Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
Electronic Notes in Theoretical Computer Science 229 (2009) 109–125
Electronic Notes in Theoretical Computer Science
www.elsevier.com/locate/entcs

A Translation of Beta-binders in a Prioritized Pi-calculus [1]

Igor Cappello[a], Paola Quaglia[a,2]

[a] *Dipartimento di Ingegneria e Scienza dell'Informazione, Università di Trento*

Abstract

A translation of Beta-binders in pi@ is presented. Beta-binders is a bio-inspired formalism that allows the modelling of processes wrapped into boundaries. No notion of enclosing compartment can instead be found in pi@, a dialect of the pi-calculus where actions are associated with a priority value driving their execution and where channel names can have a composite structure.
As recently shown, pi@ is a suitable language for encoding both Bio-Ambients and Brane Calculi, two of the most well-known formalisms for modelling biological scenarios. The translation provided here, which comes together with results about the operational correspondence of Beta-binders processes and their encodings, goes in the direction of assessing pi@ as a platform for investigating the relative expressive power of various bio-inspired languages.

Keywords: Name-passing process calculi, reduction semantics.

1 Introduction

Driven by the observation of the analogies between the behaviour of biological systems and computations of distributed communicating systems [9], many research efforts have recently gone in the direction of interpreting living entities as terms of process calculi. The foreseen gain is the ability to reason about the functionality of biological systems, and hopefully about emergent behaviours, using formal techniques and analysis tools developed over mathematically sound bases.

Investigations in this field led to the definition of a set of process calculi equipped with bio-mimetic primitives and/or bio-inspired communication paradigms. Examples are BioAmbients [8], Brane Calculi [1], and Beta-binders [4]. All of the above languages provide primitives to model compartments, as well as primitives for interacting with entities external to the local wrapper. This is in line with the observation

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

[2] Corresponding author: quaglia@disi.unitn.it

doi:10.1016/j.entcs.2009.02.008

that biological entities typically have an internal processing unit, as well as a sort of surrounding border through which the internal unit can receive and communicate signals. On the other hand, each of the various bio-inspired calculi moves at a distinct level of abstraction, and is characterized by a specific communication paradigm which makes the language suitable to easily represent some particular scenarios rather than others.

A challenging question then is whether the various bio-oriented calculi can be mutually compared and to what extent. A positive answer to this question could lead to concentrate research efforts in developing theories and tools for the referential formalism and designing automated mappings from the various languages into a suitable common platform. Another interesting issue, this time more on the linguistic side, is a fine understanding of the relationship between bio-inspired compartmentalized calculi and general purpose calculi, like, e.g., the π-calculus [2,12], which have shown suitable to model some scenarios from life science (e.g., [10,11]).

Some recent works provide a preliminary answer to the above mentioned challenges. In particular, in [13,15] a dialect of the π-calculus is used to provide semantics-preserving encodings of both BioAmbients and Brane Calculi. Such dialect, which is called $\pi@$, extends π-calculus in two ways: by prioritized communications, and by polyadic synchronizations. This last feature amounts to let channel names have a composite structure. A viable $\pi@$ channel is, e.g., $x@y$ where x and y are π-calculus names for channels. Notably, $\pi@$ comes with a stochastic extension and a simulation algorithm [14] that could be the basis of automated tools for the analysis of biological behaviours.

Here we further pursue the above research direction and use $\pi@$ as target of a semantics-preserving mapping of Beta-binders, a formalism whose communication paradigm is quite different from both the one of BioAmbients and that of Brane Calculi. In particular, Beta-binders encapsulates (extended) π-calculus processes into compartments with interaction sites, but, differently from BioAmbients and Brane Calculi, compartments cannot be explicitly nested, nor moved the one into the other. Moreover, communication between Beta-binders compartments is driven by a notion of compatibility of interaction sites, and this has no analogy with the paradigms adopted by both BioAmbients and Brane Calculi.

The rest of the paper is organized as follows. We first review the source and the target languages of the translation (Sec. 2 and Sec. 3, respectively). The actual mapping is then defined in Sec. 4, together with the results about the operational correspondence between Beta-binders processes and their encodings. The paper ends with some concluding remarks in Section 5.

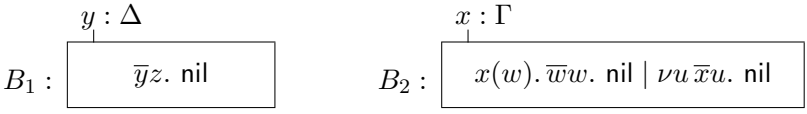
2 The source language: Beta-binders

This section reviews the definition of Beta-binders [4], a language of boxes with interaction capabilities. As in the π-calculus, the existence of a countably infinite set of names (ranged over by $x, y, z, \ldots$) is assumed. A special class of binders is used to characterize the typed interaction sites of boxes. The domain of types

is left unspecified. It can be arbitrarily instantiated under the proviso that it is decidable whether types are pairwise *compatible* or not. Processes are generated by the following grammar:

$$\begin{aligned} B &::= \mathsf{Nil} \mid \mathbb{B}[\,P\,] \mid B \parallel B \\ \mathbb{B} &::= \beta(x,\,\Gamma) \mid \beta^h(x,\,\Gamma) \mid \beta(x,\,\Gamma)\,\mathbb{B} \mid \beta^h(x,\,\Gamma)\,\mathbb{B} \\ P &::= \mathsf{nil} \mid x(w).\,P \mid \overline{x}y.\,P \mid P \mid P \mid \nu y\,P \mid !\,P \mid \\ &\qquad \mathsf{expose}(x,\,\Gamma)\,.\,P \mid \mathsf{hide}(x)\,.\,P \mid \mathsf{unhide}(x)\,.\,P \end{aligned}$$

where Nil is the deadlocked process, $\mathbb{B}[\,P\,]$ denotes the process P enclosed in a box with interaction capabilities $\mathbb{B}$, and $B_1 \parallel B_2$ is the parallel composition of B_1 and B_2. The graphical representation of the process $B_1 \parallel B_2 = \beta(y,\,\Delta)[\,\overline{y}z.\,\mathsf{nil}\,] \parallel \beta(x,\,\Gamma)[\,x(w).\,\overline{w}w.\,\mathsf{nil} \mid \nu u\,\overline{x}u.\,\mathsf{nil}\,]$ follows.



(1)

Interaction capabilities $\mathbb{B}$ are represented by sequences of elements of either the shape $\beta(x,\,\Gamma)$ or the shape $\beta^h(x,\,\Gamma)$, which are called *elementary beta binders* (unhidden and hidden, respectively). In either $\beta(x,\,\Gamma)$ or $\beta^h(x,\,\Gamma)$, the name x identifies the interaction site and is called the *subject* of the binder, while Γ is the *type* of x. No requirement is set over the domain of types $\mathcal{T}$, but for assuming that each of its possible instances comes together with the definition of a symmetric compatibility relation, and that the predicate $\mathsf{comp} : \mathcal{T} \times \mathcal{T} \to \{\mathsf{true}, \mathsf{false}\}$, which returns true iff its argument types are compatible, is decidable.

The grammar for P generates extended π-calculus processes. The deadlocked nil, as well as the input and output prefixes, and the operators for parallel composition, restriction and replication, have the same meaning as in π-calculus. The added prefixes expose, hide, and unhide are directives for changing the interaction capabilities of the enclosing box. Following the standard π-calculus terminology, x is said to be the *subject* of either the action $x(w)$ or the action $\overline{x}w$, while w is called its *object* or *parameter*. Also, the usual definitions of free names $\mathrm{fn}(\_)$, of bound names $\mathrm{bn}(\_)$, and of name substitution are extended by stipulating that $\mathsf{expose}(x,\,\Gamma)\,.\,P$ is a binder for x in P.

