;\rightarrow\; P' \mid Q} \qquad \textit{RES:}\ \frac{P \;\rightarrow\; P'}{(\nu\, x)P \;\rightarrow\; (\nu\, x)P'}$$

$$\textit{STRUCT:}\ \frac{P \equiv Q \quad P \;\rightarrow\; P' \quad P' \equiv Q'}{Q \;\rightarrow\; Q'}$$

The TAU rule represents an internal, unobservable change of state of some process P. The transition of Expr. (2) is formalised by rule COMM. The PAR rule describes the meaning of the parallel operator: each process capable of some internal transition, can evolve even when put in parallel with other processes. The RES rule allows transition of processes in presence of restricted names. The key role of restriction (as well as the semantics of the replication) is hidden by rule STRUCT, which states that if two processes are structurally congruent, then they can perform the same transitions. For an extended treatment of the π-Calculus we refer to [17, 18, 20, 21].

The key idea behind the modelling of biological systems by means of the π-Calculus is that biochemical elements can be seen as parallel processes, and their interaction as communication. In particular, each *molecule* of the system can be represented by a process and its *reaction* with other molecules can be modelled as a communication over a fixed channel. For example, the chemical reaction

$$R: \quad R_1 + R_2 \;\rightarrow\; P_1 + P_2$$

where the molecules R_1 and R_2 react according to reaction R, and release P_1 and P_2 as products of the reaction, can be modelled in π-Calculus as

$$R_1 \triangleq r.P_1 \qquad R_2 \triangleq \overline{r}.P_2 \qquad\qquad R_1 \mid R_2 \quad\rightarrow\quad P_1 \mid P_2$$

where each process is named as the corresponding molecule, and reaction R is associated with channel r.

Furthermore, the communication of restricted names between π-Calculus processes can be exploited for the modelling of local bounds between molecules. If M_1 and M_2 represent two molecules ready to bind, the corresponding expression in π-Calculus is

$$M_1 \triangleq (\nu\, b)(\overline{bind}\langle b\rangle.M_1') \qquad M_2 \triangleq bind(x).M_2'$$
$$M_1 \mid M_2 \quad\rightarrow\quad (\nu\, b)(M_1' \mid M_2'\{b/x\}) \qquad\qquad (b \notin \mathrm{fn}(M_2'))$$

where M_1' and $M_2'\{b/x\}$ (and no other process) share the name b after their reaction.

Restriction may be also exploited in order to model *compartments*. A compartment can be thought as a box separating the external environment from its content. A typical biological example is the cell: its external membrane protects from the dispersion of cell material and regulates the exchange of substances. From an external point of view, the content of the compartment is completely hidden by the compartment itself. Hence we may represent in π-Calculus the compartment C as

$$(\nu\, c_1, \ldots, c_k)\big(M_1 \mid \cdots \mid M_l \mid P_1 \mid \cdots \mid P_n)$$

where the compartment C is represented by a set of restricted names $c_1, \ldots, c_k$, the content of C is constituted by $P_1, \ldots, P_n$ under the hypothesis that

$$\mathrm{fn}(P_1 \mid \cdots \mid P_n) \subseteq \{c_1, \ldots, c_k\}$$

i.e. the direct interaction of such elements with some external process is prevented by restricting all their channels. The set of processes $M_1 \mid \cdots \mid M_l$ constitutes an interface of the compartment to the external world, that is, in the case of a cell, the membrane itself (and in particular all the transmembrane channels and proteins). These would constitute the only processes enabled to interact with the outer environment.

2.2 Modelling Biological Systems in π-Calculus

We can now try to model in π-Calculus the simple biological system drawn in Fig. 1. The figure sketches the insuline secretion process of a pancreatic β cell in response to a rise in glucose in the blood. The glucose is transported inside the cell by a transmembrane channel protein, $GLUT2$. Here it undergoes glycolysis, which leads to the production of pyruvate and ATP. The rise in ATP concentration inhibits the action of K^+ channels, which in turn causes a rise

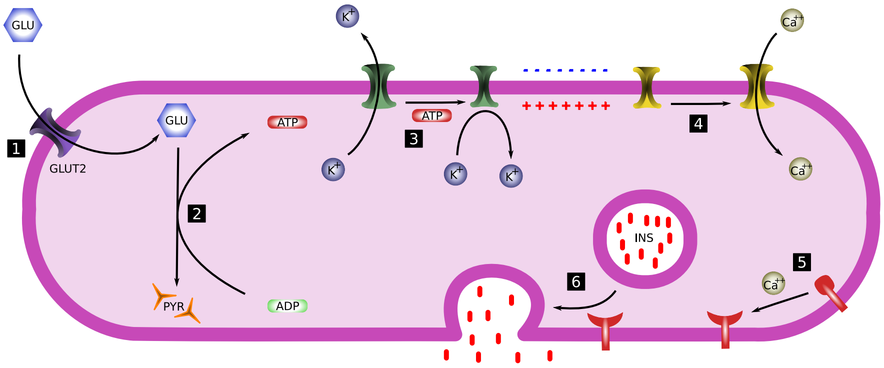

Fig. 1. Insuline secretion in pancreatic β cells in response to a rise in glucose concentration. (1) The rise in glucose concentration of the blood is reflected by a rise in glucose inside the cell, as a consequence of the action of $GLUT2$ glucose transporter. (2) The rise of glucose accelerates the conversion of ADP into ATP, with consequent rise in intracellular ATP concentration. (3) ATP inhibits the action of ATP-sensitive K^+ channels, which reduce the expulsion rate of K^+ ions from the cell. (4) The increasing presence of K^+ depolarises the membrane and triggers the opening of voltage-sensitive Ca^{2+} channels. (5) Fusion proteins are activated by Ca^{2+} ions and (6) trigger the exocytosis of secretory vesicles containing insuline.

in K^+ ions near the membrane and consequently its depolarisation. Voltage-sensitive Ca^{2+} channels are therefore activated and allow the entry of Ca^{2+} ions, which activate the fusion of the insuline-containing vesicles with the cell membrane (exocytosis), with subsequent dumping of insuline molecules into the blood.

The corresponding π-Calculus system will be composed of a compartment C representing the cell, with the shape

$$\begin{aligned}(\nu\, c_1, \dots, c_k)\big(&CHAN_1 \mid \cdots \mid CHAN_l \\ &\mid MOL_1 \mid \cdots \mid MOL_n \\ &\mid VES \mid \cdots \mid VES\big)\end{aligned}$$

where membrane channels $CHAN_i$ are the only processes aware of names external to the cell itself, unlike the molecules MOL_i and the insuline vesicles VES.

The first elements to model are the glucose molecule and the glucose channel. The entry of glucose inside the cell can be modelled as an interaction between the process representing the glucose molecule and that representing the glucose channel. If the glucose molecule is represented by the process GLU, since its interaction with processes external to the cell must be prevented after its entry, it is worth substituting all the channels of GLU with new restricted names. Furthermore, the GLU process must exhibit some sort of recursive behaviour, because it must be able to synchronise at any time with some glucose channel and then be ready again for the other chemical interactions. Such recursive behaviour may be captured by an expression like

$$GLU \triangleq glutrans.GLU + \overline{glureact}.PYR$$

where the channel *glutrans* carries the interaction with the glucose channel, while a synchronisation over *glureact* triggers the glycolysis of GLU with consequent production of pyruvate PYR. The recursive behaviour would be obtained by direct recursion (after the transportation, indicated by the synchronisation over *glutrans*, the process becomes GLU again), but since it is not allowed in the version of the π-Calculus considered here, it must be encoded by combining replication and restriction, in the following way:

$$GLU \triangleq (\nu\ g)\big(\overline{g} \ \big|\ !\ g.(glutrans.\overline{g} + \overline{glureact}.PYR)\big)$$

The internal synchronisation over g allows to spawn a new subprocess

$$(glutrans.\overline{g} + \overline{glureact}.PYR)$$

able to interact nondeterministically either over *glureact* and generate PYR, or synchronise over *glutrans* and then produce another coin $\overline{g}$, thus returning in its initial state. The above formalisation still lacks a significant detail: after the interaction over *glutrans* and the entry in the cell, the process GLU must own a new set of restricted names. Such names must be received by GLU during the interaction with the glucose transporter:

$$\begin{aligned} &GLU(glutrans, glureact) \triangleq \\ &\quad (\nu\ g)\big(\overline{g}\langle glutrans, glureact\rangle \ \big|\ !\ g(t,r).(t(nt,nr).\overline{g}\langle nt,nr\rangle + \overline{r}.PYR)\big) \end{aligned} \tag{4}$$

Now the internal synchronisation over g spawns a new subprocess $GLUSUB$

$$\begin{aligned} &GLU(glutrans, glureact) \quad \rightarrow \\ &(\nu\ g)\big(GLUSUB \ \big|\ !\ g(t,r).(t(nt,nr).\overline{g}\langle nt,nr\rangle + \overline{r}.PYR)\big) \end{aligned}$$

with

$$GLUSUB \triangleq glutrans(nt,nr).\overline{g}\langle nt,nr\rangle + \overline{glureact}.PYR$$

where the placeholders t, r have been replaced by *glutrans*, *glureact* respectively in consequence of the input/output operation. $GLUSUB$ is ready to react over *glureact* or to trigger the simulated movement of GLU inside some other compartment whose respective names for the reaction and transportation of glucose are received as *nr* and *nt* by GLU at the time of the transportation itself.

The glucose transporter $GLUT2$, represented by the process $GLUCHAN$, can be easily modelled as

$$GLUCHAN(gt_{out}, gt_{in}, gr_{in}) \triangleq \quad !\ \overline{gt_{out}}\langle gt_{in}, gr_{in}\rangle$$

where gt_{out} is the channel for glucose transportation *outside the cell*, gt_{in} is the name used for the same purpose but *inside the cell*, and gr_{in} is the channel for the glycolysis of GLU inside the cell. As previously noticed, $GLUCHAN$ is a cross-compartment process, since it must be aware of channels both inside and outside the cell.

The effect of glycolysis is the conversion of ADP into ATP. Even if such process involves several other chemical components, for sake of simplicity it can be easily modelled as direct interaction between the ADP and GLU molecules. For the same reason, the inhibition of K^+ channel proteins by ATP can be modelled as direct interaction between the ATP molecule and the K^+ channel proteins, over some name $inhk$:

$$\begin{aligned} ADP(glureact, inhk) &\triangleq \quad \overline{glureact}.ATP(inhk) \\ ATP(inhk) &\triangleq \quad \overline{inhk} \end{aligned}$$

Since the consequent behaviour of the ATP molecule is not relevant for the present purposes, the corresponding process lacks any recursive formalisation.

In order to model the potassium molecule and channel protein, similar considerations can be applied. In particular, the potassium molecule K may be represented as

$$\begin{aligned} &K(ktrans, kreact) \triangleq \\ &\qquad (\nu\ k)\big(\overline{k}\langle ktrans, kreact\rangle \ \big|\ !\ k(t,r).(t(nt,nr).\overline{k}\langle nt, nr\rangle + \overline{r})\big) \end{aligned}$$

Despite of their similarity, the processes GLU and K differ in the product of their reactions: GLU reduces to PYR, while in the current formalisation K reduces to the null process.

The expression of the K^+ channel protein, $KCHAN$, is a little more complex than the corresponding process $GLUCHAN$. In fact, $KCHAN$ can be inhibited in consequence of ATP binding. In other words, its recursive behaviour shall be interrupted after an interaction over the $inhk$ channel. This effect can be achieved by a slight modification of the recursion used for GLU:

$$\begin{aligned} &KCHAN(kt_{in}, kt_{out}, kr_{out}, inhk) \triangleq \\ &\qquad (\nu\ kc)\big(\overline{kc}\ \big|\ !\ kc.(inhk + \overline{kt_{in}}\langle kt_{out}, kr_{out}\rangle.\overline{kc})\big) \end{aligned}$$

After the interaction over $inhk$, the coin $\overline{k}c$ is not replicated and any further action of the process $KCHAN$ is prevented.

The polarisation of the membrane caused by the rise in K^+ concentration after the inhibition of K^+ channels can be modelled by a fictitious process POL triggered by K^+ ions, which then activates the Ca^{2+} channels:

$$POL(kreact, caact) \triangleq \quad kreact.\overline{caact}$$

The Ca^{2+} channel protein is formalised as the process $CACHAN$, which closely resembles $GLUCHAN$:

$$CACHAN(cat_{out}, cat_{in}, car_{in}, caact) \triangleq \quad caact.!\ \overline{cat_{out}}\langle cat_{in}, car_{in}\rangle$$

The channel is activated only after interaction over *caact*, as required. The calcium process CA is defined exactly as K:

$$\begin{aligned}&CA(catrans, careact) \triangleq \\ &\qquad (\nu\ c)\big(\overline{c}\langle catrans, careact\rangle \;\big|\; !\ c(t,r).(t(nt,nr).\overline{c}\langle nt,nr\rangle + \overline{r})\big)\end{aligned}$$

The stimulation of the exocytosis by the rise in Ca^{2+} concentration can be modelled as mediated by a docking protein $DOCKP$ which is activated by Ca^{2+} ions and trigger the expulsion of the insuline molecules INS contained in the vesicle VES:

$$DOCKP(careact, dockves, ins_{out}) \triangleq careact.\overline{dockves}\langle ins_{out}\rangle$$

Since the vesicle VES (as well as all the INS molecules inside it) is completely embedded inside the cell, the $DOCKP$ process must also communicate to VES the external name(s) ins_{out} which enables the communication of the processes INS with the environment surrounding the cell.

$$\begin{aligned}VES(dockves) \triangleq \quad & (\nu\ ins_{in})\big(dockves(ins_{out}).!\ ins_{in}.INS(ins_{out}) \\ & \qquad\qquad \big|\ INS(ins_{in}) \ \big|\ \cdots \ \big|\ INS(ins_{in})\big)\end{aligned}$$

The process VES represents another compartment, where all the names of the embedded processes (INS) are restricted, while only one cross-compartment process filters their interaction with the external environment. In this very simple model, the INS processes are formalised as

$$INS(ins) \triangleq \quad \overline{ins}$$

Finally, the whole system can be summarised as

$$\begin{aligned}SYS \triangleq \quad & GLU(gt_{out}, gr_{out}) \ \big|\ K(kt_{out}, kr_{out}) \ \big|\ Ca(cat_{out}, car_{out}) \ \big| \\ & (\nu\ gt_{in}, gr_{in}, kt_{in}, kr_{in}, cat_{in}, car_{in}, inhk, caact, dockves, ins_{in}) \\ & \Big(GLU(gt_{in}, gr_{in}) \ \big|\ GLUCHAN(gt_{out}, gt_{in}, gr_{in}) \\ & \ \big|\ K(kt_{in}, kr_{in}) \ \big|\ KCHAN(kt_{out}, kt_{in}, kr_{in}, inhk) \ \big| \\ & \ \big|\ ADP(gr_{in}, inhk) \ \big|\ ATP(inhk) \ \big|\ POL(kr_{in}, caact) \\ & \ \big|\ Ca(cat_{in}, car_{in}) \ \big|\ CACHAN(cat_{out}, cat_{in}, car_{in}, caact) \\ & \ \big|\ DOCKP(car_{in}, dockves, ins_{out}) \ \big|\ VES(dockves)\Big)\end{aligned}$$

This modelling approach in π-Calculus allows to express somehow the idea of compartments, but reveals several drawbacks.

The most evident is the need to encode compartment scoping by restricting *all* the free names of the enclosed processes. As even worse consequence we have that cross-compartment processes must handle all the names of the interacting processes, taking care of the correspondence between distinct names representing the same channel inside different compartments. The model becomes even more complicated when such processes need to create new restricted names and communicate them externally. If we then require compartment operations which affect their nesting structure (merging, splitting, creation, movement of whole compartments) the above approach becomes practically unfeasible, both for the difficulties in name handling and the impossibility of ensuring atomicity of such complex operations without a purely centralised implementation.

Such issues can be overcome by the simple extensions that are presented in the next subsections.

2.3 Polyadic Synchronisation

In the π-Calculus, channels and transmitted names are usually synonyms. Polyadic synchronisation [16] consists in giving *structure* to channels: each channel is composed of one or more names and identified by all of them in relation to the exact sequence of their occurrence. For example, an email address is usually written in the form *username@domain*, where *username* and *domain* are two strings – two names – both necessary to identify the given email address. Moreover, their order is crucial since *domain@username* specifies another, likely nonexistent, address. Similarly, polyadic synchronisation (in its simplest form) provides the capability of writing channels as $name_1@name_2$. In other words, a channel is indicated by a vector of two names $(name_1, name_2)$ and communication between two processes may happen only if they are pursuing a synchronisation along channels denoted by the same names.

Apart from this, communication happens in the same way as in the π-Calculus. For example, the transition

$$\overline{polyadic@comm}\langle d\rangle.P \mid polyadic@comm(x).Q \quad \rightarrow \quad P \mid Q\{d/x\}$$

produces the same renaming effect of a π-Calculus transition, but with one difference: in the π-Calculus, the transmission of a name always stands for the transmission of a channel, while in the above example the transmitted name constitutes only one component of it.

An extended form of polyadic synchronisation allows for the use of more than two names for each channels, like in the following example:

$$\overline{@c_1@c_2@c_3}\langle d\rangle.P \mid c_1@c_2@c_3(x).Q \quad \rightarrow \quad P \mid Q\{d/x\}$$

In general, there is no limit to the length of the vector of names representing a channel.

For concision and readability, polyadic synchronisation is often used also in conjunction with polyadic communication:

$$\overline{polyadic@comm}\langle a,b,c\rangle.P \mid polyadic@comm(x,y,z).Q \quad \rightarrow$$
$$P \mid Q\{a/x, b/y, c/z\}$$

The benefits in terms of simplicity of biological modellings are immediate in presence of compartments. For example, if the π-calculus process

$$M(m_1,\dots,m_n) \triangleq \pi.M' + \dots$$

is a molecule subjected to movement across the cellular membrane (like the GLU process of Expr. (4)), all of its channels must be restricted after its conveyance into the cell:

$$M(m_1^{out}, \ldots, m_n^{out}) \mid (\nu\, m_1^{in}, \ldots, m_n^{in}, \ldots)(M(m_1^{in}, \ldots, m_n^{in}) \mid \cdots)$$

In particular, any channel protein able to convey the above molecule across different compartments should be aware of all the names $m_1^{out}, \ldots, m_n^{out}, m_1^{in}, \ldots, m_n^{in}$. Polyadic synchronisation allows to model the compartment just as *one* restricted name. For example, the above system may be converted as

$$M(m_1, \ldots, m_n, c_{out}) \mid (\nu\, c_{in}, \ldots)(M(m_1, \ldots, m_n, c_{in}) \mid \cdots)$$

where each input or output action $\pi.M'$ inside M has been encoded as $\pi@c.M'$, and c represents the compartment M lies in (c_{out} or c_{in} in the above example). In this way, only one restricted name is needed to formalise a new compartment, regardless of all the channels the enclosed processes may use. This substantial simplification affects also the formalisation of cross-compartment processes: they need to handle only one additional name for each compartment they partially reside in.

2.4 Priority

The idea behind the notion of priority applied here to the π-Calculus [22] is very similar to the mechanisms adopted for the implementation of schedulers which allow to give processes several levels of priority, in dependence of their requirements (responsiveness, cpu load, real-time constraints, etc) for the task they accomplish.

In this paper we consider a particular kind of priority characterised by global, immediate preemption: each process denoted by high priority holds the central processing unit and executes its job before any low priority process may perform some other task. In π-Calculus setting, this is equivalent to force high priority synchronisations or communications to happen before any low priority action. A high priority action is indicated by underlining the name of the channel. For example, the expression

$$\overline{l}\langle a\rangle.P \mid \overline{\underline{h}}\langle b\rangle.Q$$

contains two processes with different, increasing priority. In the above situation, both are blocked: in fact, no other process is ready to receive over the channel l or h. In presence of some process listening on channel l, the first process may react in the following way:

$$\overline{l}\langle a\rangle.P \mid l(x).P' \mid \overline{\underline{h}}\langle b\rangle.Q \quad \rightarrow \quad P \mid P'\{a/x\} \mid \overline{\underline{h}}\langle b\rangle.Q$$

In fact, this would be the only possible transition since no other reduction is available. Conversely, when both high and low priority actions are enabled, low-priority synchronisations can occur only after high-priority ones:

$$\overline{l}\langle w\rangle \mid l(x).P \mid \overline{\underline{h}}\langle y\rangle \mid \underline{h}(z).Q \quad \not\rightarrow \quad \mathbf{0} \mid P\{w/x\} \mid \overline{\underline{h}}\langle y\rangle \mid \underline{h}(z).Q$$

$$\overline{l}\langle w\rangle \mid l(x).P \mid \overline{\underline{h}}\langle y\rangle \mid \underline{h}(z).Q \quad \rightarrow \quad \overline{l}\langle w\rangle \mid l(x).P \mid \mathbf{0} \mid Q\{y/z\} \quad \rightarrow \quad \mathbf{0} \mid P\{w/x\} \mid \mathbf{0} \mid Q\{y/z\}$$

The first of the two transitions is not allowed because interactions on low-priority channel l may happen only after the high-priority communication on channel $\underline{h}$.