Notational conventions. We write $\tilde{u}$ as a shorthand for the tuple $u_1 \ldots u_n$ of names, and use $\nu\tilde{u}$ for $\nu u_1 \ldots \nu u_n$. Also, with a slight abuse of notation, we sometime read tuples as sets. The set of the subjects of all the elementary beta binders in $\mathbb{B}$ is denoted by $\mathsf{sub}(\mathbb{B})$, and we write $\mathbb{B} = \mathbb{B}_1\mathbb{B}_2$ to mean that $\mathbb{B}$ is given by the juxtaposition of $\mathbb{B}_1$ and $\mathbb{B}_2$. A binder $\mathbb{B}$ is said to be well-formed when the subjects of its elementary components are all distinct. We use $\Delta, \Delta_1, \ldots, \Gamma, \Gamma_1, \ldots$ to range over site types, and $B, B_1, \ldots$ to range of Beta-binders processes. When all the binders $\mathbb{B}_1, \mathbb{B}_2, \ldots$ occurring in B are well-formed, B itself is said to be *well-formed*. The symbol β^+ is sometimes used to stay for either β or β^h. Moreover,

$P_1 \equiv P_2$ if $P_1 \equiv_\alpha P_2$

$P \mid \mathsf{nil} \equiv P, \quad P_1 \mid P_2 \equiv P_2 \mid P_1, \quad P_1 \mid (P_2 \mid P_3) \equiv (P_1 \mid P_2) \mid P_3, \quad !\,P \equiv P \mid !\,P$

$\nu z\ \mathsf{nil} \equiv \mathsf{nil}, \quad \nu z\,\nu w\,P \equiv \nu w\,\nu z\,P, \quad \nu z\,(P_1 \mid P_2) \equiv P_1 \mid \nu z\,P_2$ provided $z \notin \mathrm{fn}(P_1)$

$\mathbb{B}\mathbb{B}'[\,P_1\,] \equiv_b \mathbb{B}'\mathbb{B}[\,P_2\,]$ if $P_1 \equiv P_2$

$B \equiv_b B'$ if $(B = \beta^+(x:\Delta)\mathbb{B}[\,P\,]$ and $B' = \beta^+(y:\Delta)\mathbb{B}[\,P\{y/x\}\,])$ or

$(B' = \beta^+(x:\Delta)\mathbb{B}[\,P\,]$ and $B = \beta^+(y:\Delta)\mathbb{B}[\,P\{y/x\}\,])$

with y fresh in P and in $\mathsf{sub}(\mathbb{B})$

$B \parallel \mathsf{Nil} \equiv_b B, \quad B_1 \parallel B_2 \equiv_b B_2 \parallel B_1, \quad B_1 \parallel (B_2 \parallel B_3) \equiv_b (B_1 \parallel B_2) \parallel B_3$

Table 1
Structural congruences $\equiv$ and $\equiv_b$.

the meta-variables $\mathbb{B}_1, \mathbb{B}_2, \ldots$ are overloaded to stay for either a beta binder or the empty string. For instance, we write $\beta(x, \Gamma)\,\mathbb{B}_1[\,P\,]$ to mean a process that could have no other interface besides x. Notice, however, that by definition each process has to have at least one (possibly hidden) interface. So, for example, $\mathbb{B}_1$ is meant to be different from the empty string in $\mathbb{B}_1[\,P\,]$. □

The operational semantics of Beta-binders makes use of both a structural congruence over pi-processes and a structural congruence over boxes. They are the smallest relations satisfying the laws in Table 1, where $\equiv_\alpha$ is used to denote α-equivalence. It is intended that the notion of α-equivalence is extended to deal with expose binders in the natural way. Also notice that the second law for $\equiv_b$ is a sort of α-conversion axiom for boxes.

The reduction relation describing the operational semantics of Beta-binders is defined by the axioms and rules collected in Tab. 2. This is actually a subset of the Beta-binders transition system. The complete semantics of the language also provides ways to join boxes together and to split a box in two. This is achieved by introducing specific axioms that can be applied when some desired conditions are satisfied. Joining and splitting activities, however, are purely semantic affairs. They have no syntactic counterpart in the language, i.e. they do not correspond to any primitive or operator. As such, joining and splitting do not seem to naturally belong to the domain of a compilation process.

The axiom intra concerns communications between pi-processes within the same box. For instance, given the process $B_1 \parallel B_2$ in (1), it allows the inference of the transition

$$B_1 \parallel B_2 \rightarrowtriangle \beta(y, \Delta)[\,\overline{y}z.\ \mathsf{nil}\,] \parallel \beta(x, \Gamma)[\,\nu u\,(\overline{u}u.\ \mathsf{nil} \mid \mathsf{nil})\,]\,.$$

The axiom inter describes possible interactions between boxes, and shows how compatibility of types is used to match complementary actions performed by parallel

$$\text{(intra)}\quad \frac{P \equiv \nu\tilde{u}\,(x(w).\,P_1 \mid \overline{x}z.\,P_2 \mid P_3)}{\mathbb{B}[\,P\,] \rightarrowtriangle \mathbb{B}[\,\nu\tilde{u}\,(P_1\{^{z}/_{w}\} \mid P_2 \mid P_3)\,]}$$

$$\text{(inter)}\quad \frac{P \equiv \nu\tilde{u}\,(x(w).\,P_1 \mid P_2) \qquad Q \equiv \nu\tilde{v}\,(\overline{y}z.\,Q_1 \mid Q_2)}{\beta(x,\,\Gamma)\,\mathbb{B}_1[\,P\,] \parallel \beta(y,\,\Delta)\,\mathbb{B}_2[\,Q\,] \rightarrowtriangle \beta(x,\,\Gamma)\,\mathbb{B}_1[\,P'\,] \parallel \beta(y,\,\Delta)\,\mathbb{B}_2[\,Q'\,]}$$

where $P' = \nu\tilde{u}\,(P_1\{^{z}/_{w}\} \mid P_2)$ and $Q' = \nu\tilde{v}\,(Q_1 \mid Q_2)$

provided $\mathrm{comp}(\Gamma, \Delta)$ and $x, z \notin \tilde{u}$ and $y, z \notin \tilde{v}$

$$\text{(expose)}\quad \frac{P \equiv \nu\tilde{u}\,(\mathsf{expose}(x,\,\Gamma)\,.\,P_1 \mid P_2)}{\mathbb{B}[\,P\,] \rightarrowtriangle \mathbb{B}\,\beta(y,\,\Gamma)[\,\nu\tilde{u}\,(P_1\{^{y}/_{x}\} \mid P_2)\,]}$$

provided $y \notin \tilde{u} \cup \mathsf{sub}(\mathbb{B}) \cup \mathrm{fn}(P_2)$

$$\text{(hide)}\quad \frac{P \equiv \nu\tilde{u}\,(\mathsf{hide}(x)\,.\,P_1 \mid P_2)}{\mathbb{B}\,\beta(x,\,\Gamma)[\,P\,] \rightarrowtriangle \mathbb{B}\,\beta^{h}(x,\,\Gamma)[\,\nu\tilde{u}\,(P_1 \mid P_2)\,]}$$

provided $x \notin \tilde{u}$

$$\text{(unhide)}\quad \frac{P \equiv \nu\tilde{u}\,(\mathsf{unhide}(x)\,.\,P_1 \mid P_2)}{\mathbb{B}\,\beta^{h}(x,\,\Gamma)[\,P\,] \rightarrowtriangle \mathbb{B}\,\beta(x,\,\Gamma)[\,\nu\tilde{u}\,(P_1 \mid P_2)\,]}$$

provided $x \notin \tilde{u}$

$$\text{(redex)}\quad \frac{B \rightarrowtriangle B'}{B \parallel B'' \rightarrowtriangle B' \parallel B''} \qquad \text{(struct)}\quad \frac{B \equiv_b B_1 \qquad B_1 \rightarrowtriangle B_2 \qquad B_2 \equiv_b B'}{B \rightarrowtriangle B'}$$

Table 2
Beta-binders semantics.

Beta-binders processes. For example, assuming $\mathrm{comp}(\Gamma, \Delta)$, for the parallel composition in (1) we have:

$$B_1 \parallel B_2 \rightarrowtriangle \beta(y,\,\Delta)[\,\mathsf{nil}\,] \parallel \beta(x,\,\Gamma)[\,\overline{x}z.\ \mathsf{nil} \mid \nu u\,\overline{x}u.\ \mathsf{nil}\,]\,.$$

The axioms expose, hide, and unhide cause the modification of the interaction capabilities of the box at hand by, respectively, adding a new site to it, hiding an active site, and unhiding an inactive site.