For sake of clarity, communications between channels with the same name but different priorities are forbidden. For example, in the following system there are no possible reductions:

$$\overline{\underline{c}}\langle a\rangle.P \mid c(x).Q$$

An additional level of priority can be denoted by double underlining a channel name:

$$\overline{l}\langle a\rangle.P \mid \overline{\underline{h}}\langle b\rangle.Q \mid \overline{\underline{\underline{u}}}\langle c\rangle.R$$

If any reduction is available over the $\underline{\underline{u}}$ channel, it preempts any reduction over $\underline{h}$ and l. Additional levels of priority require the introduction of integer numbers as labels added to each input/output operation. For this reason the above expression may be also written as follows:

$$\overline{2\!:\!l}\langle a\rangle.P \mid \overline{1\!:\!h}\langle b\rangle.Q \mid \overline{0\!:\!u}\langle c\rangle.R$$

where 0 denotes the highest level of priority. The choice of identifying higher priority levels with decreasing integer numbers is arbitrary and does not affect the semantics or expressiveness of the language.

Since three levels of priority suffice for the purposes of this paper, only the first of these two syntaxes will be used in the encodings and modelling examples, for its conciseness and readability.

For a detailed survey of priority in process algebras, we refer to [22].

At first sight, the idea of priority seems foreign and unnecessary for biological modelling purposes. Anyway, the complexity of this realm makes almost impossible the design of a suitable and complete modelling language. The presence of several priority levels for operations allows to compose high-level, complex operations as sequences of several simple, low-level steps (transitions) by avoiding any interference of external processes with the involved elements. Such low-level transitions can be composed in different ways, depending on the abstraction adopted to formalise the system of interest. This effect can be achieved by encoding each high-level operation as a list of high priority actions, preceded by a single low priority operation which act as a *guard*. For example, if the processes P_1, P_2, P_3, P_4 represents four proteins which can bind together and form a new complex C, they can be modelled in π-Calculus by exploiting the restriction operator and representing the binding as the sharing of a private name:

$$(\nu\ b)P_1 \mid P_2 \mid P_3 \mid P_4 \quad \rightarrow \quad \cdots \quad \rightarrow \quad (\nu\ b)\big(P_1' \mid P_2' \mid P_3' \mid P_4'\big)$$

Since more than one transition is needed to accomplish the whole process, if one of the requirements is its atomicity (this is often necessary when the modelled processes may meanwhile interact with other elements and give rise to undesirable situations not pertaining to the original model) then there is no way to obtain a satisfactory modelling in π-Calculus. Priority instead allows to ensure that no other process can interfere during the formation of the above complex. The four molecules can be formalised as

$$P_1 \triangleq \bar{l}\langle b\rangle.\overline{\underline{h_1}}\langle b\rangle.\overline{\underline{h_2}}\langle b\rangle.P_1' \qquad\qquad P_2 \triangleq l(x).P_2'$$
$$P_3 \triangleq \underline{h_1}(x).P_3' \qquad\qquad P_4 \triangleq \underline{h_2}(x).P_4'$$

so that after the first synchronisation of P_1 and P_2 over the low priority channel l, a sequence of (two) high priority communications between P_1 and P_3, P_4 is triggered and cannot be interrupted by any other process. In fact, the possibility of performing the initial low priority action over l guarantees that any other high priority operation occasionally available in the system has been previously consumed.

3 Description of the formalism

3.1 The π@ language

The π@ language joins the expressiveness of polyadic communication and priority in order to model both localisation of processes inside compartments and atomicity of complex operations that require more than one reduction step for their completion. Thanks to the simplicity of such extensions, π@ is very close to the π-Calculus: from a syntactical point of view the only difference is the structure of channels, composed of multiple names and tagged by the priority of the action. We use μ to denote a vector of names $x_1, \ldots, x_n$ and $k : \mu$ to denote a channel, that is a natural number k specifying the priority level followed by a vector of names μ. In particular, $\overline{k : \mu}$ represents an output operation along channel $k : \mu$, while $k : \alpha$ stands for a generic input, output or silent action τ of priority level k.

Definition 4. *Let*

$\mathcal{N}$ *be a set of names on finite alphabet,* $x, y, z, \ldots \in \mathcal{N}$ *;*
$\mathcal{N}^+ = \bigcup_{i>0} \mathcal{N}^i \;, \qquad \mu \in \mathcal{N}^+$ *;*
$\overline{\mathcal{N}}^+ = \{\overline{\mu} \mid \mu \in \mathcal{N}^+\}$ *;*
$\alpha \in \left(\overline{\mathcal{N}}^+ \cup \mathcal{N}^+ \cup \{\tau\} \right)$ *;*

The syntax of π@ *defined in terms of the following grammar:*

$$P \;::=\; \mathbf{0} \;\Big|\; \sum_{i\in I} \pi_i.P_i \;\Big|\; P \mid Q \;\Big|\; !\,P \;\Big|\; (\nu\, x)P$$
$$\pi \;::=\; k\colon\tau \;\Big|\; k\colon\mu(x) \;\Big|\; \overline{k\colon\mu}\langle x\rangle$$

As previously introduced, the following abbreviations are used for readability:

$$
\begin{array}{rclcrcl}
\mu(x) & = & 2\!:\!\mu(x) & \qquad & \overline{\mu}\langle x\rangle & = & \overline{2\!:\!\mu}\langle x\rangle \\
\underline{\mu}(x) & = & 1\!:\!\mu(x) & \qquad & \underline{\overline{\mu}}\langle x\rangle & = & \overline{1\!:\!\mu}\langle x\rangle \\
\underline{\underline{\mu}}(x) & = & 0\!:\!\mu(x) & \qquad & \underline{\underline{\overline{\mu}}}\langle x\rangle & = & \overline{0\!:\!\mu}\langle x\rangle
\end{array}
$$

The definition for structural congruence $\equiv$ is exactly the same as given for π-Calculus, where the function fn is naturally extended to the $\pi @$ syntax, that is

$$
\begin{aligned}
\mathrm{fn}(k : \mu(y)) &\stackrel{def}{=} \{\mu_1, \ldots, \mu_n\} \\
\mathrm{fn}(\overline{k : \mu}\langle y\rangle) &\stackrel{def}{=} \{\mu_1, \ldots, \mu_n, y\}
\end{aligned}
$$

where $\mu = \mu_1 @ \cdots @ \mu_n$. The reduction semantics is very similar, but defined in terms of an auxiliary function $I^k(P)$, representing the set of actions of priority k which the process P may immediately execute. For example, if

$$P = a.Q \;\big|\; \underline{b} \;\big|\; \underline{\underline{\overline{c}}}.R \;\big|\; \underline{\overline{d}} + \underline{\underline{e}}.S \;\big|\; \overline{a}.T$$

then $I^0(P) = \{\overline{c}, e\}$, $I^1(P) = \{b, \overline{d}\}$, $I^2(P) = \{a, \overline{a}, \tau\}$, where the availability of τ action derives from the interaction of the first and last process.

Definition 5. *Let $I^k(P)$ be*

$$
\begin{aligned}
I^k\big(\sum_i l_i\!:\!\alpha_i.P_i\big) &= \{\alpha_i \mid l_i = k\}; \\
I^k\big((\nu\; y)\; P\big) &= I^k(P) \setminus \big\{\alpha \mid y \in \{x_1, \ldots, x_n\} \wedge \\
&\qquad (\alpha = x_1 @ \ldots @ x_n \;\vee\; \alpha = \overline{x_1 @ \ldots @ x_n})\big\}; \\
I^k\big(!P\big) &= I^k(P \mid P); \\
I^k\big(P \mid Q\big) &= I^k(P) \cup I^k(Q) \cup \{\tau \mid I^k(P) \cap \overline{I^k(Q)} \neq \emptyset\}, \\
&\quad \overline{I^k(Q)} = \big\{\overline{\alpha} \mid \alpha \in I^k(Q)\big\}
\end{aligned}
$$

$\pi @$ semantics is given in terms of the following reduction system:

$$
\frac{\tau \notin \bigcup_{i<k} I^i(M)}{k\!:\!\tau.P + M \;\rightarrow_k\; P}
\qquad
\frac{P \;\rightarrow_k\; P'}{(\nu\; x)P \;\rightarrow_k\; (\nu\; x)P'}
$$

$$
\frac{\tau \notin \bigcup_{i<k} I^i(M \mid N)}{(k\!:\!\mu(y).P + M) \mid (\overline{k\!:\!\mu}\langle z\rangle.Q + N) \;\rightarrow_k\; P\{z/y\} \mid Q}
$$

$$
\frac{P \;\rightarrow_k\; P' \qquad \tau \notin \bigcup_{i<k} I^i(P \mid Q)}{P \mid Q \;\rightarrow_k\; P' \mid Q}
\qquad
\frac{P \equiv Q \quad P \;\rightarrow_k\; P' \quad P' \equiv Q'}{Q \;\rightarrow_k\; Q'}
$$

π@ reduction rules are exactly the same of π-Calculus, except for the additional condition $\tau \notin \bigcup_{i<k} I^i(\ldots)$ which avoids the execution of low priority actions if higher priority communications (represented by τ actions) are immediately available.

3.2 Modelling the insuline example in π@

The benefits deriving from polyadic synchronisation and priority for biological modellings can be noticed as soon as even very simple systems are formalised, like the one in Fig. 1 already described into the π-Calculus. We have previously discussed why a faithful description of a compartment in π-Calculus implies that all the names of the enclosed processes are restricted, in order to prevent their interaction with the external environment. On the contrary, π@ allows to characterise each compartment by means of a single restricted name, as shown in Sect. 2.3. Therefore, the shape of the system of Fig. 1 becomes

$$\begin{aligned}(\nu\ c_{in})\big(&CHAN_1 \ \big|\ \cdots \ \big|\ CHAN_l\\ &\big|\ MOL_1 \ \big|\ \cdots \ \big|\ MOL_n\\ &\big|\ VES \ \big|\ \cdots \ \big|\ VES\big)\end{aligned}$$

where the cell is represented by the restricted name c_{in}, the cross-compartment processes $CHAN_1, \ldots, CHAN_l$ communicate to the external environment by a corresponding name c_{out}, and the set of free names of the processes MOL_1, ..., MOL_n, VES is irrelevant for their confinement inside the compartment, as soon as c_{out} does not appear in such set and c_{in} constitutes one of the names which identify each channel.

Therefore, the GLU process of Expr. (4) may be translated as follows:

$$\begin{aligned}&GLU(comp, gt, gr) \triangleq (\nu\ g)\\ &\qquad\big(\overline{g}\langle comp\rangle \ \big|\ \ !\ g(c).(gt@c(c_{new}).\overline{g}\langle c_{new}\rangle + \overline{gr@c}.PYR)\big)\end{aligned} \tag{5}$$

The behaviour of the process is almost the same: replication is exploited for encoding the recursive behaviour of the molecule, which is always ready either to react or to be moved into another compartment. These two capabilities are formalised by an interaction over $gr@c$ and $gt@c$ respectively, where c represents the name of the compartment the molecule lies in. The movement of the molecule to another compartment is reflected by receiving a new compartment name and forgetting the previous one. Since such name appears in all the channels that allow the molecule to interact with the other processes, GLU can interact only with elements enclosed in its same compartment. The simplification of Expr. (4) w.r.t. Expr. (5) is minimal for the reason that the description of the GLU molecule includes only two public channels, but it becomes significant as soon as their number grows.

It is worth remarking that the current semantics of GLU may not correspond to the intended behaviour of the glucose molecule in the original biological model:

if it requires the movement of the molecule across compartments to be instantaneous, then Expr. (5) does not constitute a correct formalisation because of the additional internal step needed to spawn the molecule inside the target compartment. Priority addresses exactly this very common modelling problem: it is possible to use high priority reductions to model all the operations that in the original model are purely atomic or not present at all:

$$\begin{aligned} &GLU(comp, gt, gr) \triangleq (\nu\ g) \\ &\qquad \big(\underline{\overline{g}}\langle comp\rangle \ \big|\ \ !\ \underline{g}(c).(gt@c(c_{new}).\underline{\overline{g}}\langle c_{new}\rangle + \overline{gr@c}.PYR)\big) \end{aligned} \tag{6}$$

The corresponding expression of the $GLUT2$ glucose transporter can be slightly simplified as well:

$$GLUCHAN(c_{out}, c_{in}, gt) \triangleq \ \ !\ \overline{gt@c_{out}}\langle c_{in}\rangle$$

Now, the channel process does not need to know all the names of the transported molecule. This is a very valuable property, since it means that there is no need to change the expression of the glucose transporter each time the formalisation of GLU changes, for example after the addition of new reaction capabilities. As previously discussed, $GLUCHAN$ is characterised by the presence of two compartment names, for the reason that it represents a cross-compartment molecule.

The processes representing ADP and ATP are almost unchanged:

$$\begin{aligned} ADP(glureact, inhk, c) &\triangleq \ \ \overline{glureact@c}.ATP(inhk, c) \\ ATP(inhk, c) &\triangleq \ \ \overline{inhk@c} \end{aligned}$$

In this case the use of polyadic synchronisation seems to make more complicated the expression of the above molecules, but the benefit will appear after the formalisation of the whole cell.

For the description of the potassium molecule K the same principles leading to Expr. (6) can be applied:

$$\begin{aligned} &K(kt, kr, comp) \triangleq \\ &\qquad (\nu\ k)\big(\underline{\overline{k}}\langle comp\rangle \ \big|\ !\ \underline{k}(c).(kt@c(c_{new}).\underline{\overline{k}}\langle c_{new}\rangle + \overline{kr@c})\big) \end{aligned}$$

Even in this case the high priority of internal reductions can provide a more faithful modelling, like for the potassium channel $KCHAN$:

$$\begin{aligned} &KCHAN(c_{in}, c_{out}, inhk) \triangleq \\ &\qquad (\nu\ kc)\big(\underline{\overline{kc}} \ \big|\ !\ \underline{kc}.(inhk + \overline{kt@c_{in}}\langle c_{out}\rangle.\underline{\overline{kc}})\big) \end{aligned}$$

The remaining processes can be translated in the same way:

$$
\begin{aligned}
POL(c, kreact, caact) &\triangleq \quad kreact@c.\overline{caact@c}\\
CACHAN(c_{out}, c_{in}, cat, caact) &\triangleq \quad caact@c_{in}.!\ \overline{cat@c_{out}}\langle c_{in}\rangle\\
CA(comp, cat, car) &\triangleq \quad (\nu\ ca)\big(\underline{\overline{ca}}\langle comp\rangle \ \big|\\
&\qquad !\ \underline{ca}(c).(cat@c(c_{new}).\underline{\overline{ca}}\langle c_{new}\rangle + \overline{car@c})\big)\\
DOCKP(c_{in}, car, dockves, c_{out}) &\triangleq \quad car@c_{in}.\overline{dockves@c_{in}}\langle c_{out}\rangle\\
INS(c, ins) &\triangleq \quad \overline{ins@c}
\end{aligned}
$$

Since the vesicle VES constitutes another, nested compartment inside the cell, it can be as well represented by a new restricted name:

$$
\begin{aligned}
VES(c_{out}, dv, ins) \triangleq \quad & (\nu\ vc_{in})\big(dv@c_{out}(c_{ext}).!\ ins@vc_{in}.INS(c_{ext}, ins)\\
& \qquad \big|\ INS(vc_{in}, ins)\ \big|\ \cdots\ \big|\ INS(vc_{in}, ins)\big)
\end{aligned}
$$

The internal compartment name for VES is vc_{in}, while c_{out} stands for the name corresponding to the cell compartment. c_{ext} will be bound to the name of the compartment surrounding the cell, where the insuline molecules will be dumped after the exocytosis of the vesicle VES.

If the process of exocytosis is abstracted as a single, atomic operation, then priority can be used for the loop of the process expelling all the insuline molecules. It may be modified as follows:

$$dv@c_{out}(c_{ext}).!\ \underline{ins@vc_{in}}.INS(c_{ext}, ins)$$

and the INS molecules accordingly:

$$INS(c, ins) \triangleq \quad \underline{\overline{ins@c}}$$

In this way any sequence of movement of INS molecules would appear as atomic. The presence of the high priority action immediately after the replication constitutes a very dangerous operation: a high priority loop of this kind may block the whole system if there is no guaranty on its termination. In the case of the VES process the finite number of INS molecules guarantees such termination.

Finally, the whole system is represented as follows:

$$
\begin{aligned}
SYS \triangleq \quad & GLU(c_{out}, gt, gr)\ \big|\ K(c_{out}, kt, kr)\ \big|\ Ca(c_{out}, cat, car)\ \big|\\
& (\nu\ c_{in})\\
& \Big(GLU(c_{in}, gt, gr)\ \big|\ GLUCHAN(c_{out}, c_{in}, gt)\\
& \ \big|\ K(kt, kr, c_{in})\ \big|\ KCHAN(c_{in}, c_{out}, inhk)\ \big|\\
& \ \big|\ ADP(gr, inhk, c_{in})\ \big|\ ATP(inhk, c_{in})\ \big|\ POL(c_{in}, kr, caact)\\
& \ \big|\ Ca(c_{in}, cat, car)\ \big|\ CACHAN(c_{out}, c_{in}, cat, caact)\\
& \ \big|\ DOCKP(c_{in}, car, dockves, c_{out})\ \big|\ VES(c_{out}, dockves, ins)\Big)
\end{aligned}
$$

Like the VES compartment, the β cell is denoted by only one restricted name, c_{in}.

3.3 The core-π@ language

The π@ language is characterised by the capability of using an unbounded number of names for channels and of priority levels for reductions. The writing of complex encoding like those presented in Sect. 4.5 and 4.8 can be significantly simplified by such flexibility. Anyway, two levels of priority and two names for each channel are sufficient for most of the purposes, like the modelling examples previously shown. For this reason we introduce a subcalculus of π@, denoted as *core-π@*, characterised by only two levels of priority (i.e. normal actions and prioriotised actions) and two names for each channel. A valuable property of this core-π@ is its straighforward mapping into its stochastic counterpart Sπ@ [23, 24].

We introduce three distinct sets of names $\mathcal{N}, \mathcal{P}, \mathcal{C}$ denoting respectively unprioritised actions, prioritised actions and compartments. Each channel $x@a$ is denoted by an action name and a compartment name. In order to keep notation simple, compartment names may be omitted when superfluous.

Definition 6. *Let $\mathcal{N}, \mathcal{P}, \mathcal{C}$ be distinct sets of names on finite alphabet, with m, n ranging over $\mathcal{N}$, p, q over $\mathcal{P}$, a, b over $\mathcal{C}$ and x, y over $\mathcal{X} = \mathcal{N} \cup \mathcal{P} \cup \mathcal{C}$. The syntax of the core-π@ language is defined as*

$$P \;::=\; \mathbf{0} \;\Big|\; \sum_{i \in I} \pi_i.P_i \;\Big|\; P \mid Q \;\Big|\; !\,\pi.P \;\Big|\; (\nu\, x)P$$

$$\pi \;::=\; \tau \;\Big|\; n@a(\mathbf{x}) \;\Big|\; \overline{n}@a\langle \mathbf{x} \rangle \;\Big|\; \underline{\tau} \;\Big|\; \underline{p}@a(\mathbf{x}) \;\Big|\; \underline{\overline{p}}@a\langle \mathbf{x} \rangle$$

where $\mathbf{x}$ represents one or more names $x_1, \ldots, x_i$ ranging over $\mathcal{X}$.

Like for π@, the semantics of this core-π@ is given by means of a reduction system based on the following congruence relation.