3 The target language: $\pi@$

The language $\pi@$ [13] extends the π-calculus [2] in two ways: *(i)* by prioritized communications, and *(ii)* by polyadic synchronizations. An integer k, indicating the priority of an action, is associated to every channel, and communication channels can be addressed by *@-names*. These are strings μ of the shape $x_1@\ldots@x_n$ where $x_1, \ldots, x_n$ are the usual π-calculus names. Processes are given by the following

$$\text{(tau)}\quad \frac{\tau \notin \bigcup_{i<k} I^i(D)}{k : \tau.\, C + D \to C} \qquad \text{(async)}\quad \frac{C \to C' \quad \tau \notin \bigcup_{i<k} I^i(C \mid D)}{C \mid D \to C' \mid D}$$

$$\text{(sync)}\quad \frac{\tau \notin \bigcup_{i<k} I^i(E \mid F)}{(k : \mu(\tilde{y}).\, C + E) \mid (k : \overline{\mu}\langle \tilde{z}\rangle.\, D + F) \to C\{\tilde{z}/\tilde{y}\} \mid D}$$

$$\text{(struct)}\quad \frac{C \equiv_@ D \quad D \to D' \quad D' \equiv_@ C'}{C \to C'} \qquad \text{(res)}\quad \frac{C \to C'}{\nu x\, C \to C'}$$

Table 3
Reduction rules for $\pi@$

grammar:

$$C ::= \mathsf{nil} \;\Big|\; \Sigma_i \pi_i.\, C_i \;\Big|\; C \mid C \;\Big|\; !\, C \;\Big|\; \nu x\, C$$
$$\pi ::= k : \tau \;\Big|\; k : \mu(y_1, \ldots, y_m) \;\Big|\; k : \overline{\mu}\langle z_1, \ldots, z_j\rangle$$

where $m, j \geq 1$. We let $C, D, \ldots$ range over $\pi@$ processes. Also, as said in Sec. 2, we keep using $\tilde{y}$ and $\tilde{z}$ as shorthands for the tuples $y_1, \ldots, y_m$ and $z_1, \ldots, z_j$, and sometime read tuples as sets. For $\mu = x_1@\ldots@x_n$, the usual π-calculus notions of free and bound names are extended as follows:

$$\mathrm{fn}(k : \mu(\tilde{y})) = \{x_1, \ldots, x_n\}, \qquad \mathrm{fn}(k : \overline{\mu}\langle \tilde{z}\rangle) = \{x_1, \ldots, x_n\} \cup \tilde{z},$$
$$\mathrm{bn}(k : \mu(\tilde{y})) = \tilde{y}, \qquad \mathrm{bn}(k : \overline{\mu}\langle \tilde{z}\rangle) = \emptyset\ .$$

The semantics of $\pi@$, given in reduction style, makes use of the structural congruence $\equiv_@$. That relation is defined as the least congruence relation satisfying either the laws defined for $\equiv$ in Table 1 (just read C for P there) or the monoidal axioms for the choice operator. Notice, however, that by $C \equiv_\alpha D$ we mean the natural adaptation of the usual notion of α-conversion to @-names. For instance, $\nu x(k : \overline{x@y}\langle z\rangle.\ \mathsf{nil}) \equiv_\alpha \nu w(k : \overline{w@y}\langle z\rangle.\ \mathsf{nil})$.

The definition of the operational semantics of $\pi@$-processes is also based on the function $I^k(C)$ which returns the set of the (relevant components of the) actions with priority k which are enabled in C. Function $I^k(C)$ is inductively defined below, where the set $T^k(C \mid D)$ is not empty iff there exists $\alpha \in I^k(C)$ such that $\alpha \neq \tau$ and $\overline{\alpha} \in I^k(D)$ (after setting, as usual, $\overline{\overline{\mu}} = \mu$). When it is not empty, $T^k(C \mid D) = \{\tau\}$.

$$I^k(\Sigma \pi_i.\, C_i) = \{\mu \mid \pi_i = k : \mu(\tilde{y}) \text{ for some } \tilde{y}\} \cup \{\overline{\mu} \mid \pi_i = k : \overline{\mu}\langle \tilde{z}\rangle \text{ for some } \tilde{z}\} \cup \{\tau \mid \pi_i = k : \tau\}$$
$$I^k(\nu y\, C) = I^k(C) \setminus \{\alpha \mid (\alpha = x_1@\ldots@x_n \vee \alpha = \overline{x_1@\ldots@x_n}) \wedge y \in \{x_1, \ldots, x_n\}\}$$
$$I^k(C \mid D) = I^k(C) \cup I^k(D) \cup T^k(C \mid D) \qquad I^k(!\, C) = I^k(C \mid C)$$

The operational semantics of $\pi@$ is given in Table 3. It enforces the condition that reductions can take place only if: *(i)* the involved actions have the lowest

priority value w.r.t. all of the enabled actions; *(ii)* in interactions the matching input and output actions have the same priority. As for the π-calculus, the notion of substitution is an integral part of the operational semantics. Again, similarly at what happens for α-conversion, name substitutions are adapted to @-names in the natural way. For example, $(k : \overline{x@y}\langle z\rangle.\ \mathsf{nil})\{^{w}/_{x}\} = k : \overline{w@y}\langle z\rangle.\ \mathsf{nil}$.

Notational convention. In what follows, we often omit both trailing occurrences of '. nil' and unrelevant parameters of input and output actions. Also, since the forthcoming translation uses only two priority values, we use underlines to improve readability and drop the prefixing 'k :' from basic actions. In detail, high priority actions π_0 and low priority actions π_1 are generated by the following grammars:

$$\pi_0 ::= \underline{\tau} \mid \underline{\mu(\tilde{y})} \mid \underline{\overline{\mu}\langle\tilde{z}\rangle} \qquad\qquad \pi_1 ::= \tau \mid \mu(\tilde{y}) \mid \overline{\mu}\langle\tilde{z}\rangle\ .$$

□

4 Mapping Beta-binders into π@

Below we describe the proposed translation $[\![\_]\!]$ of Beta-binders into π@, and present two main results about the operational correspondence between the behaviour of processes of the source language and that of their encodings.

4.1 Definition of the translation

The complete definition of the translation is reported in Tab. 4. In what follows, the encoding is described in an incremental way, by focussing on the most relevant design choices step by step, up to refining the presentation into the definition shown in Tab. 4.

The main challenge of the translation is related to the fact that two distinct communication paradigms live together within Beta-binders. One of them, quite similar to that adopted in π@, although not prioritized, follows a strict policy of action complementarity. The other one, for the description of interactions between boxes, is driven by a notion of compatibility and still implements name-passing. Moreover, the boxes of the source language are essentially closed worlds, while π@ describes flat parallel processes each of which can freely interact with any of the others. So, a main issue in the design of the encoding is the concurrent handling of both complementarity-driven (hereafter called *local*) and compatibility-driven (hereafter called *global*) communications. To overcome the above issue, the definition of the translation is underpinned by the following basic ideas.

(i) Inter-communications between boxes are not rendered as atomic communications but rather mimicked by an *ad hoc* protocol. The main actor of such a protocol is a monitor process that is in charge of mediating actual communications between encoded boxes. In detail, at its highest level, the translation of B is given by:

$$[\![B]\!] = \{\!|B|\!\} \mid \Pi_{i,j}\ CH(\Gamma_i, \Gamma_j) \text{ with } \mathrm{comp}(\Gamma_i, \Gamma_j) \text{ and } \Gamma_i, \Gamma_j \text{ occurring in } B$$

$[\![B]\!] = \{\!|B|\!\} \mid \Pi_{i,j}\ CH(\Gamma_i, \Gamma_j)$ with $\mathrm{comp}(\Gamma_i, \Gamma_j)$ and Γ_i, Γ_j occurring in B

$\{\!|\,\mathsf{Nil}\,|\!\} = \mathsf{nil}$

$\{\!|B_1 \parallel B_2|\!\} = \{\!|B_1|\!\} \mid \{\!|B_2|\!\}$

$\{\!|\mathbb{B}[\,P\,]|\!\} = \nu g\, \nu \tilde{v}\, ([\![\mathbb{B}]\!]_g \mid [\![P]\!]_g \mid \Pi_{x \in \mathrm{fn}(P)}\ GSC(g,x))$ where $\tilde{v} = \mathsf{sub}(\mathbb{B})$

$GSC(g,x) = !\, n@g@x(h).\, \overline{h}\langle g \rangle$

$CH(\Gamma, \Delta) = !\, \underline{put@\Gamma(w, k_2, f_2)}.\, (\nu k_3\, \nu f_3\, \underline{\overline{get@\Delta}\langle w, k_3, f_3\rangle}.\, (\underline{k_3}.\, \underline{\overline{k_2}} + f_3.\, \overline{f_2}) + \overline{f_2})$

$[\![\mathbb{B}]\!]_g = \nu t\, ([\![\mathbb{B}]\!]^t_g \mid TM(t) \mid EH(g,t))$

$TM(t) = t \mid !\, \underline{t}.\, t$

$EH(g,t) = !\, ex@g(x, \Gamma, h).\, \nu w\, \underline{\overline{ex@g@h}\langle w \rangle}.\, ([\![\beta(w,\, \Gamma)]\!]^t_g \mid GSC(g,w))$

$[\![\beta^+(x : \Gamma)\mathbb{B}]\!]^t_g = [\![\beta^+(x : \Gamma)]\!]^t_g \mid [\![\mathbb{B}]\!]^t_g$ assuming $[\![\mathbb{B}]\!]^t_g = \mathsf{nil}$ for $\mathbb{B}$ empty

$[\![\beta(x,\, \Gamma)]\!]^t_g = \nu r\, (BH(g,t,x,\Gamma,r) \mid !\, \underline{r}.\, BH(g,t,x,\Gamma,r))$

$[\![\beta^h(x,\, \Gamma)]\!]^t_g = \nu r\, (uh@g@x.\, \overline{r} \mid !\, \underline{r}.\, BH(g,t,x,\Gamma,r))$

$BH(g,t,x,\Gamma,r) = \underline{g@g@x(w, k_1, f_1)}.\, \overline{t}.\, PUT(t,w,\Gamma,k_1,f_1,r) +$
$\overline{t}.\, GET(g,t,x,\Gamma,r) + hd@g@x.\, uh@g@x.\, \overline{r}$

$PUT(t,w,\Gamma,k_1,f_1,r) = \nu k_2\, \nu f_2\, \underline{\overline{put@\Gamma}\langle w, k_2, f_2\rangle}.\, (\underline{k_2}.\, \underline{\overline{k_1}}.\, \underline{\overline{t}}.\, \underline{\overline{r}} + f_2.\, \overline{f_1}.\, \underline{\overline{t}}.\, \underline{\overline{r}}) + \overline{f_1}.\, \underline{\overline{t}}.\, \underline{\overline{r}}$

$GET(g,t,x,\Gamma,r) = \underline{get@\Gamma(w, k_3, f_3)}.\, (\underline{\overline{g@x}\langle w \rangle}.\, \underline{\overline{k_3}}.\, \underline{\overline{t}}.\, \underline{\overline{r}} + \overline{f_3}.\, \underline{\overline{t}}.\, \underline{\overline{r}}) + \tau.\, \underline{\overline{t}}.\, \underline{\overline{r}}$

$[\![P]\!]_g = \nu l\, (\!|P|\!)^l_g$

$(\!|\mathsf{nil}|\!)^l_g = \mathsf{nil}$

$(\!|P \mid Q|\!)^l_g = (\!|P|\!)^l_g \mid (\!|Q|\!)^l_g$

$(\!|!\, P|\!)^l_g = !\, (\!|P|\!)^l_g$

$(\!|\nu x\, P|\!)^l_g = \nu x\, ((\!|P|\!)^l_g \mid LSC(g,l,x))$

$LSC(g,l,x) = !\, n@g@x(h).\, \overline{h}\langle l \rangle$

$(\!|\mathsf{hide}(x)\,.\, P|\!)^l_g = \underline{\overline{hd@g@x}}.\, (\!|P|\!)^l_g$

$(\!|\mathsf{unhide}(x)\,.\, P|\!)^l_g = \underline{\overline{uh@g@x}}.\, (\!|P|\!)^l_g$

$(\!|\mathsf{expose}(x,\, \Gamma)\,.\, P|\!)^l_g = \nu h\, \underline{\overline{ex@g}}\langle x, \Gamma, h\rangle.\, \underline{ex@g@h(x)}.\, (\!|P|\!)^l_g$

$(\!|x(w).\, P|\!)^l_g = \underline{g@x(w)}.\, (GSC(g,w) \mid (\!|P|\!)^l_g) + \underline{l@x(w)}.\, (\!|P|\!)^l_g$

$(\!|\overline{x}w.\, P|\!)^l_g = \nu b\, \nu k_1\, \nu f_1\, (C \mid !\, \underline{b}.\, C)$ where C stays for
$\nu h\, \underline{\overline{n@g@w}}\langle h \rangle.\, \underline{h(c)}.\, (\underline{\overline{c@c@x}}\langle w, k_1, f_1\rangle.\, (\underline{k_1}.\, (\!|P|\!)^l_g + f_1.\, \underline{\overline{b}}) + \underline{\overline{l@x}\langle w \rangle}.\, (\!|P|\!)^l_g + \tau.\, \underline{\overline{b}})$

Table 4
Definition of the translation.

where, with a slight abuse of notation, we use Γ_k to mean the name corresponding to type Γ_k. Each parallel subcomponent $CH(\Gamma, \Delta)$ of the monitor acts as a *compatibility handler* for the pair (Γ, Δ): it controls the inter-communications that involve an output action over a site of type Γ and an input action over a site of type Δ. Roughly, the protocol goes as follows. An output action over a site typed by Γ engages in an interaction with $CH(\Gamma, \Delta)$, for Δ non-deterministically chosen among all the types compatible with Γ. If some input action over a site typed by Δ is ready to interact with $CH(\Gamma, \Delta)$, the relevant parameter is passed to the requesting process and the simulation of the inter-communication completes successfully. The protocol session fails otherwise, and a rollback mechanism takes the system back to the pre-existing state.

(ii) Each box $\mathbb{B}[\,P\,]$ is mapped into a parallel composition which comprises a $\pi@$ process playing the role of $\mathbb{B}$, and a process playing P. More specifically:

$$\{\!|\mathbb{B}[\,P\,]|\!\} = \nu g\, \nu \tilde{v}\, ([\![\mathbb{B}]\!]_g \mid [\![P]\!]_g \mid \ldots) \text{ where } \tilde{v} = \mathsf{sub}(\mathbb{B})\ . \tag{2}$$

(The missing portion of (2) will be illustrated later on.) Above, the shared private name g can be thought of as an identifier of the box. It is used to form @-names exploited in the implementation of global communications. Orthogonally,

$$[\![P]\!]_g = \nu l\, (\!|P|\!)^l_g$$

where the new name l uniquely identifies the content of the box, and indeed l is used to form @-names involved in local communications within the box identified by g.

(iii) Process $\{\!|\mathbb{B}[\,P\,]|\!\}$ monitors the access to $\Pi_{i,j}\, CH(\Gamma_i, \Gamma_j)$ at two distinct levels.
   (a) First, (the encoding of) one single binder in $\mathbb{B}$ at a time can interact with compatibility handlers. This ensures that the inter-communication protocol cannot involve (the encoding of) input and output actions residing in the same box, like for instance in $\beta(y,\,\Gamma)\,\beta(x,\,\Delta)[\,\overline{y}z \mid x(w)\,]$.
   (b) Second, one single parallel component of $[\![P]\!]_g$ at a time can access the handler of an elementary binder. In this way, $[\![\mathbb{B}]\!]_g$ monitors the competition to inter-communications which shows, e.g., in $\beta(y,\,\Gamma)[\,\overline{y}z \mid \overline{y}u\,] \parallel \beta(x,\,\Delta)[\,x(w)\,]$.