Definition 7. *The congruence relation $\equiv$ is defined as the least congruence satisfying alpha conversion, the commutative monoidal laws with respect to both $(\,\mid\,, \mathbf{0})$ and $(+, \mathbf{0})$ and the following axioms:*

$$\begin{aligned}
(\nu\, x)P \mid Q &\equiv (\nu\, x)(P \mid Q) && \textit{if } x \notin \mathrm{fn}(Q)\\
(\nu\, x)P &\equiv P && \textit{if } x \notin \mathrm{fn}(P)\\
!\,\pi.P &\equiv \pi.(!\,\pi.P \mid P)
\end{aligned}$$

where the function fn *is defined as*

$$\begin{aligned}
\mathrm{fn}(\tau) &\triangleq \emptyset & \mathrm{fn}(\underline{\tau}) &\triangleq \emptyset\\
\mathrm{fn}(n@a(\mathbf{x})) &\triangleq \{n, a\} & \mathrm{fn}(\overline{n}@a\langle\mathbf{x}\rangle) &\triangleq \{n, a, \mathbf{x}\}\\
\mathrm{fn}(\underline{p}@a(\mathbf{x})) &\triangleq \{p, a\} & \mathrm{fn}(\underline{\overline{p}}@a\langle\mathbf{x}\rangle) &\triangleq \{p, a, \mathbf{x}\}\\
\mathrm{fn}(\mathbf{0}) &\triangleq \emptyset & \mathrm{fn}((\nu\, x)P) &\triangleq \mathrm{fn}(P) \setminus \{x\}\\
\mathrm{fn}(\pi.P) &\triangleq \mathrm{fn}(\pi) \cup \mathrm{fn}(P) & \mathrm{fn}(\sum_{i\in I} \pi_i.P_i) &\triangleq \bigcup_i \mathrm{fn}(\pi_i.P_i)\\
\mathrm{fn}(P \mid Q) &\triangleq \mathrm{fn}(P) \cup \mathrm{fn}(Q) & \mathrm{fn}(!\, \pi.P) &\triangleq \mathrm{fn}(\pi.P)
\end{aligned}$$

Definition 8. *Core-π@ semantics is given in terms of the following reduction system:*

$$\frac{}{\underline{\tau}.P + M \twoheadrightarrow P} \quad \frac{M \not\twoheadrightarrow M'}{\tau.P + M \mapsto P} \quad \frac{P \twoheadrightarrow P'}{(\nu\, x)P \twoheadrightarrow (\nu\, x)P'} \quad \frac{P \mapsto P'}{(\nu\, x)P \mapsto (\nu\, x)P'}$$

$$\frac{}{(\underline{p}@a(\mathbf{x}).P + M) \mid (\underline{\overline{p}}@a\langle\mathbf{y}\rangle.Q + N) \twoheadrightarrow P\{\mathbf{y}/\mathbf{x}\} \mid Q}$$

$$\frac{M \mid N \not\twoheadrightarrow R}{(n@a(\mathbf{x}).P + M) \mid (\overline{n}@a\langle\mathbf{y}\rangle.Q + N) \mapsto P\{\mathbf{y}/\mathbf{x}\} \mid Q}$$

$$\frac{P \twoheadrightarrow P'}{P \mid Q \twoheadrightarrow P' \mid Q} \quad \frac{P \mapsto P' \qquad P \mid Q \not\twoheadrightarrow R}{P \mid Q \mapsto P' \mid Q}$$

$$\frac{P \equiv Q \quad P \twoheadrightarrow P' \quad P' \equiv Q'}{Q \twoheadrightarrow Q'} \quad \frac{P \equiv Q \quad P \mapsto P' \quad P' \equiv Q'}{Q \mapsto Q'}$$

The presence of only one additional level of priority allows to avoid the definition of the $I^k()$ function of Def. 5. On the other hand, the definition of the reduction relation requires two rules for each corresponding rule of π@ semantics.

4 Application examples

The key feature which differentiates many recent bio-inspired calculi from the π-Calculus is the explicit formalisation of compartments. BioAmbients is a modified version of the Ambient calculus [25, 26], where compartments are represented by *ambients*, a sort of boxes containing processes or other nested boxes. In Brane compartments are bounded by membranes, on the surface of which processes compute. Both ambients and membranes are organised in a tree structure, both can dinamically modify this structure by performing for example *merge*, *enter/exit* or *exo* operations. The central issue is *how* they modify this structure: the most observable difference is the bitonality preserved by brane semantics and totally absent in BioAmbients, which corresponds to the preservation of the parity of the nesting level of processes. As remarked in [9], this peculiarity is enough to preclude an immediate embedding of one language into the other.

Consequently, on the one hand they gain faithfulness because of their additional primitives designed to model the addressed biological phenomena, on the other their specialisation does not allow to mutually translate the models expressed in each language. Furthermore, the high abstraction level of such primitives hides the mechanisms underlying the idea of compartment, whose unfolding can reveal their strong resemblance.

π@ features were chosen to overcome all these issues: the lack of a predefined semantics for compartments together with the possibility of expressing localisation by means of polyadic synchronisation and complex atomic operations by means of priority place π@ one abstraction level underneath, as a sort of *assembly* language for compartmentalised formalisms. As previously discussed, it allows to simplify consistently w.r.t. the π-Calculus the formalisation of biological models which embed the notion of compartment. In this section we show that π@ is also able to supply the same high level features offered by bio-oriented languages like BioAmbients and Brane calculi. In particular, we show how both of them can be encoded directly into π@, by unfolding the basic functioning of compartment semantics and providing a common platform for their direct comparison and implementation.

4.1 Basic ideas

Compartment and their nesting are very intuitive abstractions: the simple statement that an object is enclosed in a box suggests that it is someway isolated from the external context; putting one box into another means that, after the operation, the inner box *with all its content* are located inside the outer one; merging the content of two boxes implies putting in the same box *all the enclosed objects*. To obtain this behaviour in π@ we must recognise the exact meaning of every operation on compartments and reproduce step by step the same semantics.

The first concept to unfold is nesting: compartments compose a dynamical tree structure which must be encoded in π@. As suggested in [21], these kind of structures can be represented as a set of processes linked by the share of private channels between parent and child nodes. The encoding of the insuline secretion process is a simple example of this situation: the cell and the insuline vesicles define the boundaries of compartments linked by the presence of cross-compartment processes (like cross-membrane molecular channels) which are able to interact with elements located in both compartments. Like in [2], the scoping of private names represents the boundaries of such compartments, but thanks to polyadic synchronisation each private name may represent an unlimited number of private communication channels, as discussed during the modelling of the insuline example. If each node is supplied with one distinctive name, the simplest way to encode the tree is by ensuring that each node knows the name identifying its parent compartment.

Therefore, trivial changes in the tree structure may affect an unlimited number of processes: the simple disclosure of a compartment implies that all contained processes must be notified of their new parent compartment name. The same situation occurs when splitting or merging the content of two compartments, like in BioAmbients *merge*+ /*merge*– and Brane $exo/exo^{\perp}$ operations. In π@ this turns out to be a sort of *multicast* communication, where specific groups of nodes – that is sibling and child processes – must receive on the proper channel a new compartment name. This result is achieved by a smart use of priority levels: a high priority loop notifies in turn all the interested processes and ends when such processes do not exist anymore. By a single line of code, we obtain in π@ the same mechanism typical of broadcast communication:

$$BCAST \quad \triangleq \quad !\, \underline{\underline{bcast}}(x,y).(\underline{\tau} + \underline{\underline{\overline{x}}}\langle y\rangle.\underline{\underline{\overline{bcast}}}\langle x,y\rangle)$$

The above process can be triggered by an output operation $\underline{\underline{\overline{bcast}}}\langle chn, newchn\rangle$ and terminate when no high priority synchronisations are available, leaving no residual terms. Obviously, a high priority complementary output loop $!\ \underline{\underline{\overline{bcast}}}\langle chn, newchn\rangle$ would cause the system to hang, since it prevents any other computation with normal priority. The avoidance of such high priority and non-terminating loops is often not trivial. In particular, an homomorphic translation of the replication operator would immediately cause them to appear, as we will discuss in the following encodings. This is one of the most difficult translation issues and will force us to represent *indirectly* the encoded replicated processes, by keeping explicit track of each replicated instance in our encoding functions.

4.2 Requirements

The fundamental criterion guiding any encoding is the preservation of some addressed semantics. According to [27], this often means that the encoding function $[\![\cdot]\!]$ must at least fulfill the notion of *operational correspondence*, characterised by two complementary properties: completeness and soundness. The first means that every possible execution of the source language may be simulated by its translation, the second ensures that all the states reached by the translation correspond to some state of the source. Since all the languages we consider are Turing-complete (even Brane [28], despite of its simplicity), as usual for concurrent languages we require some additional criteria. As remarked in [29], a *reasonable* encoding should also preserve the degree of distribution of the source language (i.e. homomorphism w.r.t. parallel composition) and should not depend on the channel (or compartment) names of the term to be encoded. This also implies a very valuable property, that is modular compilation, as discussed in [30]. In addition to the cited criteria, we also require the encoding to preserve the termination or diverging behaviour of the translated term, in order to obtain a totally faithful encoding function. The following definition formalises the notion of *reasonable encoding* used in this paper.

Definition 9. *An encoding $[\![\cdot]\!]$ is reasonable if it enjoys the following properties:*

1. *homomorphism w.r.t. parallel composition:*

$$[\![\, P_1 \mid P_2 \,]\!] = [\![\, P_1 \,]\!] \mid [\![\, P_2 \,]\!]$$

2. *renaming preserving: for any permutation of the source names θ,*

$$[\![\, \theta(P) \,]\!] \;=\; \theta([\![\, P \,]\!])$$

3. *termination invariance:*

$$P\Downarrow \quad \Longleftrightarrow \quad [\![\, P \,]\!]\Downarrow \qquad\qquad P\Uparrow \quad \Longleftrightarrow \quad [\![\, P \,]\!]\Uparrow$$

4. *operational correspondence:*
   (a) *if $P \to P'$ then $[\![\, P \,]\!] \to^* [\![\, P' \,]\!]$,*
   (b) *if $[\![\, P \,]\!] \to^* Q$ then $\exists P' : P \to^* P' \wedge Q \to^* [\![\, P' \,]\!]$.*

4.3 BioAmbients

The BioAmbient calculus rises as an enhancement of the π-Calculus in order to overcome the same technical difficulties we encountered during the modelling example of insuline secretion in β cells. More precisely, BioAmbients joins the communication power of the π-Calculus and the compartment abstraction given by the Ambient calculus [25, 26]. Compartments are represented by ambients, denoted by square brackets:

$$Sys \triangleq [P \mid Q \mid [R]]$$

The above system Sys is composed of one root ambient containing three elements: the processes P and Q, and another nested ambient $[R]$, which in turns contains another process R.

Processes can communicate in the style of the π-Calculus, but communication capabilities are extended to fit the needs of the new setting denoted by ambients. Processes can interact if they lie in the same ambient or in *nearby* ambients, where two ambients are nearby if one of them is directly nested into the other or they are children of the same parent ambient. Therefore, four directions of communications are introduced:

- intra-ambient, for processes inside the same ambient;
- sibling-to-sibling, when processes lie in compartments children of the same parent ambient;
- child-to-parent, for a process willing to communicate with a process located in the parent ambient;
- parent-to-child, representing the counterpart of the previous one.

Intra-ambient communication is denoted by the prefix *local*:

$$Sys \triangleq [local\ chan!\{d\}.P \mid local\ chan?\{x\}.Q \mid [R]]$$

The process $local\ chan!\{d\}.P$ is ready to send some datum d over the channel $chan$ to some process which must be located into same ambient. $local\ chan?\{x\}.Q$

is a candidate for such synchronisation, since it is listening on the same channel inside the same ambient. For these reason, the above system can reduce in the following way:

$$\begin{aligned} Sys \triangleq\ & [local\ chan!\{d\}.P \mid local\ chan?\{x\}.Q \mid [R]] \quad \rightarrow \\ & [P \mid Q\{d/x\} \mid [R]] \end{aligned}$$

The effect of the communication is the same as for the π-Calculus: the name d is received and substituted for the local placeholder x in Q.

Sibling-to-sibling communication is denoted by the prefix $s2s$:

$$\begin{aligned} Sys \triangleq\ & [[s2s\ chan!\{d\}.P \mid Q] \mid [s2s\ chan?\{x\}.R \mid S]] \quad \rightarrow \\ & [[P \mid Q] \mid [R\{d/x\} \mid S]] \end{aligned}$$

Parent-to-child and child-to-parent are complementary, denoted by $c2p$ and $p2c$ prefixes, and permit both the input/output directions:

$$\begin{aligned} Sys_1 \triangleq\ & [[c2p\ chan!\{d\}.P \mid Q] \mid p2c\ chan?\{x\}.R \mid S] \quad \rightarrow \\ & [[P \mid Q] \mid R\{d/x\} \mid S] \\ Sys_2 \triangleq\ & [[c2p\ chan?\{x\}.P \mid Q] \mid p2c\ chan!\{d\}.R \mid S] \quad \rightarrow \\ & [[P\{d/x\} \mid Q] \mid R \mid S] \end{aligned}$$

In Sys_1 the outer process R receives the datum, while in Sys_2 it sends the datum to the inner process P.

In addition to the above communications, BioAmbients inherits from Mobile Ambients the operations needed to change dynamically the structure of nesting of ambients. Such operations are called *capabilities*. Processes cause the movement of whole ambients across the nesting tree, in agreement with the basic intuitions behind compartment semantics. For example, it is possible to cause the merging of two sibling ambients (children of the same parent ambient) in the following way:

$$\begin{aligned} & [[merge{+}\ n.P \mid Q] \mid [merge{-}\ n.R \mid S] \mid T] \quad \rightarrow \\ & [[P \mid Q \mid R \mid S] \mid T] \end{aligned}$$

The $merge+$ and $merge-$ capabilities are complementary and can be triggered only if the name n matches.

Ambients can also enter some other sibling ambient

$$\begin{aligned} & [[enter\ n.P \mid Q] \mid [accept\ n.R \mid S] \mid T] \quad \rightarrow \\ & [[[P \mid Q] \mid R \mid S] \mid T] \end{aligned}$$

or exit their own parent ambient

$$\begin{aligned} & [[[exit\ n.P \mid Q] \mid expel\ n.R \mid S] \mid T] \quad \rightarrow \\ & [[[P \mid Q] \mid [R \mid S] \mid T] \end{aligned}$$

by additional complementary capabilities, *enter*/*accept* and *exit*/*expel*.

We now give the definitions for the syntax, structural congruence and reduction semantics of BioAmbients, in the same style of π-Calculus and π@ semantics. For further details we refer to [11].

Definition 10. *Let $\mathcal{N}$ be a set of names on a finite alphabet, $n, m, p, \ldots \in \mathcal{N}$. The syntax of BioAmbients is defined as*

$$\begin{array}{rcl}
\pi & ::= & \$n!\{m\} \;\Big|\; \$n?\{m\} \\
\$ & ::= & local \;\Big|\; s2s \;\Big|\; p2c \;\Big|\; c2p \\
M, N & ::= & enter\ n \;\Big|\; accept\ n \;\Big|\; exit\ n \;\Big|\; expel\ n \;\Big|\; merge{+}\ n \;\Big|\; merge{-}\ n \\
P, Q & ::= & (new\ n)P \;\Big|\; P \mid Q \;\Big|\; !\ P \;\Big|\; [P] \;\Big|\; \sum_{i \in I} \pi_i.P_i \;\Big|\; \sum_{i \in I} M_i.P_i
\end{array}$$

Definition 11. *The congruence relation $\equiv$ is defined as the least congruence satisfying the following rules:*

$$\begin{array}{rl}
P|Q \equiv Q|P & (P|Q)|R \equiv P|(Q|R) \\
P|\mathbf{0} \equiv P & [\mathbf{0}] \equiv \mathbf{0} \\
!\mathbf{0} \equiv \mathbf{0} & !P \equiv P|!P \\
(new\ n)\mathbf{0} \equiv \mathbf{0} & (new\ n)(new\ m)P \equiv (new\ m)(new\ n)P \\
(new\ n)(P|Q) \equiv P|(new\ n)Q & \textit{if } n \notin \mathrm{fn}(P) \\
(new\ n)[P] \equiv [(new\ n)P] & \\
\$n?\{m\}.P \equiv \$n?\{p\}.P\{p/m\} & \textit{if } p \notin \mathrm{fn}(P) \\
(new\ n)P \equiv (new\ m)P\{m/n\} & \textit{if } m \notin \mathrm{fn}(P)
\end{array}$$

where $\mathrm{fn}(P)$ *is naturally extended to BioAmbients processes.*

Definition 12. *BioAmbients semantics is given in terms of the reduction system described by the following rules:*

$$\begin{array}{rcl}
[(T + enter\ n.P)|Q]|[T' + accept\ n.R)|S] & \to & [[P|Q]|R|S] \\
[[(T + exit\ n.P)|Q]|(T' + expel\ n.R)|S] & \to & [P|Q]|[R|S] \\
[(T + merge{+}\ n.P)|(Q]|[T' + merge{-}\ n.R)|S] & \to & [P|Q|R|S] \\
(T + local\ n!\{m\}.P)|(local\ n?\{p\}.Q + T') & \to & P|Q\{m/p\} \\
(T + p2c\ n!\{m\}.P)|[(c2p\ n?\{p\}.Q + T')|R] & \to & P|[Q\{m/p\}|R] \\
[R|(T + c2p\ n!\{m\}.P)]|(p2c\ n?\{p\}.Q + T') & \to & [R|P]|Q\{m/p\} \\
[R|(T + s2s\ n!\{m\}.P)]|[(s2s\ n?\{p\}.Q + T')|S] & \to & [R|P]|[Q\{m/p\}|S]
\end{array}$$

$$\frac{P \;\to\; Q}{(\nu\ n)P \;\to\; (\nu\ n)Q} \quad \frac{P \;\to\; Q}{[P] \;\to\; [Q]} \quad \frac{P \;\to\; Q}{P|R \;\to\; Q|R} \quad \frac{P \equiv P' \quad P \;\to\; Q \quad Q \equiv Q'}{P' \;\to\; Q'}$$

4.4 Modelling the insuline example in BioAmbients

The example of Fig. 1 can be exploited again to explain the basic modelling ideas which may be applied in BioAmbients.

Since BioAmbients embeds directly the π-Calculus (we can obtain a straightforward translation just by substituting each π input/output operation with a *local* communication) we may start sketching the modelling of the GLU molecule of Expr. (4) in the following way:

$$GLU(\ldots) \triangleq \quad (\nu\ g)(local\ g!\{\ldots\} \mid !\ local\ g?\{\ldots\}.GLU'(g,\ldots))$$

GLU' should both listen for a possible transportation of the molecule into another compartment and communicate its ability of reacting as glucose. We may then try to take advantage of the multiple directions of communication of BioAmbients and use just a unique name glu in the *local* direction for reacting, and in the $c2p$ and $p2c$ directions for modelling the movement to a new compartment:

$$\begin{array}{rcl} GLU'(g, glu, \ldots) & \triangleq & local\ glu!\{\}.PYR(\ldots) + \\ & & p2c\ glu?\{g', glu', \ldots\} + \\ & & c2p\ glu?\{g', glu', \ldots\} \end{array}$$

Unfortunately, in this way we cannot exploit at all the abstraction of ambient and neither the directions of communication, because we are still modelling the localisation of processes by means of names: even if GLU' receives some new set of names after the $p2c$ or $c2p$ communication, it is not able to change ambient without some *exit* or *enter* capability. Consequently, in order to take some advantage from the novel primitives introduced in BioAmbients, we must forget the idea of name as means to model mobility or localisation, and leave this job to ambients: even a simple molecule like GLU should be modelled as an ambient on its own. Furthermore it may be possible to exploit the way ambients move as a whole in order to simplify the expression for GLU:

$$GLU \triangleq \quad [!\ enter\ glu \mid !\ exit\ glu \mid c2p\ glu?\{\}.PYR]$$

The above formalisation embeds the idea that the movement into a new ambient and the chemical reactivity are completely orthogonal and independent. The *exit glu* capability is superfluous, since in the simple model of the system we are considering the glucose molecules are not going to leave the cell once they entered, but it is valid in general for any molecule or element undergoing some sort of "passive" conveyance into other compartments without knowledge of its direction. The movement of the ambient as a whole preserves the integrity of the process even if composed of several parallel subprocesses, but the previous expression does not describe correctly the real behaviour of the glucose molecule: in fact, after its degradation into PYR, the GLU process is still able to be transported across compartments, even if the molecule itself may not exist anymore. Consequently we are forced to exploit the same expedient used for π-Calculus and π@ modelling, even in presence of ambients:

$$\begin{aligned} GLU \triangleq \quad & [local\ glu!\{\} \mid !\ local\ glu?\{\}.(enter\ glu.local\ glu!\{\} + \\ & \qquad exit\ glu.local\ glu!\{\} + \\ & \qquad c2p\ glu?.PYR)] \end{aligned}$$

Now the movement of the molecule is allowed in either direction and disabled after its degratation. It is worth noticing how the introduction of multiple communication directions (*local*, $s2s$, $p2c$, $c2p$) and capabilities (*merge*, *enter*/*accept*, *exit*/*expel*) reduces the number of channel names needed: each name embeds in fact seven distinct interactions. The scoping induced by ambients allows also to avoid the use of restriction, at least in this simple case.

The corresponding expression for the glucose channel becomes very simple:

$$GLUCHAN \triangleq \quad !\ accept\ glu$$

Even if the previous expression of GLU is correct intuitively, the mixing of communications and capabilities is not allowed in BioAmbients. A slight correction allows to overcome this issue:

$$\begin{aligned} GLU \triangleq \quad & [local\ glu!\{\} \mid !\ local\ glu?\{\}.(s2s\ gludock!\{\}.enter\ glu.local\ glu!\{\}+ \\ & \qquad c2p\ gludock!\{\}.exit\ glu.local\ glu!\{\}+ \\ & \qquad c2p\ glu!.PYR)] \end{aligned}$$

and $GLUCHAN$ must be corrected accordingly:

$$GLUCHAN \triangleq \quad !\ s2s\ gludock?\{\}.accept\ glu$$

The direction $s2s$ is justified by the structure of the system: the GLU molecule external to the cell is an ambient sibling of the ambient represented by the cell itself, where the process $GLUCHAN$ resides.