In more detail:

$$[\![\mathbb{B}]\!]_g = \nu t\, ([\![\mathbb{B}]\!]^t_g \mid TM(t) \mid \ldots) \tag{3}$$

where

$$TM(t) = t \mid !\,\underline{t}.\, t$$

$$[\![\beta^+(x:\Gamma)\mathbb{B}]\!]^t_g = [\![\beta^+(x:\Gamma)]\!]^t_g \mid [\![\mathbb{B}]\!]^t_g \quad \text{assuming } [\![\mathbb{B}]\!]^t_g = \mathsf{nil} \text{ for } \mathbb{B} \text{ empty}$$

$$[\![\beta(x,\,\Gamma)]\!]^t_g = \nu r\, (BH(g,t,x,\Gamma,r) \mid !\,\underline{r}.\, BH(g,t,x,\Gamma,r))$$

$$[\![\beta^h(x,\,\Gamma)]\!]^t_g = \nu r\, (uh@g@x.\,\overline{r} \mid !\,\underline{r}.\, BH(g,t,x,\Gamma,r))$$

$$\begin{aligned} BH(g,t,x,\Gamma,r) = {} & \underline{g@g@x(w,k_1,f_1)}.\,\overline{t}.\, PUT(t,w,\Gamma,k_1,f_1,r) + \\ & \overline{t}.\, GET(g,t,x,\Gamma,r) + \\ & hd@g@x.\, uh@g@x.\,\overline{r} \end{aligned}$$

$$\begin{aligned} PUT(t,w,\Gamma,k_1,f_1,r) = {} & \nu k_2\, \nu f_2\, \underline{\overline{put@\Gamma}\langle w,k_2,f_2\rangle}.\, (\underline{k_2}.\,\underline{\overline{k_1}}.\,\overline{t}.\,\overline{r} + f_2.\,\overline{f_1}.\,\overline{t}.\,\overline{r}) + \\ & \overline{f_1}.\,\overline{t}.\,\overline{r} \end{aligned}$$

$$GET(g,t,x,\Gamma,r) = \underline{get@\Gamma(w,k_3,f_3)}.\,(\underline{\overline{g@x}\langle w\rangle}.\,\underline{\overline{k_3}}.\,\overline{t}.\,\overline{r} + \overline{f_3}.\,\overline{t}.\,\overline{r}) + \tau.\,\overline{t}.\,\overline{r}$$

$$CH(\Gamma,\Delta) = !\,\underline{put@\Gamma(w,k_2,f_2)}.\,(\nu k_3\, \nu f_3\, \underline{\overline{get@\Delta}\langle w,k_3,f_3\rangle}.\,(\underline{k_3}.\,\overline{k_2} + f_3.\,\overline{f_2}) + \overline{f_2})$$

Above, process $BH(g,t,x,\Gamma,r)$ is the *binder handler* for $\beta(x,\,\Gamma)$, and PUT and GET implement attempts to inter-communications driven, respectively, by an output and by an input over x. The actions complementary to either $\underline{g@g@x(w,k_1,f_1)}$ (in BH) or $\underline{\overline{g@x}\langle w\rangle}$ (in GET) are offered by $[\![P]\!]_g$. Actual interactions of BH with either $CH(\Gamma,\Delta)$ or $CH(\Delta,\Gamma)$ occur over the channel $put@\Gamma$ or over $get@\Gamma$, respectively. The third alternative component of BH

goes for handling hiding/unhiding of binders through interactions with the encoding of hiding/unhiding internal prefixes, which are defined as follows:

$$(\!|\mathsf{hide}(x)\,.\,P|\!)^l_g = \overline{hd@g@x}.\,(\!|P|\!)^l_g$$

$$(\!|\mathsf{unhide}(x)\,.\,P|\!)^l_g = \overline{uh@g@x}.\,(\!|P|\!)^l_g\ .$$

Process TM can be thought of as a *token manager* and implements the kind of monitoring we discussed in (a) above. Before executing an action over either $put@\Gamma$ or $get@\Gamma$, process BH has to grant itself the token t which is a shared resource of all of the binder handlers in $[\![\mathbb{B}]\!]^t_g$. The token is then released by means of a $\underline{\overline{t}}$ action at the end of the protocol session, which may terminate either successfully (triggering the high priority k_i actions) or unsuccessfully (triggering either the low priority f_i actions or the low priority τ action in GET). The form of monitoring commented upon in (b) is granted by the definition of $[\![\beta(x,\,\Gamma)]\!]^t_g$. For each unhidden elementary binder, one single copy of the corresponding handler BH is initially available for interaction with $[\![P]\!]_g$, while the replicated component of $[\![\beta(x,\,\Gamma)]\!]^t_g$ can be unfolded only at the execution of $\underline{\overline{r}}$, i.e. either at the end of a protocol session or when possible hiding-unhiding activities are over. Analogously, if the binder is initially hidden, then the corresponding handler BH becomes available only after the execution of an unhide directive.

On the ground of the above overview about the translation principles, we can now complete the definitions partially stated in (2) and (3). Both the two omissions relate to a component that is exploited to ensure that the inter-communication protocol is not run unproperly, i.e. for output actions with a bound object. To better comment on this point, we focus on the axiom inter of Tab. 2. If $\mathrm{comp}(\Gamma,\Delta)$, and $P \equiv \nu\tilde{u}\,(x(w).\,P_1 \mid P_2)$ and $Q \equiv \nu\tilde{v}\,(\overline{y}z.\,Q_1 \mid Q_2)$, such axiom allows to infer an execution step leading from $\beta(x,\,\Gamma)\,\mathbb{B}_1[\,P\,] \parallel \beta(y,\,\Delta)\,\mathbb{B}_2[\,Q\,]$ to

$$\beta(x,\,\Gamma)\,\mathbb{B}_1[\,\nu\tilde{u}\,(P_1\{z/w\} \mid P_2)\,] \parallel \beta(y,\,\Delta)\,\mathbb{B}_2[\,\nu\tilde{v}\,(Q_1 \mid Q_2)\,]\ .$$

This, however, can happen only under the proviso that $x, z \notin \tilde{u}$ and $y, z \notin \tilde{v}$.

To obtain a semantics preserving translation of Beta-binders into $\pi@$, we need to guarantee that a session of the inter-communication protocol is spawned only if the subjects and the objects of the involved input and output action meet the highlighted side conditions on the involved names. This is easily achieved for the conditions over the action subjects, namely to ensure that $x \notin \tilde{u}$ and $y \notin \tilde{v}$. In fact, if x is bound in P then the encoding of P cannot synchronize with the binder handler of $[\![\beta(x,\,\Gamma)]\!]^t_g$ over the channel $g@x$ (see process GET above). This will be fully clear after introducing $[\![P]\!]_g$. We can just anticipate that, since $[\![\beta(x,\,\Gamma)\,\mathbb{B}_1]\!]_g$ and $[\![P]\!]_g$ are run in parallel, under the above hypotheses the scope of x would be restricted to $[\![P]\!]_g$, leading, at best, at a composition looking like

$$\underline{\overline{g@x}\langle z\rangle}.\,C \mid \nu x\,(\underline{g@x(w)}.\,D)\ .$$

For analogous reasons, if $y \in \tilde{v}$ then the encoding of Q cannot synchronize with the binder handler of $[\![\beta(y,\,\Delta)]\!]^{t'}_{g'}$ over the channel $g'@g'@y$. Yet another one of the

four conditions on names is not problematic. This is the case for $z \notin \tilde{u}$, which is actually a non-issue, as it could be best seen in the Beta-binders labelled transition system [5]. Intuitively, if $z \in \tilde{u}$, all we have to do in order to make the inter axiom applicable is to choose a suitable α-conversion of P. Looking at this same case from the perspective of the π-calculus semantics, what happens here is that the name substitution $\{z/w\}$ has to be applied to the process $\nu\tilde{u}\, x(w).\, P_1$. Whether or not $z \in \tilde{u}$ is completely irrelevant. The name substitution takes successfully place anyhow, possibly inducing the refreshing of $\tilde{u}$ to avoid capturing z. We are now left with the fourth side condition on names: $z \notin \tilde{v}$. This is the most delicate point. Seen on the Beta-binders side, this requirement enforces the assumption that the border of boxes is the furthest limit that scope extrusion can reach. In $\pi@$, though, just as in π-calculus, there is no explicit way to prevent scope extrusion. To overcome this issue, the translation makes use of *scope handlers* that are enquired to understand whether or not a certain name can be used as argument of inter-communications. In detail:

$$GSC(g,x) = !\, n@g@x(h).\, \underline{\overline{h}\langle g\rangle}$$

$$LSC(g,l,x) = !\, n@g@x(h).\, \underline{\overline{h}\langle l\rangle}$$

Operationally, before using x as object of an inter-communication an enquiry is sent over the channel $n@g@x$, and either g or l is sent back to the requesting process depending on which kind of scope handler (*global* or *local*) is in the scope of x.