The formalisation of GLU as an ambient affects the expression of ADP. The *local* reaction (inside the cell) of glycolysis becomes a inter-ambient communication, reflected by the $p2c$ direction:

$$ADP \triangleq \quad p2c\ glu?\{\}.ATP$$

The interaction of ATP can be instead considered *local*, as soon as the potassium channel is not modelled as an ambient:

$$ATP \triangleq \quad local\ inhk!\{\}$$

Under this hypothesis, the process $KCHAN$ is not substantially different from the corresponding π-Calculus expression:

$$\begin{aligned} KCHAN \triangleq \quad & (\nu\ kc)(local\ kc!\{\} \mid !\ local\ kc?\{\}.(local\ inhk?\{\} \\ & \quad + p2c\ kdock!\{\}.expel\ k\{\}.local\ kc!\{\})) \end{aligned}$$

The loop which spawns a new $KCHAN$ subprocess is disabled after the inhibition by the ATP molecule. The expulsion of potassium K from the current ambient is modelled in agreement with the previous considerations about the molecule GLU:

$$\begin{aligned} K \triangleq \quad & [local\ k!\{\} \mid !\ local\ k?\{\}.(s2s\ kdock!\{\}.enter\ k.local\ k!\{\}+ \\ & \qquad\qquad\qquad\qquad\quad c2p\ kdock!\{\}.exit\ k.local\ k!\{\}+ \\ & \qquad\qquad\qquad\qquad\quad c2p\ k!)] \end{aligned}$$

The other processes can be encoded by following similar considerations:

$$\begin{aligned} POL \triangleq \quad & p2c\ k?\{\}.local\ act!\{\} \\ CACHAN \triangleq \quad & local\ act?\{\}.!\ s2s\ cadock!\{\}.accept\ ca \\ CA \triangleq \quad & [local\ ca!\{\} \mid !\ local\ ca?\{\}. \\ & \quad (s2s\ cadock!\{\}.enter\ ca.local\ ca!\{\}+ \\ & \quad c2p\ cadock!\{\}.exit\ ca.local\ ca!\{\}+ \\ & \quad c2p\ ca!\{\})] \\ DOCKP \triangleq \quad & p2c\ ca?\{\}.expel\ dockves \end{aligned}$$

The encoding of the vesicle VES requires more attention. In the biological model, the insuline molecules do *never* lie inside the cell. Therefore, if in BioAmbients VES is represented by an ambient, one way to carry the operation would be to make the VES ambient exit the cell and afterwards dump all the insuline molecules INS in the blood:

$$VES \triangleq \quad [exit\ dockves.!\ expel\ ins \mid [exit\ ins.INS] \mid \cdots \mid [exit\ ins.INS]]$$

Anyway, this encoding does not exploit the expressive power of ambients. A smarter approach would merge the content of VES with the ambient corresponding to the blood vessel after the expulsion of the vesicle from the cell:

$$VES \triangleq \quad [exit\ dockves.merge+\ \ insves \mid [INS] \mid \cdots \mid [INS]]$$

This encoding requires that some complementary process is ready for the merge operation in the parent ambient of the cell:

$$VESMERGE \triangleq \quad !\ merge-\ \ insves$$

It is worth remarking that in both cases the exocytosis process is *not modelled atomically.* Finally, the system is given by the following expression:

$$\begin{aligned} SYS \triangleq \quad & GLU \mid K \mid CA \mid VESMERGE \mid \\ & [GLU \mid GLUCHAN \mid K \mid KCHAN \mid ADP \mid ATP \mid POL \mid \\ & \quad CA \mid CACHAN \mid DOCKP \mid VES] \end{aligned}$$

Thanks to the introduction of ambients, there is no need to keep explicit trace of the free names of each process. Unfortunately, possible problems emerging from the need of atomicity for complex operations are not resolved, exactly like in π-Calculus.

After this short introduction to BioAmbients modelling, we are now able to grasp the key ideas which allow to encode BioAmbients into the $\pi@$ calculus.

4.5 Encoding BioAmbients into $\pi@$

As we have just seen, ambients are containers organised in a tree structure: running processes and nested sub-ambients are located inside them. If each node of the tree represents an ambient, nodes are complex structures: each node may contain zero or more parallel processes and may interact with zero or more nested sub-ambients. Consequently, for the implementation of this tree structure into $\pi@$, each encoded BioAmbients process must be aware of the name of its containing (immediate) ambient, but also of the name indicating the parent of its immediate ambient. In other words, the encoding function $[\![\cdot]\!]^{\alpha}$ from BioAmbients processes to $\pi@$ processes which we are now ready to formalise requires the (bound) names a and pa, representing the immediate ambient and the parent ambient of each BioAmbients process respectively. This is in accordance with the modelling of cross-compartment objects in $\pi@$ (like ion channels) previously shown: in fact, every BioAmbients process may likely interact with some other process placed in some child or parent or sibling ambient, that is every BioAmbients process is a potential cross-compartment element.

The similarity between the basic π-Calculus operators and the corresponding operators inherited by Bioambients allows to encode some of them homomorphycally in $\pi@$. This is the case of parallel composition and restriction:

$$\begin{array}{rcl}
[\![P \mid Q]\!]^{\alpha}_{a,pa} & \triangleq & [\![P]\!]^{\alpha}_{a,pa} \mid [\![Q]\!]^{\alpha}_{a,pa} \\
[\![(new\ n)P]\!]^{\alpha}_{a,pa} & \triangleq & (\nu\ n)[\![P]\!]^{\alpha}_{a,pa}
\end{array}$$

According to the previous considerations, the encoding function is decorated with two names representing the ambient a where the encoded process resides and the outer ambient pa. The name a of the ambient can be used for *local* communication directions: [1]

$$\begin{array}{rcl}
[\![local\ n!\{m\}.P]\!]^{\alpha *}_{a,pa} & \triangleq & \overline{local@n@a}\langle m\rangle.([\![P]\!]^{\alpha}_{a,pa}) \\
[\![local\ n?\{m\}.P]\!]^{\alpha *}_{a,pa} & \triangleq & local@n@a(m).([\![P]\!]^{\alpha}_{a,pa})
\end{array}$$

The name pa of the parent ambient can be exploited for any $s2s$, $c2p/p2c$ or ambient capability (*merge*, *exit*/*expel*, *enter*/*accept*). For example, the $s2s$ com-

[1] The difference between the two mutually recursive encoding functions $[\![\ ]\!]^{\alpha}_{a,pa}$ and $[\![\ ]\!]^{\alpha *}_{a,pa}$ will be clarified later on.

munication can be encoded in the following way:

$$\begin{array}{rcl} [\![\ s2s\ n!\{m\}.P\ ]\!]^{\alpha *}_{a,pa} & \triangleq & \overline{s2s@n@pa}\langle m\rangle.([\![\ P\ ]\!]^{\alpha}_{a,pa}) \\ [\![\ s2s\ n?\{m\}.P\ ]\!]^{\alpha *}_{a,pa} & \triangleq & s2s@n@pa(m).([\![\ P\ ]\!]^{\alpha}_{a,pa}) \end{array}$$

This encoding explains *where* such sibling-to-sibling communication is reasonably happening, that is inside the only compartment known by both processes: their parent ambient. Nevertheless, the effect of the communication is limited to the substitution of a name in the scope of the receiving process $[\![\ P\ ]\!]^{\alpha}_{a,pa}$. The possibility of exploiting an unbounded number of names (three in this case) for each $\pi@$ channel allows to model easily the "triple matching" typical of BioAmbients actions: in fact, in addition to their "proximity" (i.e. the localisation in the same ambient or parent ambient), the interaction between two processes can happen only if both the direction (*local*, $c2p/p2c$ or $s2s$) and the name n match.

The same considerations hold for $c2p/p2c$ communication:

$$\begin{array}{rcl} [\![\ p2c\ n!\{m\}.P\ ]\!]^{\alpha *}_{a,pa} & \triangleq & \overline{p2c@n@a}\langle m\rangle.([\![\ P\ ]\!]^{\alpha}_{a,pa}) \\ [\![\ c2p\ n?\{m\}.P\ ]\!]^{\alpha *}_{a,pa} & \triangleq & p2c@n@pa(m).([\![\ P\ ]\!]^{\alpha}_{a,pa}) \end{array}$$

While two processes able to perform a *local* or $s2s$ communication lie in a symmetrical position, the $c2p/p2c$ operation introduces asymmetry in the system, since one process must be located in the parent ambient of the other. This is reflected by the compartment names used in the $\pi@$ encoding: the outer process uses its ambient name a to communicate, while the inner process its parent ambient name pa. There would be no way to make such operation happen inside the child ambient, since the outer $\pi@$ process does not know (and *must* not know, in agreement with the abstraction of compartment scoping previously discussed) any name associated with the inner ambients.

The correspondence between ambient and parent ambient names of nested compartments is stated by the encoding of the compartment operator $[\cdot]$:

$$[\![\ [\ P\ ]\ ]\!]^{\alpha}_{a,pa} \quad \triangleq \quad (\nu\ c)[\![\ P\ ]\!]^{\alpha}_{c,a}$$

At first sight it corresponds to the encoding of the restriction operator of BioAmbients. The substantial difference can be devised in the names appearing as parameters of the encoding function in the right-hand side of the expression: the restricted name c represents the new ambient of the process P, while a represents both the ambient of $[\ P\ ]$ and the parent ambient of P.

So far, all the problems deriving from the introduction of compartments have been solved by the only use of polyadic synchronisation. The atomicity-related problems noticed during the formalisation of compartments in the π-Calculus would emerge now in the attempt of encoding BioAmbients capabilities in $\pi@$. Consider for example the simple *merge* capability:

$$\begin{array}{l} [merge+\ .P \mid Q_1 \mid \cdots \mid Q_n] \mid [merge-\ .R \mid S_1 \mid \cdots \mid S_m] \quad \rightarrow \\ [P \mid Q_1 \mid \cdots \mid Q_n \mid R \mid S_1 \mid \cdots \mid S_m] \end{array} \qquad (7)$$

The encoding in $\pi @$ of the above system before the reduction would require two distinct restricted names corresponding to each of the ambients ready to be merged. After their merging, only one of such names must be present. This means that either $n+1$ or $m+1$ $\pi @$ processes must have replaced the name corresponding to their current ambient. Furthermore, BioAmbients semantics requires that all of these $n+1$ or $m+1$ changes in the structure of the system happen instantaneously, that is no other communication or capability can be executed meanwhile. The only way to grant this constraint in the standard π-Calculus (without introducing some divergent behaviour) would be to put as guard some *centralised monitor process* which enables the occurrence of one operation at a time, with the consequence of excluding any concurrent feature from the encoded system.

The presence of priority in $\pi @$ allows instead to overcome easily such kind of issues without any explicit centralised mechanism. The occurrence of the above *merge* synchronisation can be followed by a *sequence of high priority synchronisations* which notify all the involved processes of their new ambient (or parent ambient) name. This sequence of high priority operation is actually a *loop* which must terminate when all the addressed processes have received the desired data. A remarkable feature of such loop is that for any compartment-related operation of BioAmbients (but also of Brane) *all these processes reside in the same compartment*. This peculiarity allows to express the loop in a very general way:

$$BCAST \quad \triangleq \quad !\,\underline{\underline{bcast}}(x,y,z).(\underline{\underline{\overline{x@y}}}\langle z\rangle.\underline{\underline{\overline{bcast}}}\langle x,y,z\rangle + \underline{\tau})$$

The $BCAST$ process implements a loop triggered by the receiving of three names: the first two names are used to identify the channel $x@y$ over which the communication loop will be executed. The third name is a datum which is sent to all the processes listening on $x@y$, representing the name of some ambient where some group of processes is going to move. The use of two levels of (high) priority (denoted by single or double-underlined actions, corresponding to integer levels 1 and 0 respectively) provides two very important properties:

- once triggered, the loop will execute entirely before the occurrence of any other encoded BioAmbients operation, since all such operations – both communications, as we have showed, and capabilities, as we will show – are guarded by an initial *low priority* (integer level 2 of priority, corresponding to no underline) actions;
- the loop will end without leaving any "garbage" which may interfere with the system later on.

The first property is granted by the high priority of the replicated *bcast* input. After the communication of the datum z over the channel $x@y$, the process spawns another copy of itself by means of the same expedient previously showed for the encoding of recursive behaviour by exploitation of replication. The presence of the τ action with an intermediate prioritised level allows to terminate the loop when all the processes listening on $x@y$ have actually received the datum z, so that the remainder of the $BCAST$ process disappears completely *before* any other communication or capability may be performed.

Consequently, the implementation in $\pi@$ of the *merge* capability requires to trigger the $BCAST$ loop with the correct parameters. First, we need to understand *where* two processes performing a *merge*+ /*merge*– capability are located: by checking the semantic rules of Def. 12 we can see that such processes must be located in sibling ambients (ambients that are children of the same outer ambient), i.e. the corresponding synchronisation in $\pi@$ happens in the parent ambient, exactly as showed for the $s2s$ communication. Second, the communication must carry on the name associated with one of the two ambients, so that the receiving process can use it as new compartment name for all of its subsequent synchronisations and forward it to all the other processes affected by the structural change in the ambient tree. Since the merging of two ambients is a symmetrical operation, the choice of the name to be communicated is arbitrary. Here we choose to keep the name associated with the ambient of the process exhibiting the *merge*+ capability:

$$[\![\ merge+\ \ n.P\ ]\!]^{\alpha *}_{a,pa} \quad \triangleq \quad \overline{merge@n@pa}\langle a\rangle.([\![\ P\ ]\!]^{\alpha}_{a,pa})$$

In agreeement with the above considerations, the encoding of *merge*+ is located in the parent ambient pa and communicates the name a of its ambient to the process ready for the complementary *merge*–:

$$\begin{aligned}[\![\ merge-\ \ n.P\ ]\!]^{\alpha *}_{a,pa} \quad \triangleq \quad & merge@n@pa(x). \\ & \underline{\overline{bcast}}\langle merge, a, x\rangle.([\![\ P\ ]\!]^{\alpha}_{x,pa})\end{aligned}$$

The *merge*– operation implies the receiving of some name to be substituted for the placeholder x which will be used as new ambient in $([\![\ P\ ]\!]^{\alpha}_{x,pa})$, while the parent ambient pa remains unchanged exactly like in the original BioAmbients system. The subterm $\underline{\overline{bcast}}\langle merge, a, x\rangle$ triggers the $BCAST$ loop which in turn notifies all the sibling processes (represented by the siblings $S_1, \ldots, S_m$ of R in Expr. (7)) of their new ambient x (which, again, at this time has been replaced by the name of the ambient sent by the process performing the complementary *merge*+ capability). The name *merge* used to denote the channels $merge@n@a$ and $merge@n@pa$ is completely arbitrary and allows to distinguish this broadcast-like communication occurring after the merging of compartments from the other broadcast-like loops that are triggered by the translation of the other BioAmbients capabilities.

Similar considerations lead to the encoding of the *enter*/*accept* and *exit*/*expel* capabilities. Their partial asymmetry requires to bring attention to the names transmitted during the reduction and the way they are used afterwards.

The *enter*/*accept* reduction is triggered by two processes whose localisation is symmetrical w.r.t. the global tree structure of the system: they are inside ambients children of the same parent ambient, exactly like in the case of the *merge* operation. After the reduction, anyway, their situation is asymmetrical since their respective ambients are one the child of the other. Consequently, the process performing the *accept* capability is not going to change location but must communicate the name corresponding to its ambient to the process ready for the *enter* capability, which will use such name to denote its new parent ambient:

$$\begin{array}{rcl} [\![\, accept\ n.P \,]\!]^{\alpha *}_{a,pa} & \triangleq & \overline{enter@n@pa}\langle a\rangle.([\![\, P \,]\!]^{\alpha}_{a,pa}) \\ [\![\, enter\ n.P \,]\!]^{\alpha *}_{a,pa} & \triangleq & enter@n@pa(x).\overline{\underline{bcast}}\langle pa, a, x\rangle.([\![\, P \,]\!]^{\alpha}_{a,x}) \end{array}$$

As for the *merge*– operation, the encoding of the *enter* capability triggers the $BCAST$ loop which notifies all the involved processes of their new parent ambient.

The *exit*/*expel* reduction is the converse of the previous one: the two processes performing it lie in an asymmetrical position and end to be symmetrically distributed w.r.t. the ambient tree. In this case the process exhibiting the *exit* capability and all of its siblings must be notified of their new parent ambient, represented by the name of the parent ambient of the process performing the complementary *expel*:

$$\begin{array}{rcl} [\![\, expel\ n.P \,]\!]^{\alpha *}_{a,pa} & \triangleq & \overline{expel@n@a}\langle pa\rangle.([\![\, P \,]\!]^{\alpha}_{a,pa}) \\ [\![\, exit\ n.P \,]\!]^{\alpha *}_{a,pa} & \triangleq & expel@n@pa(x).\overline{\underline{bcast}}\langle pa, a, x\rangle.([\![\, P \,]\!]^{\alpha}_{a,x}) \end{array}$$

The *enter*/*accept* and *exit*/*expel* capabilities differ from *merge*+ /*merge*– for a subtle particular: in the *merge*– operation the name *merge* was sent as first parameter to $BCAST$, while in the other two the name *pa* was used. We may have used two distinct names $enter, exit$ for indicating the change of the name of the parent compartment in consequence of each of such capabilities, but their effect is actually the same, so they can be condensed in the simple idea of "substitution of parent compartment name". Similar reasoning can be argued for the encoding of Brane actions.

A substantial consequence of the above encodings of BioAmbients capabilities is the silent assumption that *all the encoded processes are always listening on the right channels for possible changes of their ambient or parent ambient names*, independently of the other actions they are ready to perform. Furthermore, *after the receiving of a new ambient or parent ambient name in consequence of the triggering of the $BCAST$ loop, each notified process must return exactly in its previous state*, except for the substitution of the old ambient or parent ambient name with the new one just received. As an example, consider again the situation of Expr. (7): the $\pi@$ processes corresponding to $S_1, \ldots, S_m$ are going to change their ambient name after the $BCAST$ loop triggered by P and R. This means that each of those m processes is listening on some channel $merge@a$, with a representing their current ambient name. More exactly, each of them is listening simultaneously on

- $merge@a$ in order to be notified of the merging of their ambient,
- $merge@pa$ for the merging of their parent ambient, and
- $pa@a$ for the change of the name of their parent ambient in consequence of an *exit* or an *accept* operation.

In order to denote concisely this property, we may say that the $\pi@$ encoding of each BioAmbients communication or capability is put in nondeterministic choice with the three options expressed by the following process:

$$TREE(b,a,pa) \quad \triangleq \quad \underline{\underline{pa@a}}(x).\overline{\underline{\underline{b}}}\langle a,x\rangle + \underline{\underline{merge@pa}}(x).\overline{\underline{\underline{b}}}\langle a,x\rangle + \underline{\underline{merge@a}}(x).\overline{\underline{\underline{b}}}\langle x,pa\rangle \tag{8}$$

The three expressions with shape $\overline{\underline{\underline{b}}}\langle a,x\rangle$ or $\overline{\underline{\underline{b}}}\langle x,pa\rangle$, as we will see later on, spawn another copy of the original process which will use the first received name as ambient and the second one as parent ambient. It is worth noticing the use of the received name x after each choice branch:

- the name received over $pa@a$ is used as new parent ambient name, as required by the *exit* and *enter* capabilities;
- the name received over $merge@a$ is used as new ambient name, when the merging operation affects the local ambient;
- the name received over $merge@pa$ is used as new parent ambient name, when the merging operation affects the parent ambient.

After the execution of the $BCAST$ loop, all the notified processes return to their previous state, that is they are ready again to behave as $S_1,\ldots,S_m$ in the case of Expr. (7), even if they are located in a new compartment. In other words, *the encoding of each communication or capability is represented by a loop which allows the process to return in its previous state after receiving some new ambient or parent ambient name.* With a slight abuse of notation we may then write that each communication or capability choice of BioAmbients can be encoded as follows:

$$\begin{aligned} &[\![\textstyle\sum_{i\in I,\ I\neq\emptyset} \xi_i.P_i]\!]^{\alpha}_{a,pa} \quad \triangleq \\ &\qquad (\nu\ s)(\overline{\underline{\underline{s}}}\langle a,pa\rangle \ \big|\ !\ \underline{\underline{s}}(na,npa).SUBSUM(s,na,npa)) \end{aligned} \tag{9}$$

where $SUBSUM$ is just a shorthand for the following expression (since its definition does not depend only on the names s, na, npa but also on the branches of the choice $\sum_i \xi_i.P_i$):

$$\begin{aligned} &SUBSUM(s,na,npa) \quad \triangleq \\ &\qquad \textstyle\sum_{i\in I,\ I\neq\emptyset} [\![\xi_i.P_i]\!]^{\alpha *}_{na,npa} \ + \ TREE(s,na,npa) \end{aligned} \tag{10}$$

Here, replication is exploited as usual for the modelling of recursive behaviour, which is needed in order to make each process return to the original state after the interaction with some triggered $BCAST$ loop. The presence of the $TREE$ process as part of the choice grants that each encoded process is listening on the right channels and ready to "be passively moved" inside some new ambient or parent ambient in consequence of the structural change in the nesting tree that has been triggered by another pair of processes. $\sum \xi_i.P_i$ represents either a choice between capabilities or between communications, since they are kept distinct in the definition of BioAmbients grammar. In the particular case of a single-branched choice in the form $\pi.P$ or $M.P$ the encoding is the same even if the choice operator is not written explicitly, because we still need to preserve the implicit choice branches with recursive behaviour expressed by $TREE$. The function $[\![\, \cdot \,]\!]^{\alpha *}$ decorated with an additional star denotes the encoding of each communication or capability, in order to distinguish it from the encoding of single-branched choices.