The missing portion of (2) serves the purpose of setting the proper global scope handlers at the outermost level. Precisely:

$$\{\!|\mathbb{B}[\,P\,]|\!\} = \nu g\, \nu\tilde{v}\, (\llbracket \mathbb{B} \rrbracket_g \mid \llbracket P \rrbracket_g \mid \Pi_{x\in \mathrm{fn}(P)}\, GSC(g,x)) \text{ where } \tilde{v} = \mathsf{sub}(\mathbb{B}).$$

Further scope handlers are added by the encoding of expose prefixes, restricted internal processes, and input prefixes. In the case of expose prefixes this task is carried on by the *expose handler* that completes (3) as shown below.

$$\llbracket \mathbb{B} \rrbracket_g = \nu t\, (\llbracket \mathbb{B} \rrbracket_g^t \mid TM(t) \mid EH(g,t))$$

$$EH(g,t) = !\, ex@g(x,\Gamma,h).\, \nu w\, \underline{\overline{ex@g@h}\langle w\rangle}.\, (\llbracket \beta(w,\,\Gamma) \rrbracket_g^t \mid GSC(g,w))\ .$$

The above handler is triggered by encoded expose directives:

$$(\!|\mathsf{expose}(x,\,\Gamma)\,.\,P|\!)_g^l = \nu h\, \overline{ex@g}\langle x,\Gamma,h\rangle.\, \underline{ex@g@h(x)}.\, (\!|P|\!)_g^l\ .$$

Up to structural congruence we have in fact that

$$EH(g,t) \mid (\!|\mathsf{expose}(x,\,\Gamma)\,.\,P|\!)_g^l \rightarrow^2$$

$$EH(g,t) \mid \nu w\, (\llbracket \beta(w,\,\Gamma) \rrbracket_g^t \mid GSC(g,w) \mid (\!|P|\!)_g^l\{w/x\})$$

where the global scope handler $GSC(g,w)$ goes along with the fact that w is free in $(\!|P|\!)_g^l\{w/x\}$. By contrast, the encoding of restricted processes induces the insertion of a local scope handler:

$$(\!|\nu x\, P|\!)_g^l = \nu x\, ((\!|P|\!)_g^l \mid LSC(g,l,x))\ .$$

The translation of input prefixes takes into account the fact that input activities may non-deterministically (i.e. with equal priority) be involved in either local or global communications. In the first case the actual parameter of the communication can be either free or bound within the box, but its relative scope handler in anyhow already in place. On the other hand, if the input action is fired for a successful inter-communication, then, as discussed above, the received parameter is surely free. Then, since such a parameter could be fresh w.r.t. the free names of the box, a global scope handler is added in this case. Formally:

$$(\!|x(w).\,P|\!)^l_g = \underline{g@x(w)}.\,(GSC(g,w) \mid (\!|P|\!)^l_g) + \underline{l@x(w)}.\,(\!|P|\!)^l_g \quad .$$

We eventually comment on the encoding of output prefixes:

$$(\!|\overline{x}w.\,P|\!)^l_g = \nu b\,\nu k_1\,\nu f_1\,(C \mid !\,\underline{b}.\,C) \qquad \text{where } C \text{ stays for}$$

$$\nu h\,\overline{n@g@w}\langle h\rangle.\,\underline{h(c)}.$$

$$(\overline{\underline{c@c@x}}\langle w,k_1,f_1\rangle.\,(\underline{k_1}.\,(\!|P|\!)^l_g + f_1.\,\underline{\overline{b}}) + \overline{\underline{l@x}}\langle w\rangle.\,(\!|P|\!)^l_g + \tau.\,\underline{\overline{b}})\ .$$

The first step of $(\!|\overline{x}w.\,P|\!)^l_g$ is to contact the scope handler for w. If w is a name with local scope, then c is instantiated by l and no action complementary to $\overline{\underline{l@l@x}}$ can be found in the encoded process. Hence only two alternatives are feasible for C: engaging in a local communication over $l@x$ with high priority, or silently causing a rollback to the initial configuration with low priority. On the other hand, if w is recognized to be a suitable name for global communications, then c gets instantiated by g and, with the same high priority, C can either engage in a session of the inter-communication protocol or take part into a local communication. Also, it may happen that no synchronization over either $g@g@x$ or $l@x$ can occur. This is the case, e.g., when the box has no binder named x and the internal process has no unguarded input over x. If so, then a copy of C is spawned and the above behaviour is replicated once more from the very beginning. The same rollback mechanism is triggered when, through f_1, process C receives from the handler of the binder x the notification that the session of the inter-communication protocol failed.

To conclude the presentation of the translation, a schematic overview of what is involved in the inter-communication protocol is shown in Fig. 1. There we consider a session of the protocol relative to the Beta-binders process $B_1 \parallel B_2$ graphically reported in the lower portion of Fig. 1. The upper portion of the figure shows an interconnection graph whose nodes represent the $\pi@$ processes involved. For instance, BH_1^y denotes the handler for the binder y of B_1, and BH_2^x the handler for the binder x of B_2. The edges of the graph are labelled to record two kinds of information: the relevant channel used for communications between the end-points of the edge (e.g., t_2 above the rightmost edge), and the relevant private resources shared by such end-points (e.g., (t_2) below the rightmost edge). The temporal ordering of the communications involved in the protocol session is reported in UML style in Fig. 2. In that diagram, arrows point to the receiver, and labels keep track of both communication channel and actual parameters. Notice in particular the communications between BH_1^y and $TM_1 = t_1 \mid !\,\underline{t_1}.\,t_1$, and those between BH_2^x

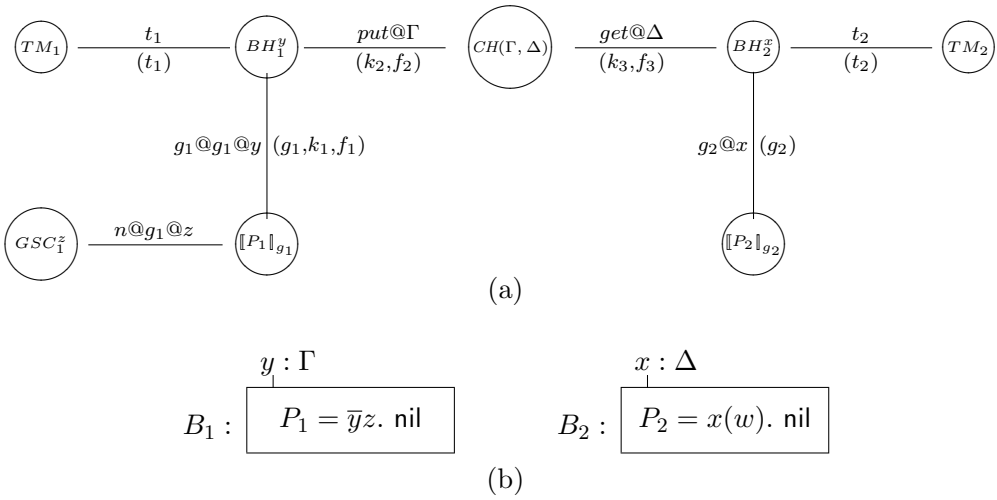


Fig. 1. Schema of the interconnection topology (a) for a specific inter-communication (b).