This π@ encoding of the choice operator determines severe consequences on the expression of BioAmbients replication. If we consider the following example

$$\begin{array}{ll} [\,!\, merge{+}\, n.P \mid Q] \mid [merge{-}\,.R \mid S\,] & \rightarrow \\ [\,!\, merge{+}\, n.P \mid \;P \mid Q \mid R \mid S\,] & \end{array} \tag{11}$$

then we may suppose to encode the replication homomorphically:

$$[\![\, !\, P \,]\!]^{\alpha}_{a,pa} \quad \triangleq \quad !\, [\![\, P \,]\!]^{\alpha}_{a,pa} \tag{12}$$

Unfortunately this would cause a high priority loop without termination in Expr. (9). In fact, each encoded process $[\![\, \sum \xi_i.P_i \,]\!]^{\alpha}_{a,pa}$ undergoes an internal, high priority reduction on the private channel s which spawns a new copy of a subprocess $SUBSUM(s, a, pa)$. Consequently, a direct replication of Expr. (9) by Expr. (12) would cause an unbounded number of such reductions to happen immediately. Their high priority level would hang the entire system. It would be possible to overcome the problem by correcting the encoding of the choice in the following way:

$$\begin{array}{l} [\![\, \sum_{i\in I,\, I\neq\emptyset} \xi_i.P_i \,]\!]^{\alpha}_{a,pa} \quad \triangleq \\ \qquad (\nu\, s)(SUBSUM(s, a, pa) \mid !\, \underline{\underline{s}}(na, npa).SUBSUM(s, na, npa)) \end{array} \tag{13}$$

with $SUBSUM$ corresponding to Expr. (10). The internal high priority reduction is eliminated because the spawned term $SUBSUM(s, a, pa)$ is now explicitly written in the encoding. The replication of Expr. (13) by Expr. (12) now can successfully translate the BioAmbients system of Expr. (11). In fact, after the first reduction on $merge@n@pa$, a copy of $[\![\, P \,]\!]^{\alpha}$ is spawned while the original $[\![\, !\, merge{+}\, n.P \,]\!]^{\alpha}$ is kept and ready to execute another $merge{+}$ reduction, in accordance with the right-hand side of Expr. (11).

Anyway, Expr. (12) is far from being correct. In fact, if we consider the following system

$$\begin{array}{ll} [merge{+}\,.P \mid Q] \mid [merge{-}\,.R \mid !\, S] & \rightarrow \\ [P \mid Q \mid R \mid !\, S] & \end{array} \tag{14}$$

where the replication acts on the process S which is sibling of the process undergoing the $merge{-}$ reduction, we may immediately notice the presence of another high priority non-terminating loop, even without unfolding the encoding of the whole system. We just need to recall the meaning of "replication of S" as "an unbounded number of copies of S":

$$! \, S \quad \equiv \quad ! \, S \mid S \quad \equiv \quad ! \, S \mid S \mid S \quad \equiv \quad ! \, S \mid S \mid S \mid S \mid \cdots$$

The encoding of replication of Expr. (12) would require *each of the copies of* $\llbracket S \rrbracket^{\alpha}$ *represented by* $! \llbracket S \rrbracket^{\alpha}$ *to be notified of the change of compartment name* triggered by the *merge* operation. Since such number of copies is unbounded and the loop is characterised by high priority, the system would hang immediately.

Consequently, we are forced to introduce an indirect encoding of BioAmbients replication by changing the intuitive abstraction of the replication operator itself. Even if $! \, P$ represents an unlimited number of copies of P, there is no need to unfold all of such copies. More precisely, we may think that there is no need to unfold more than just *one* copy of P at each time. In other words, we may represent $! \, P$ as $! \, P \mid P$ and consider to unfold a new copy of P *only* when its previous copy undergoes some reduction $P \to P'$. Actually this is equivalent to keep each replication in a sort of *normal form* where each replicated process is exactly unfolded once. Supposing that $P \to P'$, the reduction

$$! \, P \quad \to \quad ! \, P \mid P'$$

would be then written as

$$! \, P \mid P \quad \to \quad ! \, P \mid P \mid P'$$

where the copy of P in the right-hand side of the expression is unfolded from $! \, P$ only after the reduction of the first copy of P in the left-hand side to P'.

This behaviour can be straightforwardly obtained in $\pi @$ by encoding each BioAmbients replication as a loop where one copy of $\llbracket P \rrbracket^{\alpha}$ is always unfolded and undergoes the corresponding reductions of the BioAmbients process P. During any of such reductions, $\llbracket P \rrbracket^{\alpha}$ causes another copy of itself to spawn in order to grant that the semantics of $! \, P$ is preserved. Furthermore, the encoding of $! \, P$ must observe the same migration rules expressed by the $TREE$ process of Expr. (8). In order to allow $\llbracket P \rrbracket^{\alpha}$ to spawn a new copy of itself whenever it undergoes some reduction we must introduce a new parameter in the encoding function, which represents the private channel over which the spawning event will be communicated: this parameter is needed because the encoding of the action which causes P to reduce may be several nested calls later w.r.t. the recursive definition of $\llbracket \, ! \, P \rrbracket^{\alpha}$. In fact, consider the following system:

$$S \quad \triangleq \quad ! \, (\mathit{local} \; n?.Q_1' \mid Q_2)$$

Its encoding in $\pi @$ requires three recursive calls of the encoding function $\llbracket \cdot \rrbracket^{\alpha}$, one for each of the following subsystems:

$$S \triangleq ! \, P \qquad\qquad P \triangleq Q_1 \mid Q_2 \qquad\qquad Q_1 \triangleq \mathit{local} \; n?.Q_1'$$

By the previous considerations, we can keep one copy of P always unfolded and write S in the following way:

$$S \quad \triangleq \quad !\,P \mid P$$

After the reduction of Q_1 to Q_1', the system should appear as

$$S \mid local\ n!.\mathbf{0} \quad \rightarrow \quad !\,P \mid P \mid Q_1' \mid Q_2$$

where one copy of P is still unfolded. The unfolding of the corresponding $\pi@$ process $[\![\ P\ ]\!]^{\alpha}$ must be triggered by the *local* n communication of $[\![\ Q_1\ ]\!]^{\alpha}$. In other words, the $\pi@$ process $[\![\ Q_1\ ]\!]^{\alpha}$ must cause $[\![\ !\,P\ ]\!]^{\alpha}$ to spawn a new copy of $[\![\ P\ ]\!]^{\alpha}$, even if $[\![\ !\,P\ ]\!]^{\alpha}$ is defined recursively in function of $[\![\ P\ ]\!]^{\alpha}$, in turn defined in function of $[\![\ Q_1\ ]\!]^{\alpha}$. The only way to achieve this result is by introducing some new name k passed as parameter from $[\![\ !\,P\ ]\!]^{\alpha}$ until the recursive call of the encoding function $[\![\ Q_1\ ]\!]^{\alpha}$, which will use it to spawn a new copy of $[\![\ P\ ]\!]^{\alpha}$ as soon as the *local* n operation is executed. Hence, the encoding of the *local* communication (and of all the other communications and capabilities) should be modified as follows:

$$[\![\ local\ n?\{m\}.P\ ]\!]^{\alpha *}_{k,a,pa} \quad \triangleq \quad local@n@a(m).([\![\ P\ ]\!]^{\alpha}_{a,pa} \mid \underline{\underline{\overline{unfold@k}}}) \qquad (15)$$

$\underline{\underline{\overline{unfold@k}}}$ spawns a new copy of $[\![\ P\ ]\!]^{\alpha}$ in the encoding of $!\,P$:

$$\begin{array}{rcl} [\![\ !\,P\ ]\!]^{\alpha}_{a,pa} & \triangleq & (\nu\ k)(BANG(k,a,pa) \mid [\![\ P\ ]\!]^{\alpha}_{k,a,pa} \mid \\ & & !\ \underline{new@k}(na,npa).[\![\ P\ ]\!]^{\alpha}_{k,na,npa}) \\ BANG(k,a,pa) & \triangleq & !\ \underline{k}(na,npa).SUBBANG(k,na,npa) \\ & & \mid SUBBANG(k,a,pa) \\ SUBBANG(k,na,npa) & \triangleq & \underline{\underline{unfold@k}}.\overline{\underline{new@k}}\langle na,npa\rangle.\overline{\underline{k}}\langle na,npa\rangle + \\ & & TREE(k,na,npa) \end{array} \qquad (16)$$

The recursive call $[\![\ P\ ]\!]^{\alpha}_{k,na,npa}$, together with the new encodings suggested by Expr.(15), ensures that each communication or capability appearing after the above replication will spawn a new copy of $[\![\ P\ ]\!]^{\alpha}$ with a high priority reduction on the channel $unfold@k$. Such spawning is mediated by the $BANG$ subprocess, which regulates also the relocation of $[\![\ !\,P\ ]\!]^{\alpha}_{a,pa}$ by direct embedding of the $TREE$ subprocess discussed previously. It is worth remarking that such relocation in practice affects only the subprocess $BANG$, which is the only process regulating *where* (in terms of ambient and parent ambient) the following copies of $[\![\ P\ ]\!]^{\alpha}$ will be spawned. In accordance with the previous considerations, one copy (at least one, more precisely) of $[\![\ P\ ]\!]^{\alpha}$ is always kept unfolded. The relocation of this spawned copy into some new ambient or parent ambient after some *merge* or *exit*/*expel* or *enter*/*accept* capability happens transparently, since it already embeds in the correct way the subprocess $TREE$ as part of each encoded choice.

Two remarks allow to refine Expr. (16), which is still not correct.

First, replications may be *nested* and more than one unfolding may be needed after some reduction. As an example, consider the following system

$$S \triangleq !\,(\nu\ x)!\ local\ n?.R'$$

which we rewrite in terms of some additional shorthands:

$$S \triangleq !\ P \qquad P \triangleq (\nu\ x)Q \qquad Q \triangleq !\ R \qquad R \triangleq local\ n?.R'$$

The unfolding of one copy of each replicated process leads to the following expression:

$$S \quad \equiv \quad !\ P \mid (\nu\ x)(!\ R \mid R)$$

The reduction of R to R' requires *two unfoldings*: one of P and one of R, corresponding to *one unfolding for each replication appearing before the communication or capability* (i.e. each replication appearing as ancestor not followed by choice in the syntactic tree generating the BioAmbients expression). Consequently, one name k as additional parameter in the corresponding $\pi@$ encoding is not enough if more than one nested replication is present in the BioAmbients process. Hence, the parameter k must be replaced by the set $K = \{k_1, k_2, \ldots, k_n\}$ containing one name for each encoded replication which must be unfolded. This set is added with a new name after the encoding of each replication and becomes empty after the encoding of any BioAmbients choice, since any subsequent reduction of R' does not affect the replication of P and R anymore.

Second, the constant unfolding of one copy of a replicated process is not sufficient to express its full behaviour. Consider for example the following system:

$$S \quad \triangleq \quad !\ P \qquad\qquad P \ \triangleq \ (local\ n?.P_1 + local\ n!.P_2)$$

If we unfold only one copy of P, we miss the reduction that may happen between two distinct copies of P themselves. In fact we have that

$$S \quad \equiv \quad !\ P \mid P \mid P \qquad \rightarrow \qquad !\ P \mid P_1 \mid P_2$$

where P_1 and P_2 follow the reduction over *local* n. Consequently, at least *two* copies of $[\![\ P\ ]\!]^{\alpha}$ must be constantly kept unfolded in the corresponding encoded process.

Expr. (16) can be then modified as follows:

$$\begin{aligned}
[\![\ !\ P\ ]\!]^{\alpha}_{K,a,pa} \quad \triangleq \quad & (\nu\ k)(BANG(k,a,pa) \mid \\
& [\![\ P\ ]\!]^{\alpha}_{K\cup\{k\},a,pa} \mid [\![\ P\ ]\!]^{\alpha}_{K\cup\{k\},a,pa} \mid \\
& !\ \underline{new@k}(na,npa).[\![\ P\ ]\!]^{\alpha}_{K\cup\{k\},na,npa})
\end{aligned}$$

The encoding of communications and capabilities is updated accordingly:

$$\llbracket\, local\ n?\{m\}.P \,\rrbracket^{\alpha *}_{K,a,pa} \;\triangleq\; local@n@a(m).\Big(\llbracket\, P \,\rrbracket^{\alpha}_{\emptyset,a,pa} \mid \underline{\underline{\overline{unfold@k_1}}} \mid \cdots \mid \underline{\underline{\overline{unfold@k_n}}}\Big)$$

with $K = \{k_1, \ldots, k_n\}$.

The full definition of $\llbracket \cdot \rrbracket^{\alpha}$ is given in Table 1 and Table 2. Here, the names *oa* and *opa* represent two fictitious names needed for the correct initialisation of the encoding function, and corresponding to the outermost ambient and parent ambient of the entire encoded system.

The encoding function $\llbracket \cdot \rrbracket^{\alpha}$ enjoys the requirements discussed in section 4.2, as stated by the following theorem.

Theorem 1. $\llbracket \cdot \rrbracket^{\alpha}$ *is a reasonable encoding (modulo structural congruence), that is: let* P, P_1, P_2 *be BioAmbients processes, let* Q *be a* $\pi @$ *process, then*

1. $\llbracket P_1 \mid P_2 \rrbracket^{\alpha} = \llbracket P_1 \rrbracket^{\alpha} \mid \llbracket P_2 \rrbracket^{\alpha}$;
2. *for any permutation of the source names* θ, $\llbracket \theta(P) \rrbracket^{\alpha} = \theta(\llbracket P \rrbracket^{\alpha})$;
3. $P \Downarrow$ *iff* $\llbracket P \rrbracket^{\alpha} \Downarrow$, $P \Uparrow$ *iff* $\llbracket P \rrbracket^{\alpha} \Uparrow$;
4. *(a) if* $P \to P_1$ *then* $\exists P_2 : P_2 \equiv P_1 \wedge \llbracket P \rrbracket^{\alpha} \to^* \llbracket P_2 \rrbracket^{\alpha}$;
   (b) if $\llbracket P \rrbracket^{\alpha} \to^* Q$ *then* $\exists P_1 : P \to^* P_1 \wedge Q \to^* \llbracket P_1 \rrbracket^{\alpha}$.

4.6 Brane Calculi

The peculiar spatial rearrangement typical of biological membranes inspired the definition of Brane Calculi [9]. Here membranes constitute both the boundaries of compartments and the place where "computation" happens, that is where concurrent processes are thought to be located and to interact with the surrounding environment. Membranes are denoted by $(\!|\, \cdot \,|\!)$ and can be nested, exactly like ambients. In the following expression, the system S is composed of an outer membrane whose behaviour is specified by σ:

$$S \;\triangleq\; \sigma(\!|\, P \circ \rho(\!|\, Q \,|\!) \circ R \,|\!) \tag{17}$$

The outer membrane contains another membrane, whose behaviour is specified by ρ and whose content is Q.

S, P, Q, R are called *systems*: each system represents a collection of zero or more membranes that may be nested as we have just seen or composed in parallel by means of the operator $\circ$, like the three systems P, $\rho(\!|\, Q \,|\!)$ and R. Systems specify the global structure of each Brane expression, i.e. the shape of the tree of nested or sibling membranes.

σ and ρ are called *(mem)branes*: each brane specifies the behaviour of the membrane w.r.t. the other membranes, by indicating which type of *actions* may be performed. For example, if

$$\sigma \;\triangleq\; exo^{\perp}.\sigma' \qquad\qquad \rho \;\triangleq\; exo.\rho'$$

$$
\begin{array}{rcl}
[\![\, \mathbf{0} \,]\!]^{\alpha} & \triangleq & \mathbf{0} \\
[\![\, P \mid Q \,]\!]^{\alpha} & \triangleq & [\![\, P \,]\!]^{\alpha} \mid [\![\, Q \,]\!]^{\alpha} \\
[\![\, (new\ n)P \,]\!]^{\alpha} & \triangleq & [\![\, (new\ n)P \,]\!]^{\alpha}_{\emptyset,oa,opa} \\
[\![\, [\, P\,] \,]\!]^{\alpha} & \triangleq & [\![\, [\, P\,] \,]\!]^{\alpha}_{\emptyset,oa,opa} \\
[\![\, !\, P \,]\!]^{\alpha} & \triangleq & [\![\, !\, P \,]\!]^{\alpha}_{\emptyset,oa,opa} \\
[\![\, \mathbf{0} \,]\!]^{\alpha}_{K,a,pa} & \triangleq & \mathbf{0} \\
[\![\, P \mid Q \,]\!]^{\alpha}_{K,a,pa} & \triangleq & [\![\, P \,]\!]^{\alpha}_{K,a,pa} \mid [\![\, Q \,]\!]^{\alpha}_{K,a,pa} \\
[\![\, (new\ n)P \,]\!]^{\alpha}_{K,a,pa} & \triangleq & \nu\, n [\![\, P \,]\!]^{\alpha}_{K,a,pa} \\
[\![\, [\, P\,] \,]\!]^{\alpha}_{K,a,pa} & \triangleq & \nu\, c [\![\, P \,]\!]^{\alpha}_{K,c,a} \\
[\![\, !\, P \,]\!]^{\alpha}_{K,a,pa} & \triangleq & (\nu\, b)(BANG(b,a,pa) \mid \\
 & & \qquad [\![\, P \,]\!]^{\alpha}_{K\cup\{b\},a,pa} \mid [\![\, P \,]\!]^{\alpha}_{K\cup\{b\},a,pa} \mid \\
 & & \qquad !\, \underline{\underline{new@b}}(na,npa).[\![\, P \,]\!]^{\alpha}_{K\cup\{b\},na,npa}) \\
[\![\, \sum_{i\in I,\ I\neq\emptyset} \xi_i.P_i \,]\!]^{\alpha}_{K,a,pa} & \triangleq & BCAST \mid \nu\, s(!\, \underline{\underline{s}}(na,npa). \\
 & & \Big(\sum_{i\in I,\ I\neq\emptyset} [\![\, \xi_i.P_i \,]\!]^{\alpha *}_{K,na,npa} \\
 & & \qquad + TREE(s,na,npa) \Big) \mid \\
 & & \sum_{i\in I,\ I\neq\emptyset} [\![\, \xi_i.P_i \,]\!]^{\alpha *}_{K,a,pa} + TREE(s,a,pa)) \\
 & & \\
BANG(b,a,pa) & \triangleq & !\, \underline{\underline{b}}(na,npa).SUBBANG(b,na,npa) \\
 & & \mid SUBBANG(b,a,pa) \\
SUBBANG(b,na,npa) & \triangleq & \underline{\underline{unfold@b}}.\overline{\underline{\underline{new@b}}}\langle na,npa\rangle.\overline{\underline{\underline{b}}}\langle na,npa\rangle + \\
 & & \quad TREE(b,na,npa) \\
TREE(b,na,npa) & \triangleq & \underline{\underline{npa@na}}(x).\overline{\underline{\underline{b}}}\langle na,x\rangle + \underline{\underline{merge@npa}}(x).\overline{\underline{\underline{b}}}\langle na,x\rangle + \\
 & & \underline{\underline{merge@na}}(x).\overline{\underline{\underline{b}}}\langle x,npa\rangle \\
BCAST & \triangleq & !\, \underline{\underline{bcast}}(x,y,z).(\overline{\underline{\underline{x@y}}}\langle z\rangle.\overline{\underline{\underline{bcast}}}\langle x,y,z\rangle + \underline{\tau})
\end{array}
$$

Table 1. Encoding of BioAmbients processes into π@.

the system of Expr. (17) becomes

$$S \quad \triangleq \quad exo^{\perp}.\sigma'(\!|\; P \circ exo.\rho'(\!|\; Q \;|\!) \circ R \;|\!) \qquad (18)$$

σ specifies that the outer membrane is ready to perform an $exo^{\perp}$ action, while ρ is ready to perform the complementary exo action. The $exo/exo^{\perp}$ reduction

$$
\begin{array}{rcl}
[\![\ enter\ n.P\ ]\!]^{\alpha *}_{K,a,pa} & \triangleq & enter@n@pa(x).\overline{\overline{bcast}}\langle pa,a,x\rangle.([\![\ P\ ]\!]^{\alpha}_{\emptyset,a,x}\ \mid\ \Pi_K) \\
[\![\ accept\ n.P\ ]\!]^{\alpha *}_{K,a,pa} & \triangleq & \overline{enter@n@pa}\langle a\rangle.([\![\ P\ ]\!]^{\alpha}_{\emptyset,a,pa}\ \mid\ \Pi_K) \\
[\![\ exit\ n.P\ ]\!]^{\alpha *}_{K,a,pa} & \triangleq & expel@n@pa(x).\overline{\overline{bcast}}\langle pa,a,x\rangle.([\![\ P\ ]\!]^{\alpha}_{\emptyset,a,x}\ \mid\ \Pi_K) \\
[\![\ expel\ n.P\ ]\!]^{\alpha *}_{K,a,pa} & \triangleq & \overline{expel@n@a}\langle pa\rangle.([\![\ P\ ]\!]^{\alpha}_{\emptyset,a,pa}\ \mid\ \Pi_K) \\
[\![\ merge-\ \ n.P\ ]\!]^{\alpha *}_{K,a,pa} & \triangleq & merge@n@pa(x). \\
 & & \quad \overline{\overline{bcast}}\langle merge,a,x\rangle.([\![\ P\ ]\!]^{\alpha}_{\emptyset,x,pa}\ \mid\ \Pi_K) \\
[\![\ merge+\ \ n.P\ ]\!]^{\alpha *}_{K,a,pa} & \triangleq & \overline{merge@n@pa}\langle a\rangle.([\![\ P\ ]\!]^{\alpha}_{\emptyset,a,pa}\ \mid\ \Pi_K) \\
[\![\ local\ n!\{m\}.P\ ]\!]^{\alpha *}_{K,a,pa} & \triangleq & \overline{local@n@a}\langle m\rangle.([\![\ P\ ]\!]^{\alpha}_{\emptyset,a,pa}\ \mid\ \Pi_K) \\
[\![\ local\ n?\{m\}.P\ ]\!]^{\alpha *}_{K,a,pa} & \triangleq & local@n@a(m).([\![\ P\ ]\!]^{\alpha}_{\emptyset,a,pa}\ \mid\ \Pi_K) \\
[\![\ s2s\ n!\{m\}.P\ ]\!]^{\alpha *}_{K,a,pa} & \triangleq & \overline{s2s@n@pa}\langle m\rangle.([\![\ P\ ]\!]^{\alpha}_{\emptyset,a,pa}\ \mid\ \Pi_K) \\
[\![\ s2s\ n?\{m\}.P\ ]\!]^{\alpha *}_{K,a,pa} & \triangleq & s2s@n@pa(m).([\![\ P\ ]\!]^{\alpha}_{\emptyset,a,pa}\ \mid\ \Pi_K) \\
[\![\ p2c\ n!\{m\}.P\ ]\!]^{\alpha *}_{K,a,pa} & \triangleq & \overline{p2c@n@a}\langle m\rangle.([\![\ P\ ]\!]^{\alpha}_{\emptyset,a,pa}\ \mid\ \Pi_K) \\
[\![\ c2p\ n?\{m\}.P\ ]\!]^{\alpha *}_{K,a,pa} & \triangleq & p2c@n@pa(m).([\![\ P\ ]\!]^{\alpha}_{\emptyset,a,pa}\ \mid\ \Pi_K) \\
[\![\ c2p\ n!\{m\}.P\ ]\!]^{\alpha *}_{K,a,pa} & \triangleq & \overline{c2p@n@pa}\langle m\rangle.([\![\ P\ ]\!]^{\alpha}_{\emptyset,a,pa}\ \mid\ \Pi_K) \\
[\![\ p2c\ n?\{m\}.P\ ]\!]^{\alpha *}_{K,a,pa} & \triangleq & c2p@n@a(m).([\![\ P\ ]\!]^{\alpha}_{\emptyset,a,pa}\ \mid\ \Pi_K) \\
 & & \\
\Pi_K & \triangleq & \overline{\overline{unfold@k_1}}\ \mid\ \cdots\ \mid\ \overline{\overline{unfold@k_n}}\quad , \\
 & & \text{with } K=\{k_1,\ldots,k_n\}
\end{array}
$$

Table 2. Encoding of BioAmbients communications and capabilities into $\pi@$.

formalises exocytosis, corresponding to the same phenomenon described in the example of insuline secretion. In consequence of the exocytosis, the content Q of the inner membrane is expelled out of the external membrane, and the two membranes are merged together:

$$S \quad \to \quad S' \qquad\qquad S' \quad \triangleq \quad Q\ \circ\ \sigma'|\rho'(\!|\ P\ \circ\ R\ |\!) \tag{19}$$

The expression $\sigma'|\rho'$ denotes the parallel composition of branes, where the '$|$' operator is kept distinct from the '$\circ$' operator for the parallel composition of systems. The peculiar behaviour of exocytosis preserves *bitonality*, that is the parity of the level of nesting of each Brane process w.r.t. the tree structure of membranes. In fact, consider the level of nesting of Q before and after the reduction: if we match such level with the number of membranes that surround a given process, then Q passes from level 2 to level 0 of nesting. Any $exo/exo^{\perp}$ operation will never move some process P from an odd level to an even one, or vice versa. This property is preserved not only by the $exo/exo^{\perp}$ action, but by all the actions of Brane.

The $phago/phago^{\perp}$ reduction causes the inverse effect of the exocytosis: an external process is engulfed by a sibling membrane by surrounding it with a portion of the membrane itself. For example, if

$$Q \quad \triangleq \quad phago.\gamma(\!| \ T \ |\!) \qquad\qquad \rho' \quad \triangleq \quad phago^{\perp}(\tau).\rho''$$

then the system S' of Expr. (19) becomes

$$S' \quad \triangleq \quad phago.\gamma(\!| \ T \ |\!) \ \circ \ \sigma'|phago^{\perp}(\tau).\rho''(\!| \ P \ \circ \ R \ |\!)$$

and can undergo the following reduction:

$$S' \quad \rightarrow \quad \sigma'|\rho''(\!| \ P \ \circ \ R \ \circ \ \tau(\!| \ \gamma(\!| \ T \ |\!) \ |\!) \ |\!)$$

The external system $\gamma(\!|T|\!)$ has been engulfed and surrounded by the membrane τ, which is thought as a portion of the original external membrane of S'.

The *pino* action is the simplest operation on membranes: it corresponds to the inward bending of a membrane which produces a new internal membrane without any content:

$$pino(\sigma).\rho(\!| \ P \ |\!) \quad \rightarrow \quad \rho(\!| \ P \ \circ \ \sigma(\!| \ |\!) \ |\!)$$

The *pino* action has no complementary $pino^{\perp}$ co-action.

In [9], two calculi are presented: the phago-exo-pino and the mate-bud-drip variants.

A study on the relative expressive power of the two variants is reported in [28], where it is shown that the phago-exo-pino calculus is strictly more expressive than the mate-bud-drip one. Therefore, only the phago-exo-pino variant is considered here.

The formal definition of Brane follows, given in terms of a reduction semantics exactly like for the previous calculi.

Definition 13. *Let $\mathcal{N}$ be a set of names on a finite alphabet, $n, m, p, \ldots \in \mathcal{N}$. The syntax of Brane is defined as*

$$\begin{array}{rcl} P, Q & ::= & \diamond \ \big| \ P \circ Q \ \big| \ !P \ \big| \ \sigma(\!|P|\!) \\ \sigma, \tau & ::= & \mathbf{0} \ \big| \ \sigma|\tau \ \big| \ !\sigma \ \big| \ a.\sigma \\ a & ::= & phago_n \ \big| \ phago_n^{\perp}(\sigma) \ \big| \ exo_n \ \big| \ exo_n^{\perp} \ \big| \ pino(\sigma) \end{array}$$

Definition 14. *The congruence relation $\equiv$ is defined as the least congruence satisfying the following rules:*

$$
\begin{array}{cc}
P \circ Q \equiv Q \circ P & \sigma|\tau \equiv \tau|\sigma \\
P \circ (Q \circ R) \equiv (P \circ Q) \circ R & \sigma|(\tau|\rho) \equiv (\sigma|\tau)|\rho \\
P \circ \diamond \equiv P & \sigma|\mathbf{0} \equiv \sigma \\
!\diamond \equiv \diamond & !\mathbf{0} \equiv \mathbf{0} \\
!(P \circ Q) \equiv !P \circ !Q & !(\sigma|\tau) \equiv !\sigma|!\tau \\
!!P \equiv !P & !!\sigma \equiv !\sigma \\
!P \equiv P \circ !P & !\sigma \equiv \sigma|!\sigma \\
\mathbf{0}(\!| \diamond |\!) \equiv \diamond &
\end{array}
$$

$$
\begin{array}{rcrc}
P \equiv Q \Longrightarrow & P \circ R \equiv Q \circ R & \sigma \equiv \tau \Longrightarrow & \sigma|\rho \equiv \tau|\rho \\
P \equiv Q \Longrightarrow & !P \equiv !Q & \sigma \equiv \tau \Longrightarrow & !\sigma \equiv !\tau \\
P \equiv Q \wedge \sigma \equiv \tau \Longrightarrow & \sigma(\!|P|\!) \equiv \tau(\!|Q|\!) & \sigma \equiv \tau \Longrightarrow & a.\sigma \equiv a.\tau
\end{array}
$$

Definition 15. *Brane semantics is given in terms of the reduction system described by the following rules:*

$$
\frac{P \to Q}{P \circ R \to Q \circ R} \qquad \frac{P \to Q}{\sigma(\!|P|\!) \to \sigma(\!|Q|\!)} \qquad \frac{Q \equiv P \quad P \to P' \quad P' \equiv Q'}{Q \to Q'}
$$

$$
\begin{array}{rcl}
phago_n.\sigma|\sigma_0(\!|P|\!) \;\circ\; phago_n^{\perp}(\rho).\tau|\tau_0(\!|Q|\!) & \to & \tau|\tau_0(\!|\rho(\!|\sigma|\sigma_0(\!|P|\!)|\!) \circ Q|\!) \\
exo_n^{\perp}.\tau|\tau_0(\!|exo_n.\sigma|\sigma_0(\!|P|\!) \circ Q|\!) & \to & P \;\circ\; \sigma|\sigma_0|\tau|\tau_0(\!|Q|\!) \\
pino(\rho).\sigma|\sigma_0(\!|P|\!) & \to & \sigma|\sigma_0(\!|\rho(\!| \diamond |\!) \circ P|\!)
\end{array}
$$

For further details on Brane calculi we refer to [9].

4.7 Modelling the insuline example in Brane

In order to allow the reader to become a little familiar with the Brane language, we now show how the insuline secretion model can be expressed in this calculus. The explicit presence in the target model of molecules and molecule channels makes unfeasible its expression in the core phago-exo-pino variant of Brane that we formalised in the previous section.

Therefore we consider an additional class of actions which are proposed in [9] in order to handle explicitly sets of molecules, reactions and cross-membrane conveyance. The grammar of processes is then extended with molecules $m_1, \ldots, m_n$ and multisets of molecules

$$p, q \triangleq m_1 \circ \ldots \circ m_k$$

Multisets of molecules can be used to define *molecular reactions*, which constitute an additional class of Brane actions and have the following shape:

$$p_1(p_2) \rightrightarrows q_1(q_2)$$

with p_1, p_2, q_1, q_2 multisets of molecules. Like the other actions, molecular reactions are thought to be located on membranes and are part of branes. p_1 and p_2 represent the multisets of molecules which must be present outside and inside the membrane respectively, in order to allow the molecular reaction to occur. The multiset q_1 represents the molecules released outside the membrane after the reaction, while q_2 the molecules released inside it.

A cross-membrane channel like the calcium channel of Fig. 1 can be easily modelled as follows:

$$S \quad \triangleq \quad CA^{2+} \quad \circ \quad CA^{2+} \quad \circ \quad !CA^{2+}() \rightrightarrows (CA^{2+})(\!| \; P \; |\!)$$

The molecular reaction $!CA^{2+}() \rightrightarrows (CA^{2+})$ is enabled independently of the molecules present inside the membrane, and requires (at least) one molecule of CA^{2+} to be located outside of it. After the reaction, the molecule is released inside the membrane while disappears outside of it:

$$S \quad \rightarrow \quad CA^{2+} \quad \circ \quad !CA^{2+}() \rightrightarrows (CA^{2+})(\!| \; CA^{2+} \; \circ \; P \; |\!)$$

The chemical reaction

$$m_1 + m_2 \quad \rightarrow \quad m_3 + m_4$$

where the molecules m_1, m_2 appear as reactants and m_3, m_4 as products of the reaction, may be represented in Brane by the following action if we model the reaction as happening on the outer surface of the membrane:

$$m_1 \circ m_2() \rightrightarrows m_3 \circ m_4()$$

or

$$(m_1 \circ m_2) \rightrightarrows (m_3 \circ m_4)$$

if we want it to be located on the inner surface.

The explicit handling of molecular reactions in the calculus allows to model the processes GLU, ATP, ADP, PYR, CA, K and INS as simple molecules. Their behaviour is completely passive and specified by the molecular reactions present in the other processes.

The cross-membrane channel $GLUCHAN$, whose function is the conveyance of glucose inside the cell, is expressed straightforwardly:

$$GLUCHAN \quad \triangleq \quad !GLU() \rightrightarrows (GLU)$$

Glycolysis can be modelled as molecular reaction occurring inside the cell membrane:

$$GLYCOL \quad \triangleq \quad !(GLU \circ ADP) \rightrightarrows (ATP \circ PYR)$$

The encoding of the potassium channel requires it to be deactivated after its binding to some ATP molecule. The simplest way to obtain this behaviour with a single molecular reaction is the introduction of a fictitious catalyst $KCHANCAT$

present inside the cell, which allows the channel process $KCHAN$ to move the potassium ions across the membrane as soon as it does not disappear:

$$KCHAN \quad \triangleq \quad !\,(KCHANCAT \circ K) \rightrightarrows K(KCHANCAT)$$

Such catalyst disappears in consequence of the presence of ATP:

$$KCHANINH \quad \triangleq \quad !\,(KCHANACT, ATP) \rightrightarrows ()$$

The process of polarisation of the membrane POL can be modelled as molecular reaction as well:

$$POL \quad \triangleq \quad (K) \rightrightarrows (ACT)$$

ACT is another fictitious molecule which activates the calcium channel:

$$CACHAN \quad \triangleq \quad (ACT) \rightrightarrows ().!\,CA() \rightrightarrows (CA)$$

The exocytosis of the vesicles is now rendered straightforwardly by the $exo/exo^{\perp}$ reduction:

$$VES \quad \triangleq \quad exo(\!|\; INS \circ \cdots \circ INS |\!)$$

Obviously, on the external membrane there must be some process ready to execute the complementary action:

$$DOCKP \quad \triangleq \quad (CA) \rightrightarrows ().exo^{\perp}$$

Finally, the system can be expressed as follows:

$$\begin{aligned} SYS \quad \triangleq \quad & GLU \circ K \circ CA \circ \\ & \sigma(\!|\; KCHANACT \circ K \circ ADP \circ ATP \circ CA \circ VES \;|\!) \end{aligned}$$

with

$$\begin{aligned} \sigma \quad \triangleq \quad & GLUCHAN \mid GLYCOL \mid KCHAN \mid KCHANINH \mid \\ & POL \mid CACHAN \mid DOCKP \end{aligned}$$

The addition of molecular reactions makes Brane a powerful language for biological modelling. This is evident in the formalisation of the previous example, which perfectly suited the bitonal properties of Brane. Anyway, the expression of behaviours that are not included in the design of the calculus (like the calcium channel $CACHAN$, or other compartment-related operations which do not preserve bitonality) requires either the extension of the language with additional primitives or the introduction of intermediate, fictitious elements which may lead again to the same atomicity problems encountered during the modelling with π-Calculus or BioAmbients.

4.8 Encoding Brane into $\pi@$

Like ambients, membranes are organised in tree structures: each node of the tree may contain membrane processes or nested membranes. Unlike BioAmbients, Brane Calculi present two main entities: systems and branes. Their distinction implies slightly different translations, because the encoding function of systems needs only two parameters (K, the set corresponding to the bang operators in front of the system and pc, the name representing the parent compartment) while an additional parameter is needed for encoding branes (c, the name of the compartment where the brane process resides). In fact, branes represent the boundaries of compartments: each new membrane corresponds to the definition of a new compartment, i.e. the name of the immediate compartment where the associated $\pi@$ process is located. Consequently, the function $[\![\, \cdot \,]\!]^{\beta}$ from Brane to $\pi@$ has two formal parameters when applied to the parallel composition of systems

$$[\![\, P \circ Q \,]\!]^{\beta}_{K,pc} \quad \triangleq \quad [\![\, P \,]\!]^{\beta}_{K,pc} \mid [\![\, Q \,]\!]^{\beta}_{K,pc}$$

and three for the parallel composition of branes:

$$[\![\, \sigma \mid \rho \,]\!]^{\beta}_{K,c,pc} \quad \triangleq \quad [\![\, \sigma \,]\!]^{\beta}_{K,c,pc} \mid [\![\, \rho \,]\!]^{\beta}_{K,c,pc}$$

Their encoding is almost the same as the encoding of parallel composition of BioAmbients processes. The names c and pc play the same role of a and pa for the encoding function $[\![\, \cdot \,]\!]^{\alpha}$. The rise of the additional name c occurs after the first application of $[\![\, \cdot \,]\!]^{\beta}$ to branes:

$$[\![\, \sigma(\!|P|\!) \,]\!]^{\beta}_{K,pc} \quad \triangleq \quad (\nu\ c)([\![\, \sigma \,]\!]^{\beta}_{K,c,pc} \mid [\![\, P \,]\!]^{\beta}_{K,c})$$

The encoding of a membrane corresponds to the introduction of the new name c that is used as compartment for branes, and as parent compartment for the inner system P. Any encoded brane occurring in $[\![\, P \,]\!]^{\beta}_{K,c}$ will use the above name c as parent compartment and will be able to interact over it with the surrounding branes represented here by $[\![\, \sigma \,]\!]^{\beta}_{K,c,pc}$. The name pc of the parent compartment is known only by the branes placed in the outer membrane, in accordance with the same intuition of compartment nesting exploited for the encoding of ambients.

The basic actions $exo/exo^{\perp}$, $phago/phago^{\perp}$ and $pino$ are encoded almost like BioAmbients capabilities: each operation of the original language is translated with a synchronisation followed by a sequence of high priority actions which manage the reorganisation of the tree structure and the unfolding of replicated processes involved in the computation.

For example, the translation of the $exo/exo^{\perp}$ actions requires three names for each channel and triggers the $BCAST$ loop whose definition is exactly the same as for $[\![\ ]\!]^{\alpha}$:[2]

$$[\![\ exo_n.\sigma\ ]\!]^{\beta *}_{K,c,pc} \quad \triangleq \quad exo@n@pc(x).\underline{\underline{\overline{bcast}}}\langle exo, c, x\rangle.([\![\ \sigma\ ]\!]^{\beta}_{\emptyset,pc,x} \mid \Pi_K) \tag{20}$$

$$[\![\ exo^{\perp}_n.\sigma\ ]\!]^{\beta *}_{K,c,pc} \quad \triangleq \quad \overline{exo@n@c}\langle pc\rangle.([\![\ \sigma\ ]\!]^{\beta}_{\emptyset,c,pc} \mid \Pi_K) \tag{21}$$

The asymmetry of the situation reminds the encoding of the *exit*/*expel* capability, since the internal process listens on the channel $exo@n@pc$ intuitively located in the parent compartment pc, while the external process is ready to send on the local channel $exo@n@c$. The main difference is in the choice and use of the transmitted name pc by the $TREE$ subprocess and in the compartment where the broadcast effect of $BCAST$ occurs. In order to understand the rearrangement of the membrane tree structure, consider the following example:

$$S \triangleq \underbrace{\cdots \circ \overbrace{\sigma(\!|\ \underbrace{\tau(\!|\ P\ |\!)}_{c_3}\ |\!)}^{c_2}}_{c_1} \qquad \sigma \triangleq exo^{\perp}.\sigma'|\rho \qquad \tau \triangleq exo.\rho'|\gamma$$

c_1 represents the parent compartment of the whole system S, while c_2 is both the compartment of σ and the parent compartment of τ (and of the entire subsystem $\tau(\!|\ P\ |\!)$). The system S can reduce as follows:

$$S \quad \rightarrow \quad S' \qquad\qquad S' \triangleq \underbrace{\cdots \circ P \circ \overbrace{\sigma'|\rho|\rho'|\gamma(\!|\ \cdot\ |\!)}^{c_2}}_{c_1}$$

Two kinds of structural changes can be noticed in the reduction from S to S'. First, the parent compartment of P after the reduction coincides with the parent compartment c_1 of S: this justifies the name pc transmitted by the encoded $exo^{\perp}$ process. Second, the branes ρ' and γ changed both their compartment and their parent compartment. In fact, before the reduction, they were respectively c_3 and c_2, while after the reduction they become c_2 and c_1. This means that the corresponding $\pi@$ processes must replace their previous compartment name c_3 by their parent compartment c_2, and use as new parent compartment the name c_1 sent by the process performing the $[\![\ exo^{\perp}\ ]\!]^{\beta}$ action. The recursive call $[\![\ \sigma\ ]\!]^{\beta}_{\emptyset,pc,x}$ of Expr. (20) reflects this behaviour, as well as the definition of the $TREE$ process:

$$\begin{aligned} TREE(b, c, pc) \quad \triangleq \quad & \underline{\underline{exo@pc}}(x).\underline{\overline{b}}\langle c, x\rangle\ + \\ & \underline{exo@c}(x).\underline{\overline{b}}\langle pc, x\rangle\ + \\ & \underline{\underline{pc@c}}(x).\underline{\overline{b}}\langle c, x\rangle \end{aligned}$$

---

[2] Also for this encoding, we use two mutually recursive functions $[\![\ ]\!]^{\beta}_{K,c,pc}$ and $[\![\ ]\!]^{\beta *}_{K,c,pc}$, whose precise role will be clear later on.