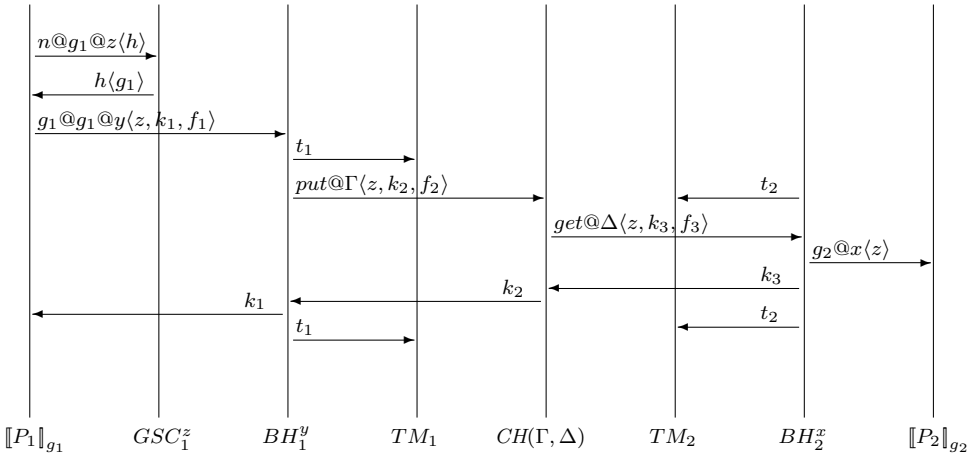


Fig. 2. Temporal ordering of communications for the inter-communication in Fig. 1.

and $TM_2 = t_2 \mid !\underline{t_2}.t_2$. Also, recall that each token manager is shared by the translations of all of the elementary binders of the given box. The communications over t_1 and t_2 show that access to $CH(\Gamma,\Delta)$ is dealt with as a critical session. This ensures that input and output actions residing within the same box (think, e.g., of $\beta(y,\Gamma)\,\beta(x,\Delta)[\,P_1 \mid P_2\,]$) cannot possibly interact via $CH(\Gamma,\Delta)$.

The layout of the diagram in Fig. 2 puts evidence on the fact that the failures driving the rollback mechanims can happen at three different levels (correspondingly to notifications over f_1, f_2, f_3).

f_1**:** If Γ were not compatible with Δ (nor with other types possibly in the system), then $CH(\Gamma,\Delta)$ would not exist and hence BH_1^y could not contact it over $put@\Gamma$. In this case, or upon receiving a higher level failure notification over f_2, the binder

handler BH_1^y would send a f_1 signal to $[\![P_1]\!]_{g_1}$.

f_2**:** If the binder $\beta(x, \Delta)$ were hidden, or the binder handler BH_2^x would not yet own the token t_2, then $CH(\Gamma, \Delta)$ could not execute its prioritized activity over the $get@\Delta$ channel. In this situation, or upon receiving over f_3, the compatibility handler would send back to BH_1^y a notification over f_2.

f_3**:** Even if BH_2^x already owns the token t_2, $[\![P_2]\!]_{g_2}$ might be unable to interact over x. For example, P_2 could be $u(w).\,\mathsf{nil}$. Under these circumstances, BH_2^x would notify the protocol failure to the compatibility handler via f_3.

Reasoning about failures, notice that the use of priorities is fundamental to provide a proper implementation of the rollback mechanism: in all of the involved processes synchronizations over the successful k_i channels have higher priority than synchronizations over the f_i channels. This ensures that failure notifications are put forward only when successful termination cannot occur.

4.2 Properties of the translation

Below we report on two main properties of the translation, which are relative to the operational correspondence between Beta-binders processes and their encodings. These properties rely on the definition of a structural congruence which extends $\equiv_@$ with the addition of the following law:

$$\nu x\,(!\,k : \mu(\tilde{y}).\,C) \mathrel{\widehat{\equiv}}_@ \mathsf{nil} \;\text{ if } x \in \mu\;. \tag{4}$$

The above axiom allows the garbage collection of possible deadlocked processes which are left as residuals in the derivative of $(\!|\overline{x}w.\,P|\!)_g^l$. Indeed, recall that the encoding of internal processes of the form $\overline{x}w.\,P$ exploits a parallel component $!\,\underline{b}.\,C$ to unfold a fresh copy of C when either $\overline{x}w$ cannot take part into a communication (neither locally nor globally) or a session of the intercommunication protocol fails. If and when a communication involving $\overline{x}w$ terminates successfully, process $!\,\underline{b}.\,C$ cannot be further unfolded and remains hanging.

Definition 4.1 The relation $\mathrel{\widehat{\equiv}}_@$ is the least congruence relation satisfying both the axioms used to define $\equiv_@$ and the garbage collection law in (4).

As expected, one step of a Beta-binders process is simulated by one or more reduction steps of its encoding.

Theorem 4.2 *If B is well-formed and $B \twoheadrightarrow B'$ then $[\![B]\!] \rightarrow^* \mathrel{\widehat{\equiv}}_@ [\![B']\!]$.*

Proof. A few auxiliary results are needed to prove the assertion. One of them is that the translation reflects structural congruence, namely that for all well-formed B and B', $B \equiv_b B'$ implies $[\![B]\!] \equiv_@ [\![B']\!]$. Below we just sketch the proof strategy of the main statement. First, by the hypothesis $B \twoheadrightarrow B'$ we deduce the possible structure of both B and B' (see [5], Lemma 4). In this way we get two processes B_1 and B_1' such that $B_1 \equiv_b B$ and $B_1' \equiv_b B'$, and which can assume five (pairs of) distinct forms, essentially depending on the axiom rule driving the derivation

of $B \twoheadrightarrow B'$. We then reason on B_1, and the proof goes on by a case analysis of its form, which in turn determines the form of the encoding $[\![B_1]\!]$. In each of the possible cases we show that, for suitable C and C',

$$[\![B_1]\!] \to C \to^* C' \mathrel{\widehat{\equiv}}_{@} [\![B_1']\!].$$

Hence, by $[\![B]\!] \equiv_{@} [\![B_1]\!]$, and the $\pi@$ structural rule, we get $[\![B]\!] \to^* C'$. Then the thesis, by $C' \mathrel{\widehat{\equiv}}_{@} [\![B_1']\!] \equiv_{@} [\![B']\!]$ which implies $C' \mathrel{\widehat{\equiv}}_{@} [\![B']\!]$. □

Theorem 4.3 below states that any reduction of $[\![B]\!]$ may either lead to a rollback, or to the encoding of a Beta-binders process reachable from B in one single step. In both cases the result holds up to structural congruence. Notice, however, that in case of rollback the garbage collection axiom is superfluous.

Theorem 4.3 *If B is well-formed and $[\![B]\!] \to C$ then there exists C' such that $C \to^* C'$ and*

- *either $C' \equiv_{@} [\![B]\!]$,*
- *or $C' \mathrel{\widehat{\equiv}}_{@} [\![B']\!]$ for some B' such that $B \twoheadrightarrow B'$.*

Proof. By definition of the translation no τ action occurs unguarded in $[\![B]\!]$. Hence the derivation of $[\![B]\!] \to C$ can only be driven by a (sync) axiom. Five distinct classes of complementary input/output actions can be involved in such synchronization. The relevant channels are:

- $ex@g$ (see $EH(g,t)$ and $(\!|\mathsf{expose}(x,\Gamma)\,.\,P|\!)^l_g$);
- $hd@g@x$ (see $BH(g,t,x,\Gamma,r)$ and $(\!|\mathsf{hide}(x)\,.\,P|\!)^l_g$);
- $uh@g@x$ (see $[\![\beta^h(x,\Gamma)]\!]^t_g$ and $(\!|\mathsf{unhide}(x)\,.\,P|\!)^l_g$);
- $n@g@w$ (see $(\!|\overline{x}w.\,P|\!)^l_g$ and either $GSC(g,w)$ or $LSC(g,l,w)$).
- t (see $TM(t)$ and $BH(g,t,x,\Gamma,r)$);

The first three kinds of channels are involved in the encoding of expose, hide, and unhide directives. In each of these cases we show that $C \to^* \equiv_{@} [\![B']\!]$ for some B' such that $B \twoheadrightarrow B'$. Hence the thesis, by $\equiv_{@} \subseteq \mathrel{\widehat{\equiv}}_{@}$.