The first branch of the choice is followed, in the previous example, by the subsystem P which receives the name of its new parent compartment. The second branch is followed by the brane γ which receives the name of its new parent compartment but replace also the name of its immediate compartment by the name of the previous parent compartment.

The third branch is needed for the encoding of the $phago/phago^{\perp}$ reduction:

$$\begin{aligned}
\llbracket\, phago_n.\sigma \,\rrbracket^{\beta *}_{K,c,pc} &\triangleq phago@n@pc(x).\underline{\overline{bcast}}\langle pc, c, x\rangle.(\llbracket\, \sigma \,\rrbracket^{\beta}_{\emptyset,c,x} \mid \varPi_K) \\
\llbracket\, phago^{\perp}_n(\rho).\sigma \,\rrbracket^{\beta *}_{K,c,pc} &\triangleq (\nu\, x)\big(\overline{phago@n@pc}\langle x\rangle. \\
&\qquad (\llbracket\, \sigma \,\rrbracket^{\beta}_{\emptyset,c,pc} \mid \llbracket\, \rho \,\rrbracket^{\beta}_{\emptyset,x,c} \mid \varPi_K)\big)
\end{aligned}$$

In this case, the name transmitted by the $\llbracket\, phago^{\perp} \,\rrbracket^{\beta}$ process corresponds to the new created membrane that surrounds the engulfed process and is used as new parent compartment.

The *pino* action has a similar, but simpler encoding:

$$\llbracket\, pino(\rho).\sigma \,\rrbracket^{\beta *}_{K,c,pc} \triangleq (\nu\, x)\tau.(\llbracket\, \sigma \,\rrbracket^{\beta}_{\emptyset,c,pc} \mid \llbracket\, \rho \,\rrbracket^{\beta}_{\emptyset,x,c} \mid \varPi_K)$$

In the right-hand side of the encoding, the reduction τ represents an invisible $\pi@$ transition, and must not be confused with the notation used for branes. The lack of complementary action and the very localised effect allows to disregard any broadcast loop. The new created membrane is represented by a new restricted name x used as local compartment by $\llbracket\, \rho \,\rrbracket^{\beta}_{\emptyset,x,c}$, whose parent compartment coincides with the immediate compartment of the encoded process $\llbracket\, pino(\rho).\sigma \,\rrbracket^{\beta}_{K,c,pc}$.

The sequential composition of branes $\sigma.\sigma'$ is encoded exactly like the choice operator of BioAmbients, as a sort of single-branch choice:

$$\begin{aligned}
\llbracket\, a.\sigma \,\rrbracket^{\beta}_{K,c,pc} &\triangleq BCAST \mid \nu\, s(!\, \underline{s}(nc, npc). \\
&\qquad (\llbracket\, a.\sigma \,\rrbracket^{\beta *}_{K,nc,npc} + TREE(s, nc, npc)) \mid \\
&\qquad \llbracket\, a.\sigma \,\rrbracket^{\beta *}_{K,c,pc} + TREE(s, c, pc))
\end{aligned}$$

The presence of two distinct replication operators in Brane leads to two slightly different encodings which reflect the fact that systems are only provided with parent compartment, while branes are also aware of their immediate compartment. The encoding of branes replication is the same as the one seen for BioAmbients replication:

$$\begin{aligned}
\llbracket\, !\, \sigma \,\rrbracket^{\beta}_{K,c,pc} &\triangleq (\nu\, b)(BANG(b, c, pc) \mid \\
&\qquad \llbracket\, \sigma \,\rrbracket^{\beta}_{K\cup\{b\},c,pc} \mid \llbracket\, \sigma \,\rrbracket^{\beta}_{K\cup\{b\},c,pc} \mid \\
&\qquad !\, \underline{\underline{new@b}}(nc, npc).\llbracket\, \sigma \,\rrbracket^{\beta}_{K\cup\{b\},nc,npc})
\end{aligned}$$

$$
\begin{array}{rcl}
BANG(b,c,pc) & \triangleq & !\,\underline{\underline{b}}(nc,npc).SUBBANG(b,nc,npc) \mid \\
& & SUBBANG(b,c,pc) \\
SUBBANG(b,nc,npc) & \triangleq & \underline{\underline{unfold@b}}.\overline{\underline{\underline{new@b}}}\langle nc,npc\rangle.\overline{\underline{\underline{b}}}\langle nc,npc\rangle + \\
& & TREE(b,nc,npc)
\end{array}
$$

The encoding of systems replication is simplified by the absence of name for the immediate compartment, which allow to shrink the definition in the following way:

$$
\begin{array}{rcl}
[\![\,!\,P\,]\!]^{\beta}_{K,pc} & \triangleq & (\nu\, b)(BANG'(b,pc) \mid \\
& & [\![\,P\,]\!]^{\beta}_{K\cup\{b\},pc} \mid [\![\,P\,]\!]^{\beta}_{K\cup\{b\},pc} \mid \\
& & !\,\underline{\underline{new@b}}(npc).[\![\,P\,]\!]^{\beta}_{K\cup\{b\},npc} \\
BANG'(b,npc) & \triangleq & !\,\underline{\underline{b}}(npc).SUBBANG'(b,npc) \mid \\
& & SUBBANG'(b,pc) \\
SUBBANG'(b,npc) & \triangleq & \underline{\underline{unfold@b}}.\overline{\underline{\underline{new@b}}}\langle npc\rangle.\overline{\underline{\underline{b}}}\langle npc\rangle + \\
& & \underline{\underline{exo@npc}}(x).\overline{\underline{\underline{b}}}\langle x\rangle
\end{array}
$$

Instead of the three branches of the $TREE$ process, only one is present, since only one of those branches (triggered by a reduction over $exo@npc$) is related to changes of the parent compartment name.

The full definition of $[\![\,\cdot\,]\!]^{\beta}$ is given in Table (3) and Table (4). Similarly to BioAmbients encoding, *oc* and *opc* are placeholders standing for the compartment and parent compartment of the outermost processes. The non-trivial encoding in $\pi@$ of molecular reactions and other extensions proposed in [9] is not strictly related to compartment semantics, but is being considered due to its relevance in chemical modelling.

Also the encoding function $[\![\,\cdot\,]\!]^{\beta}$ enjoys the requirements discussed in section 4.2:

Theorem 2. *$[\![\,\cdot\,]\!]^{\beta}$ is a reasonable encoding (modulo structural congruence), that is: let P, P_1, P_2 and ρ_1, ρ_2 be respectively Brane systems and processes, let Q be a $\pi@$ process, then*

1. $[\![\,P_1 \circ P_2\,]\!]^{\beta} = [\![\,P_1\,]\!]^{\beta} \mid [\![\,P_2\,]\!]^{\beta}$

   $[\![\,\rho_1 \mid \rho_2\,]\!]^{\beta} = [\![\,\rho_1\,]\!]^{\beta} \mid [\![\,\rho_2\,]\!]^{\beta}$
2. *for any permutation of the source names θ,* $[\![\,\theta(P)\,]\!]^{\beta} = \theta([\![\,P\,]\!]^{\beta})$;
3. $P \Downarrow$ *iff* $[\![\,P\,]\!]^{\beta} \Downarrow$, $P \Uparrow$ *iff* $[\![\,P\,]\!]^{\beta} \Uparrow$;
4. (a) *if* $P \to P_1$ *then* $\exists P_2 : P_2 \equiv P_1 \wedge [\![\,P\,]\!]^{\beta} \to^* [\![\,P_2\,]\!]^{\beta}$;

   (b) *if* $[\![\,P\,]\!]^{\beta} \to^* Q$ *then* $\exists P_1 : P \to^* P_1 \wedge Q \to^* [\![\,P_1\,]\!]^{\beta}$.

$$
\begin{aligned}
\llbracket\, \diamond \,\rrbracket^{\beta} &\triangleq \mathbf{0} \\
\llbracket\, P \circ Q \,\rrbracket^{\beta} &\triangleq \llbracket\, P \,\rrbracket^{\beta} \mid \llbracket\, Q \,\rrbracket^{\beta} \\
\llbracket\, !\, P \,\rrbracket^{\beta} &\triangleq \llbracket\, !\, P \,\rrbracket^{\beta}_{\emptyset, oc} \\
\llbracket\, \sigma(\!| P |\!) \,\rrbracket^{\beta} &\triangleq \llbracket\, \sigma(\!| P |\!) \,\rrbracket^{\beta}_{\emptyset, oc} \\
\llbracket\, \diamond \,\rrbracket^{\beta}_{K,pc} &\triangleq \mathbf{0} \\
\llbracket\, P \circ Q \,\rrbracket^{\beta}_{K,pc} &\triangleq \llbracket\, P \,\rrbracket^{\beta}_{K,pc} \mid \llbracket\, Q \,\rrbracket^{\beta}_{K,pc} \\
\llbracket\, \sigma(\!| P |\!) \,\rrbracket^{\beta}_{K,pc} &\triangleq (\nu\ c)(\llbracket\, \sigma \,\rrbracket^{\beta}_{K,c,pc} \mid \llbracket\, P \,\rrbracket^{\beta}_{K,c}) \\
\llbracket\, \mathbf{0} \,\rrbracket^{\beta}_{K,c,pc} &\triangleq \mathbf{0} \\
\llbracket\, \sigma \mid \rho \,\rrbracket^{\beta}_{K,c,pc} &\triangleq \llbracket\, \sigma \,\rrbracket^{\beta}_{K,c,pc} \mid \llbracket\, \rho \,\rrbracket^{\beta}_{K,c,pc} \\
\llbracket\, !\, P \,\rrbracket^{\beta}_{K,pc} &\triangleq (\nu\ b)(BANG'(b, pc) \mid \\
&\qquad \llbracket\, P \,\rrbracket^{\beta}_{K\cup\{b\},pc} \mid \llbracket\, P \,\rrbracket^{\beta}_{K\cup\{b\},pc} \mid \\
&\qquad !\, \underline{\underline{new@b}}(npc).\llbracket\, P \,\rrbracket^{\beta}_{K\cup\{b\},npc}) \\
\llbracket\, !\, \sigma \,\rrbracket^{\beta}_{K,c,pc} &\triangleq (\nu\ b)(BANG(b, c, pc) \mid \\
&\qquad \llbracket\, \sigma \,\rrbracket^{\beta}_{K\cup\{b\},c,pc} \mid \llbracket\, \sigma \,\rrbracket^{\beta}_{K\cup\{b\},c,pc} \mid \\
&\qquad !\, \underline{\underline{new@b}}(nc, npc).\llbracket\, \sigma \,\rrbracket^{\beta}_{K\cup\{b\},nc,npc}) \\
BANG(b, c, pc) &\triangleq !\, \underline{\underline{b}}(nc, npc).SUBBANG(b, nc, npc) \mid \\
&\qquad SUBBANG(b, c, pc) \\
BANG'(b, npc) &\triangleq !\, \underline{\underline{b}}(npc).SUBBANG'(b, npc) \mid \\
&\qquad SUBBANG'(b, pc) \\
SUBBANG(b, nc, npc) &\triangleq \underline{\underline{unfold@b}}.\overline{\underline{\underline{new@b}}}\langle nc, npc\rangle.\overline{\underline{\underline{b}}}\langle nc, npc\rangle + \\
&\qquad TREE(b, nc, npc) \\
SUBBANG'(b, npc) &\triangleq \underline{\underline{unfold@b}}.\overline{\underline{\underline{new@b}}}\langle npc\rangle.\overline{\underline{\underline{b}}}\langle npc\rangle + \\
&\qquad \underline{\underline{exo@npc}}(x).\overline{\underline{\underline{b}}}\langle x\rangle \\
TREE(b, nc, npc) &\triangleq \underline{\underline{npc@nc}}(x).\overline{\underline{\underline{b}}}\langle nc, x\rangle + \underline{\underline{exo@npc}}(x).\overline{\underline{\underline{b}}}\langle nc, x\rangle + \\
&\qquad \underline{\underline{exo@nc}}(x).\overline{\underline{\underline{b}}}\langle npc, x\rangle
\end{aligned}
$$

Table 3. Encoding of Brane processes into $\pi@$

$$
\begin{aligned}
[\![\, a.\sigma \,]\!]^{\beta}_{K,c,pc} &\triangleq BCAST \mid (\nu\, s)(!\, \underline{\underline{s}}(nc, npc). \\
&\qquad ([\![\, a.\sigma \,]\!]^{\beta *}_{K,nc,npc} + TREE(s, nc, npc)) \mid \\
&\qquad [\![\, a.\sigma \,]\!]^{\beta *}_{K,c,pc} + TREE(s, c, pc)) \\
[\![\, phago_n.\sigma \,]\!]^{\beta *}_{K,c,pc} &\triangleq phago@n@pc(x).\overline{\underline{\underline{bcast}}}\langle pc, c, x\rangle.([\![\, \sigma \,]\!]^{\beta}_{\emptyset,c,x} \mid \Pi_K) \\
[\![\, phago^{\perp}_n(\rho).\sigma \,]\!]^{\beta *}_{K,c,pc} &\triangleq (\nu\, x)\big(\overline{phago@n@pc}\langle x\rangle.([\![\, \sigma \,]\!]^{\beta}_{\emptyset,c,pc} \mid [\![\, \rho \,]\!]^{\beta}_{\emptyset,x,c} \mid \Pi_K)\big) \\
[\![\, exo_n.\sigma \,]\!]^{\beta *}_{K,c,pc} &\triangleq exo@n@pc(x).\overline{\underline{\underline{bcast}}}\langle exo, c, x\rangle.([\![\, \sigma \,]\!]^{\beta}_{\emptyset,pc,x} \mid \Pi_K) \\
[\![\, exo^{\perp}_n.\sigma \,]\!]^{\beta *}_{K,c,pc} &\triangleq \overline{exo@n@c}\langle pc\rangle.([\![\, \sigma \,]\!]^{\beta}_{\emptyset,c,pc} \mid \Pi_K) \\
[\![\, pino(\rho).\sigma \,]\!]^{\beta *}_{K,c,pc} &\triangleq (\nu\, x)\tau.([\![\, \sigma \,]\!]^{\beta}_{\emptyset,c,pc} \mid [\![\, \rho \,]\!]^{\beta}_{\emptyset,x,c} \mid \Pi_K) \\
\\
\Pi_K &\triangleq \overline{\underline{\underline{unfold@k_1}}} \mid \cdots \mid \overline{\underline{\underline{unfold@k_n}}} \quad , \\
&\qquad K = \{k_1, \ldots, k_n\} \\
BCAST &\triangleq\ !\, \underline{\underline{bcast}}(x, y, z).(\overline{\underline{\underline{x@y}}}\langle z\rangle.\overline{\underline{\underline{bcast}}}\langle x, y, z\rangle + \underline{\tau})
\end{aligned}
$$

Table 4. Encoding of Brane actions into $\pi @$

4.9 Encoding Brane into core-π@

In the encodings of BioAmbients and Brane previously defined, we used three priority levels and up to three names for each channel in π@. In order to provide similar encodings in core-π@, we need now to use not more than two names for channel and two priority levels.

The expedient for reducing the number of names needed for encoding capabilities and actions is very simple: for example, the $exo_n/exo_n^{\perp}$ action can be encoded with the only name exo_n (where the underscore character '_' is used just for sake of clarity and is part of the string representing the name) instead of the two names exo and n joint by polyadic synchronisation as $exo@n$. Anyway, the (unique) consequence of this change in the encoding is the multiplication of the names needed for representing each private name in BioAmbients. In fact, each BioAmbients name n must now be translated as a vector

$$(enter\_n, expel\_n, merge\_n, local\_n, s2s\_n, p2c\_n, c2p\_n)$$

in order to allow each different communication or capability to operate independently. Since Brane lacks restriction and name passing, its encoding in π@ is not affected by such conversion.

The reduction of the number of levels of priority influences the encoding of both languages. In the definition of the previous encoding functions, one prioritised intermediate level was used to "garbage-collect", by means of a prioritised

τ action, the $BCAST$ processes which terminated the loop with broadcast-like effect. Here we recall the definition of $BCAST$:

$$BCAST \quad \triangleq \quad !\,\underline{\underline{bcast}}(x, y, z).(\overline{\underline{\underline{\underline{x@y}}}}\langle z\rangle.\overline{\underline{\underline{bcast}}}\langle x, y, z\rangle + \underline{\tau})$$

A naive way to reduce the number of priority levels would be to decrease the priority level of both the τ transition (which would become a normal low priority transition) and all the other high priority reductions (which would still be prioritised). The definition of the $BCAST$ process would be consequently modified as follows:

$$BCAST \quad \triangleq \quad !\,\underline{bcast}(x, y, z).(\overline{\underline{x@y}}\langle z\rangle.\overline{\underline{bcast}}\langle x, y, z\rangle + \tau)$$

The problem with this definition of $BCAST$ is that there is no way to know *when* the above τ transition will be executed and the loop will terminate its action. In principle this would not constitute an important issue, since the high priority of the reductions over $x@y$ and $bcast$ would force the loop to be interrupted when all the processes waiting for a possible broadcast on the corresponding channel have been notified of the new compartment or ambient name. Unfortunately, a spurious process

$$SP \quad \triangleq \quad \overline{\underline{x@y}}\langle z\rangle.\overline{\underline{bcast}}\langle x, y, z\rangle + \tau$$

would still be around, ready to interfere with the normal behaviour of the system over some channel $x@y$. Such interference may happen for example in the encoding of the following Brane system:

$$S \quad \triangleq \quad \overbrace{\underbrace{phago.\sigma|\sigma'(\!|\ P\ |\!)}_{c_2} \circ \underbrace{phago^{\perp}(exo.\gamma).exo^{\perp}.\rho(\!|\ Q\ |\!)}_{c_3}}^{c_1}$$

c_1 represents the name of the parent compartment of the whole system S, c_2 and c_3 are the names corresponding to the immediate compartments of its two subsystems. S undergoes the following reductions:

$$S \quad \rightarrow \quad S' \quad \rightarrow \quad S''$$

with

$$S' \quad \triangleq \quad exo^{\perp}.\rho(\!|\ \overbrace{exo.\gamma(\!|\ \underbrace{\sigma|\sigma'(\!|\ P\ |\!)}_{c_2}\ |\!)}^{c_4} \circ\ Q\ |\!)$$

and

$$S'' \quad \triangleq \quad \overbrace{\underbrace{\sigma|\sigma'(\!|\ P\ |\!)}_{c_2} \circ \rho|\gamma(\!|\ Q\ |\!)}^{c_1}$$

After the first reduction, a fresh membrane is represented by the name c_4 which is then lost after the following exo operation. The $BCAST$ loop triggered by the encoding of the first reduction would be spawned by the process performing the *phago* action with parameters c_1, c_2, c_4, in order to make all the branes in σ' replace their parent compartment c_1 by the fresh compartment c_4. Such $BCAST$ loop may leave the following spurious process:

$$SP \quad \triangleq \quad \overline{\underline{c_1}@c_2}\langle c_4\rangle.\overline{\underline{bcast}}\langle c_1, c_2, c_4\rangle + \tau$$

The use of three priority levels, with an intermediate prioritised level for τ, would have forced SP to disappear before the transition

$$[\![\, S' \,]\!]^{\beta} \rightarrow [\![\, S'' \,]\!]^{\beta}$$

while with only two levels it may be still present in parallel composition with $[\![\, S'' \,]\!]^{\beta}$. This means that the subprocess $[\![\, \sigma|\sigma'(\!|\, P \,|\!) \,]\!]^{\beta}$ of the encoded system S'' may be still subject to the effect of SP and be moved again into the parent compartment identified by c_4, even if such compartment is not present anymore in the original Brane process.

In order to avoid this annoying side effect, we must ensure that any residual SP process is totally isolated from the the system. This can be achieved by using two names c, c_b for each compartment, c for the low-priority synchronisation of processes corresponding to actions, communications and capabilities, while c_b for the high-priority sequence of broadcast actions. After each broadcast, the name c_b is forgotten completely by all the processes and replaced by a new name c'_b for the next broadcast. Consequently, each SP process remaining after the regular end of the broadcasting loop is not able to communicate with the system, so it is going to disappear spontaneously by executing the internal τ action after a while. If we denote any pair of names n, n_b as $\ddot{n}$, the new encoding functions $[\![\, \cdot \,]\!]^{\beta'}$ from Brane to core-π@ and $[\![\, \cdot \,]\!]^{\alpha'}$ from BioAmbients to core-π@ have almost the same shape as the previous functions $[\![\, \cdot \,]\!]^{\beta}$ and $[\![\, \cdot \,]\!]^{\alpha}$. For example, the parallel composition of Brane systems is encoded as

$$[\![\, P \circ Q \,]\!]^{\beta'}_{K,\ddot{pc}} \quad \triangleq \quad [\![\, P \,]\!]^{\beta'}_{K,\ddot{pc}} \mid [\![\, Q \,]\!]^{\beta'}_{K,\ddot{pc}}$$

with the only difference of the names of compartments which are now paired as discussed. The definition of the processes $TREE$ and $BCAST$ reveals how name pairs are handled:

$$\begin{array}{rcl} BCAST & \triangleq & !\ \underline{bc}@bcast(x, y, ny_b, \ddot{z}). \\ & & \quad (\tau + \overline{\underline{x}}@y\langle ny_b, \ddot{z}\rangle.\overline{\underline{bc}}@bcast\langle x, y, ny_b, \ddot{z}\rangle) \\ TREE(b, \ddot{c}, \ddot{pc}) & \triangleq & \underline{pc}@c_b(nc_b, \ddot{x}).\overline{\underline{cycle}}@b\langle c, nc_b, \ddot{x}\rangle + \\ & & \underline{exo}@pc_b(npc_b, \ddot{x}).\overline{\underline{cycle}}@b\langle \ddot{c}, \ddot{x}\rangle + \\ & & \underline{exo}@c_b(nc_b, \ddot{x}).\overline{\underline{cycle}}@b\langle \ddot{pc}, \ddot{x}\rangle \end{array}$$

The presence of the names bc and $cycle$ is due to the definition of core-π@ grammar, which does not allow to denote channels by means of only one name.

The $BCAST$ process is not substantially changed, except for the number of names (five instead of three) handled during the loop: the first two identify the channel $x@y$ where the broadcast will occur, while $\ddot{z} = z, z_b$ denote the two names associated with the compartment undergoing some structural change. The name ny_b constitute the replacement of y for the next broadcast occurring in the related compartment.

Accordingly, the $TREE$ subprocess receives the three names $nc_b, \ddot{x}$ (with $\ddot{x} = x, x_b$) instead of only one. nc_b is used in the first branch of the choice as new broadcast name associated with c in substitution to c_b, in consistency with the encoded *phago* operation which updates the names of the parent compartment pc for all the processes in c. The encoded *exo* action is characterised by the peculiar property of eliminating already the name of the inner compartment where the broadcast happens, so that the new name nc_b is useless in the second and third branch of the $TREE$ process (and in fact it disappears). The same would happen with BioAmbients *merge* capability.

The encoding of actions should be consistently updated. The $phago^{\perp}$ co-action is not substantially changed:

$$\begin{aligned}
[\![\ phago^{\perp}_n(\rho).\sigma\ ]\!]^{\beta' *}_{K,\ddot{c},\ddot{pc}} \quad &\triangleq \quad (\nu\ \ddot{x})\big(\overline{phago_n}@pc\langle\ddot{x}\rangle. \\
&\qquad\qquad ([\![\ \sigma\ ]\!]^{\beta'}_{\emptyset,\ddot{c},\ddot{pc}} \mid [\![\ \rho\ ]\!]^{\beta'}_{\emptyset,\ddot{x},\ddot{c}} \mid \Pi_K\big)
\end{aligned}$$

Each action which triggers a $BCAST$ process requires the creation of the new name nc_b previously discussed:

$$\begin{aligned}
[\![\ phago_n.\sigma\ ]\!]^{\beta' *}_{K,\ddot{c},\ddot{pc}} \quad &\triangleq \quad phago_n@pc(\ddot{x}).(\nu\ nc_b) \\
&\qquad\qquad (\underline{\overline{bc}}@bcast\langle pc, c_b, nc_b, \ddot{x}\rangle.([\![\ \sigma\ ]\!]^{\beta'}_{\emptyset,c,nc_b,\ddot{x}} \mid \Pi_K)
\end{aligned}$$

nc_b is substituted for c_b here as well, as we can notice from the parameters used in the recursive call of the encoding function.

In Table 5 and Table 6 the full encoding of Brane into core-$\pi@$ is reported. The encoding of BioAmbients is based on the same ideas and does not require further insights, so it is left as exercise to the reader.

5 Conclusion and future perspectives

The language $\pi@$ has been introduced with the aim to play a pivotal role in comparing the many compartment-based. bio-inspired process calculi. Its distinguishing features is that it is a plain, conservative extension to π-calculus with the additional features of polyadic synchronisation and prioritised communication. Our claim is substantiated by the encoding of two well-known such formalisms, BioAmbients and Brane Calculi, into $\pi@$, and this is even reinforced by a recent paper [31] offering an encoding into $\pi@$ of Beta-binders [12, 13], another complex model for compartmentalised, bio-inspired behaviour.

The kernel language *core-$\pi@$* is extremely interesting because it can be provided of a stochastic semantics in a very simple and clean way. In [23, 24] the stochastic language S$\pi@$ (stochastic paillette) is proposed as a natural stochastic

$$
\begin{array}{rcl}
[\![\diamond]\!]^{\beta'} & \triangleq & \mathbf{0} \\
[\![P \circ Q]\!]^{\beta'} & \triangleq & [\![P]\!]^{\beta'} \mid [\![Q]\!]^{\beta'} \\
[\![! P]\!]^{\beta'} & \triangleq & [\![! P]\!]^{\beta'}_{\emptyset, \ddot{oc}} \\
[\![\sigma (\!| P |\!)]\!]^{\beta'} & \triangleq & [\![\sigma (\!| P |\!)]\!]^{\beta'}_{\emptyset, \ddot{oc}} \\
[\![\diamond]\!]^{\beta'}_{K, \ddot{pc}} & \triangleq & \mathbf{0} \\
[\![P \circ Q]\!]^{\beta'}_{K, \ddot{pc}} & \triangleq & [\![P]\!]^{\beta'}_{K, \ddot{pc}} \mid [\![Q]\!]^{\beta'}_{K, \ddot{pc}} \\
[\![\sigma (\!| P |\!)]\!]^{\beta'}_{K, \ddot{pc}} & \triangleq & (\nu\ \ddot{c})\big([\![\sigma]\!]^{\beta'}_{K, \ddot{c}, \ddot{pc}} \mid [\![P]\!]^{\beta'}_{K, \ddot{c}}\big) \\
[\![\mathbf{0}]\!]^{\beta'}_{K, \ddot{c}, \ddot{pc}} & \triangleq & \mathbf{0} \\
[\![\sigma \mid \rho]\!]^{\beta'}_{K, \ddot{c}, \ddot{pc}} & \triangleq & [\![\sigma]\!]^{\beta'}_{K, \ddot{c}, \ddot{pc}} \mid [\![\rho]\!]^{\beta'}_{K, \ddot{c}, \ddot{pc}} \\
[\![! P]\!]^{\beta'}_{K, \ddot{pc}} & \triangleq & (\nu\ b)\big(BANG'(b, \ddot{pc}) \mid \\
 & & [\![P]\!]^{\beta'}_{K \cup \{b\}, \ddot{pc}} \mid [\![P]\!]^{\beta'}_{K \cup \{b\}, \ddot{pc}} \mid \\
 & & !\ \underline{new}@b(n\ddot{pc}).[\![P]\!]^{\beta'}_{K \cup \{b\}, n\ddot{pc}}\big) \\
[\![! \sigma]\!]^{\beta'}_{K, \ddot{c}, \ddot{pc}} & \triangleq & (\nu\ b)\big(BANG(b, \ddot{c}, \ddot{pc}) \mid \\
 & & [\![\sigma]\!]^{\beta'}_{K \cup \{b\}, \ddot{c}, \ddot{pc}} \mid [\![\sigma]\!]^{\beta'}_{K \cup \{b\}, \ddot{c}, \ddot{pc}} \mid \\
 & & !\ \underline{new}@b(n\ddot{c}, n\ddot{pc}).[\![\sigma]\!]^{\beta'}_{K \cup \{b\}, n\ddot{c}, n\ddot{pc}}\big) \\
BANG(b, \ddot{c}, \ddot{pc}) & \triangleq & !\ \underline{cycle}@b(n\ddot{c}, n\ddot{pc}).SUBBANG(b, n\ddot{c}, n\ddot{pc}) \mid \\
 & & SUBBANG(b, \ddot{c}, \ddot{pc}) \\
BANG'(b, n\ddot{pc}) & \triangleq & !\ \underline{cycle}@b(n\ddot{pc}).SUBBANG'(b, n\ddot{pc}) \mid \\
 & & SUBBANG'(b, \ddot{pc}) \\
SUBBANG(b, n\ddot{c}, n\ddot{pc}) & \triangleq & \underline{unfold}@b().\overline{new}@b\langle n\ddot{c}, n\ddot{pc}\rangle.\overline{cycle}@b\langle n\ddot{c}, n\ddot{pc}\rangle + \\
 & & TREE(b, n\ddot{c}, n\ddot{pc}) \\
SUBBANG'(b, n\ddot{pc}) & \triangleq & \underline{unfold}@b().\overline{new}@b\langle n\ddot{pc}\rangle.\overline{cycle}@b\langle n\ddot{pc}\rangle + \\
 & & \quad \underline{exo}@npc_b(nnpc_b, \ddot{x}).\overline{cycle}@b\langle \ddot{x}\rangle \\
TREE(b, n\ddot{c}, n\ddot{pc}) & \triangleq & \underline{npc}@nc_b(nnc_b, \ddot{x}).\overline{cycle}@b\langle nc, nnc_b, \ddot{x}\rangle + \\
 & & \underline{exo}@npc_b(nnpc_b, \ddot{x}).\overline{cycle}@b\langle n\ddot{c}, \ddot{x}\rangle + \\
 & & \underline{exo}@nc_b(nnc_b, \ddot{x}).\overline{cycle}@b\langle n\ddot{pc}, \ddot{x}\rangle
\end{array}
$$

Table 5. Encoding of Brane processes into core-π@

version of *core-*π@, and provided with a generalisation of Gillespie simulation algorithm [32] to multi-compartments. Its stochastic semantics is obtained by the

$$
\begin{array}{rcl}
\llbracket\, a.\sigma \,\rrbracket^{\beta'}_{K,\ddot{c},\ddot{pc}} & \triangleq & BCAST \mid (\nu\ s)(!\ \underline{cycle}@s(n\ddot{c}, n\ddot{pc}). \\
 & & (\llbracket\, a.\sigma \,\rrbracket^{\beta'*}_{K,n\ddot{c},n\ddot{pc}} + TREE(s, n\ddot{c}, n\ddot{pc})) \mid \\
 & & \llbracket\, a.\sigma \,\rrbracket^{\beta'*}_{K,\ddot{c},\ddot{pc}} + TREE(s, \ddot{c}, \ddot{pc})) \\
\llbracket\, phago_n.\sigma \,\rrbracket^{\beta'*}_{K,\ddot{c},\ddot{pc}} & \triangleq & phago\_n@pc(\ddot{x}). \\
 & & \quad (\nu\ nc_b)(\overline{\underline{bc}}@bcast\langle pc, c_b, nc_b, \ddot{x}\rangle.(\llbracket\, \sigma \,\rrbracket^{\beta'}_{\emptyset,c,nc_b,\ddot{x}} \mid \Pi_K) \\
\llbracket\, phago^{\perp}_n(\rho).\sigma \,\rrbracket^{\beta'*}_{K,\ddot{c},\ddot{pc}} & \triangleq & (\nu\ \ddot{x})(\overline{phago\_n}@pc\langle\ddot{x}\rangle.(\llbracket\, \sigma \,\rrbracket^{\beta'}_{\emptyset,\ddot{c},\ddot{pc}} \mid \llbracket\, \rho \,\rrbracket^{\beta'}_{\emptyset,\ddot{x},\ddot{c}} \mid \Pi_K) \\
 & & \\
\llbracket\, exo_n.\sigma \,\rrbracket^{\beta'*}_{K,\ddot{c},\ddot{pc}} & \triangleq & exo\_n@pc(\ddot{x}). \\
 & & \quad (\nu\ nc_b)(\ \overline{\underline{bc}}@bcast\langle exo, c_b, nc_b, \ddot{x}\rangle.\llbracket\, \sigma \,\rrbracket^{\beta'}_{\emptyset,\ddot{pc},\ddot{x}} \mid \Pi_K) \\
\llbracket\, exo^{\perp}_n.\sigma \,\rrbracket^{\beta'*}_{K,\ddot{c},\ddot{pc}} & \triangleq & \overline{exo\_n}@c\langle\ddot{pc}\rangle.(\llbracket\, \sigma \,\rrbracket^{\beta'}_{\emptyset,\ddot{c},\ddot{pc}} \mid \Pi_K) \\
\llbracket\, pino(\rho).\sigma \,\rrbracket^{\beta'*}_{K,\ddot{c},\ddot{pc}} & \triangleq & (\nu\ \ddot{x})\tau.(\llbracket\, \sigma \,\rrbracket^{\beta'}_{\emptyset,\ddot{c},\ddot{pc}} \mid \llbracket\, \rho \,\rrbracket^{\beta'}_{\emptyset,\ddot{x},\ddot{c}} \mid \Pi_K) \\
 & & \\
\Pi_K & \triangleq & \overline{\underline{unfold}}@k_1\langle\rangle \mid \cdots \mid \overline{\underline{unfold}}@k_n\langle\rangle \\
 & & \ \text{with} \quad K = \{k_1, \ldots, k_n\} \\
BCAST & \triangleq & !\ \underline{bc}@bcast(x, y, ny_b, \ddot{z}). \\
 & & \quad (\tau + \overline{\underline{x}}@y\langle ny_b, \ddot{z}\rangle.\overline{\underline{bc}}@bcast\langle x, y, ny_b, \ddot{z}\rangle)
\end{array}
$$

Table 6. Encoding of Brane actions into core-π@

simple addition of polyadic synchronisation to the stochastic π-Calculus, thanks to the possibility of exploiting infinite rate transitions as alternative definition of (one additional level of) priority.

This opens the way to multi-compartment simulation algorithms also for all the bio-inspired calculi that can be encoded faithfully into *core-*π@, such as Brane Calculi and BioAmbients. Higher level semantic features like those considered in [33] may constitute a valid mean to facilitate the organisation of the code and increase readability and usability of such encodings.

One of the two key features of π@ is priority. However, the power of the global preemption mechanism which it is based on seems to prevent the definition of encodings that are not centralised (i.e., such that the execution can be parallelised or distributed). This seems an obstacle that cannot be overcome. In [34] we have studied the expressive power of two forms of priority (i.e., global and local) and showed that they give to π-calculus a strict increase of expressiveness, so strong that it seems impossible to identify prioritised construct easier to implement in a parallel, if not distributed, framework.

References

1. Fontana, W., Buss, L.: The Barrier of Objects: From Dynamical Systems to Bounded Organizations. In Casti, J., Karlqvist, A., eds.: Boundaries and Barriers, Addison-Wesley (1996) 56–116
2. Regev, A., Silverman, W., Shapiro, E.Y.: Representation and simulation of biochemical processes using the pi-calculus process algebra. In: Pacific Symposium on Biocomputing. (2001) 459–470
3. Milner, R., Parrow, J., Walker, D.: A calculus of mobile processes, i and ii. Inf. Comput. **100**(1) (1992) 1–77
4. Priami, C.: Stochastic pi-calculus. Comput. J. **38**(7) (1995) 578–589
5. Priami, C., Regev, A., Shapiro, E.Y., Silverman, W.: Application of a stochastic name-passing calculus to representation and simulation of molecular processes. Inf. Process. Lett. **80**(1) (2001) 25–31
6. Curti, M., Degano, P., Priami, C., Baldari, C.T.: Modelling biochemical pathways through enhanced pi-calculus. Theor. Comput. Sci. **325**(1) (2004) 111–140
7. Lecca, P., Priami, C., Laudanna, C., Constantin, G.: Predicting cell adhesion probability via the biochemical stochastic pi-calculus. In Haddad, H., Omicini, A., Wainwright, R.L., Liebrock, L.M., eds.: Symposium on Applied Computing, ACM (2004) 211–212
8. Chiarugi, D., Curti, M., Degano, P., Marangoni, R.: Vice: A virtual cell. [35] 207–220
9. Cardelli, L.: Brane calculi. [35] 257–278
10. Danos, V., Pradalier, S.: Projective brane calculus. [35] 134–148
11. Regev, A., Panina, E.M., Silverman, W., Cardelli, L., Shapiro, E.Y.: Bioambients: an abstraction for biological compartments. Theor. Comput. Sci. **325**(1) (2004) 141–167
12. Priami, C., Quaglia, P.: Beta binders for biological interactions. [35] 20–33
13. Degano, P., Prandi, D., Priami, C., Quaglia, P.: Beta-binders for Biological Quantitative Experiments. Electronic Notes in Theoretical Computer Science **164**(3) (2006) 101–117
14. Danos, V., Laneve, C.: Formal molecular biology. Theor. Comput. Sci. **325**(1) (2004) 69–110
15. Laneve, C., Tarissan, F.: A simple calculus for proteins and cells. Electr. Notes Theor. Comput. Sci. **171**(2) (2007) 139–154
16. Carbone, M., Maffeis, S.: On the expressive power of polyadic synchronisation in pi-calculus. Nord. J. Comput. **10**(2) (2003) 70–98
17. Milner, R., Parrow, J., Walker, D.: A calculus of mobile processes, i. Inf. Comput. **100**(1) (1992) 1–40
18. Milner, R., Parrow, J., Walker, D.: A calculus of mobile processes, ii. Inf. Comput. **100**(1) (1992) 41–77
19. Milner, R.: Communication and concurrency. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1989)
20. Milner, R.: The polyadic pi-calculus: a tutorial. In Bauer, F.L., Brauer, W., Schwichtenberg, H., eds.: Logic and Algebra of Specification. Springer-Verlag (1993) 203–246
21. Milner, R.: Communicating and mobile systems: the π-calculus. Cambridge University Press, New York, NY, USA (1999)
22. Cleaveland, R., Lüttgen, G., Natarajan, V.: Priority in process algebra. In Bergstra, J., Ponse, A., Smolka, S., eds.: Handbook of Process Algebra. Elsevier Science Publishers (2001) 711–765

23. Versari, C., Busi, N.: Stochastic simulation of biological systems with dynamical compartment structure. In Calder, M., Gilmore, S., eds.: CMSB. Volume 4695 of Lecture Notes in Computer Science., Springer (2007) 80–95
24. Versari, C., Busi, N.: Efficient Stochastic Simulation of Biological Systems with Multiple Variable Volumes. Electronic Notes in Theoretical Computer Science **194**(3) (2008) 165–180
25. Cardelli, L., Gordon, A.D.: Mobile ambients. In Nivat, M., ed.: FoSSaCS. Volume 1378 of Lecture Notes in Computer Science., Springer (1998) 140–155
26. Cardelli, L., Gordon, A.D.: Mobile ambients. Theoretical Computer Science **240**(1) (2000) 177–213
27. Nestmann, U., Pierce, B.C.: Decoding choice encodings. Inf. Comput. **163**(1) (2000) 1–59
28. Busi, N., Gorrieri, R.: On the computational power of brane calculi. In Priami, C., Plotkin, G.D., eds.: T. Comp. Sys. Biology. Volume 4220 of Lecture Notes in Computer Science., Springer (2006) 16–43
29. Palamidessi, C.: Comparing the expressive power of the synchronous and asynchronous pi-calculi. Mathematical Structures in Computer Science **13**(5) (2003) 685–719
30. de Boer, F.S., Palamidessi, C.: Embedding as a tool for language comparison. Inf. Comput. **108**(1) (1994) 128–157
31. Cappello, I., Quaglia, P.: A translation of beta-binders in a prioritized pi-calculus (2008) To appear.
32. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. J. Phys. Chem. **81**(25) (1977) 2340–2361
33. Kuttler, C., Lhoussaine, C., Niehren, J.: A stochastic pi calculus for concurrent objects. In Anai, H., Horimoto, K., Kutsia, T., eds.: AB. Volume 4545 of Lecture Notes in Computer Science., Springer (2007) 232–246
34. Versari, C., Busi, N., Gorrieri, R.: On the expressive power of global and local priority in process calculi. In Caires, L., Vasconcelos, V.T., eds.: CONCUR. Volume 4703 of Lecture Notes in Computer Science., Springer (2007) 241–255
35. Danos, V., Schächter, V., eds.: Computational Methods in Systems Biology, International Conference CMSB 2004, Paris, France, May 26-28, 2004, Revised Selected Papers. In Danos, V., Schächter, V., eds.: CMSB. Volume 3082 of Lecture Notes in Computer Science., Springer (2005)