The fourth kind of synchronization is triggered by the translation of an output prefix with actual parameter w. Specifically, suppose that the output action is $\overline{x}w$. Depending on whether w is a private resource of the box or not, the partner of the synchronization is either the local or the global scope handler for w, and a further synchronization discloses this (either l or g is received over h in $(\!|\overline{x}w.\,P|\!)^l_g$). A range of possibilities is now considered. If w is local, then with high priority the next step can be a communication over $l@x$, and with low priority it can be a τ action. (Here notice that none of the processes in Tab. 4 ever offers an input over $l@l@x$.) If instead w is global, then non-deterministically and with high priority the next reduction can be a synchronization over either $g@g@x$ or $l@x$, and with low priority it can be a τ action. In those cases when a synchronization over $l@x$ takes place we show that there exists C' such that $C \to^* C'$ and, for some B' such that $B \twoheadrightarrow B'$ and for some b and D, $C' \equiv_{@} [\![B']\!] \mid \nu b\,(!\,\underline{b}.\,D)$. Hence the thesis by $C' \mathrel{\widehat{\equiv}}_{@} [\![B']\!]$. The same is done in the case of communications over $g@g@x$ which end up in the

successful completion of a session of the inter-communication protocol (see Fig. 2 for a schematic overview of the involved reductions). When the τ alternative is taken, as well as when the inter-communication protocol is rolled-back for failure, we show instead that $C \rightarrow^* \equiv_@ [\![B]\!]$.

The fifth kind of synchronization is relative to those cases when a binder handler gets the token for accessing a compatibility handler (see t_2 in Fig. 2). Here we distinguish two main cases: either an instance of the inter-communication protocol actually takes place, or a low priority τ action forces the token be released and the binder handler be re-initialized. In the first case, be the protocol session successful or not, we get the same results discussed above for initial synchronizations over $n@g@w$. In the second case we show that $C \rightarrow^* \equiv_@ [\![B]\!]$. □

5 Concluding remarks

We presented a translation of Beta-binders into $\pi@$. This work mostly relates to the encodings of other two compartmentalized calculi, BioAmbients and Brane Calculi, into the same target language [13,15]. The three source languages largely differ in their communication paradigms. Both BioAmbients and Brane Calculi express a hierarchy of compartments and provide primitives (located within ambients or on membranes) to allow their dynamic evolution. So, the encodings presented in [13,15] mainly focus on the need to faithfully render and update the tree-like structure of compartment nesting. This is done by implementing a multicast communication mechanism which allows the timely delivery of notifications about changes of the nesting structure. As for the translation of Beta-binders, the main issue is the accurate handling of the possible interferences of the two distinct communication mechanisms that live together in the language: intra-communications and inter-communications. Then the key point is the design of appropriate interaction protocols.

In all of the three above mentioned encodings, the polyadic synchronization of $\pi@$ seems to be a convenient tool rather than a fundamental feature. In fact, although @-names bring in the flavour of a sort of partial application of name substitution, it would seem that translations of the three compartmentalized calculi could also be obtained, via more involved encodings, by using simple channel names in the π-calculus style. Things are completely different, though, when we come at the prioritized communications of $\pi@$. In the encodings of BioAmbients and Brane Calculi, priorities are mainly exploited to force the predecence of communications of structural information over all of the other communications. This, e.g., avoids that an interaction between two processes takes place if some event changed their relative positions and they are not entitled to synchronize in the updated configuration. In the case of the Beta-binders encoding, priorities are fundamental to the implementation of the transactional mechanism that allows a rollback when a session of the inter-communication protocol cannot successfully terminate. In this respect, the relevance of priorities is not surprising at all. Essentially, priorities master the proper undo of the many steps needed to simulate Beta-binders atomic

inter-communications by means of a finer grained communication paradigm.

To conclude, the translation presented in this paper, together with those of BioAmbients and Brane Calculi, goes in the direction of assessing $\pi@$ as a plausible formal ground for the investigation of the relative expressive power of compartmentalized bio-inspired languages. It goes unsaid that this kind of investigation will not be particularly easy. Indeed, the same sort of task shows to be quite demanding for encodings which are much handier than those for compartmentalized calculi (see, e.g., [16,6,7,3]).

References

[1] L. Cardelli. Brane Calculi. In V. Danos and V. Schächter, editors, *Proc. 2nd Int. Workshop on Computational Methods in Systems Biology, CMSB 2004*, volume 3082 of *LNBI*. Springer, 2005.

[2] R. Milner. *Communicating and mobile systems: the π-calculus*. Cambridge Universtity Press, 1999.

[3] U. Nestmann and B.C. Pierce. Decoding Choice Encodings. In U. Montanari and V. Sassone, editors, *Proc. 7th Int. Conference on Concurrency Theory, CONCUR 1996*, volume 1119 of *LNCS*, pages 179–194. Springer, 1996.

[4] C. Priami and P. Quaglia. Beta Binders for Biological Interactions. In V. Danos and V. Schächter, editors, *Proc. 2nd Int. Workshop on Computational Methods in Systems Biology, CMSB 2004*, volume 3082 of *LNBI*, pages 21–34. Springer, 2005.

[5] P. Quaglia. On Beta-Binders Communications. In P. Degano, R. De Nicola, and J. Meseguer, editors, *Concurrency, Graphs and Models*, volume 5065 of *LNCS*, pages 457–472. Springer, 2008.

[6] P. Quaglia and D. Walker. On synchronous and asynchronous mobile processes. In J. Tiuryn, editor, *Proc. 2nd Int. Conference on Foundations of Software Science and Computation Structures, FoSSaCS 2000*, volume 1784 of *LNCS*, pages 283–296. Springer, 2000.

[7] P. Quaglia and D. Walker. Types and full abstraction for polyadic π-calculus. *Information and Computation*, 200:215–246, 2005.

[8] A. Regev, E. M. Panina, W. Silverman, L. Cardelli, and E. Y. Shapiro. Bioambients: an abstraction for biological compartments. *Theoretical Computer Science*, 325(1):141–167, 2004.

[9] A. Regev and E. Shapiro. Cells as computations. *Nature*, 419:343, 2002.

[10] A. Regev, W. Silverman, and E. Shapiro. Representing biomolecular processes with computer process algebra: π-calculus programs of signal transduction pathways. In *American Association for Artificial Intelligence Pubblication*, 2000.

[11] A. Regev, W. Silverman, and E. Shapiro. Representation and simulation of biochemical processes using the pi-calculus process algebra. In *Proceedings of the Pacific Symposium of Biocomputing 2001*, volume 6, pages 459–470, 2001.

[12] D. Sangiorgi and D. Walker. *The π-calculus: a Theory of Mobile Processes*. Cambridge Universtity Press, 2001.

[13] C. Versari. A core calculus for a comparative analysis of bio-inspired calculi. In R. De Nicola, editor, *Proc. 16th European Symposium on Programming, ESOP 2007*, volume 4421 of *LNCS*, pages 411–425. Springer, 2007.

[14] C. Versari and N. Busi. Stochastic Simulation of Biological Systems with Dynamical Compartment Structure. In *Proc. Int. Conf. on Computational Methods in Systems Biology, CMSB 2007*, LNCS, pages 80–95. Springer, 2007.

[15] C. Versari and R. Gorrieri. $\pi@$: A pi-Based Process Calculus for the Implementation of Compartmentalised Bio-inspired Calculi. In M. Bernardo, P. Degano, and G. Zavattaro., editors, *Formal Methods for Computational Systems Biology, SFM 2008*, volume 5016 of *LNCS*, pages 449–506. Springer, 2008.

[16] N. Yoshida. Graph Types for Monadic Mobile Processes. In *Proc. 16th Conference on Foundations of Software Technology and Theoretical Comp. Sc., FST&TCS 1996*, volume 1180 of *LNCS*, pages 371–386. Springer, 1996.