

# Projective Brane Calculus

Vincent Danos<sup>1\*</sup> and Sylvain Pradalier<sup>2</sup>

<sup>1</sup> Université Paris 7 & CNRS

<sup>2</sup> ENS Cachan

**Abstract.** A refinement of Cardelli’s *brane calculus* [1] is introduced where membrane actions are directed. This modification brings the language closer to biological membranes and also obtains a symmetric set of membrane interactions. An associated structural congruence, termed the *projective* equivalence, is defined and shown to be preserved under all possible system evolutions. Comparable notions of projective equivalence can be developed in other hierarchical process calculi and might be of interest in other applications.

## 1 Introduction

An estimated third of the proteins in eukaryotic cells are membrane proteins. Fusions and fissions of membranes happen often in cells, so often in fact that the outer plasmic membrane renews itself completely in half an hour! Both figures give a sense of how intense membrane-bound trafficking is, and of how important a component of cell computing it is.

Biological membrane interactions seem worth a formal investigation, in the larger perspective of laying out formal languages to describe, simulate, combine and otherwise analyze biological systems. Living organisms are complicated organizations and while engineering is clearly needed to understand them further, it seems a formal approach might also prove useful in defining suitable levels of abstractions, workable notions of properties and observations, and appropriate tools for constructing, investigating, and refining models.

Languages specifically dealing with protein-protein and protein-DNA interactions, are already well under way [2–8]. Comparatively few are addressing membrane interactions. The idea of *membrane computing* was introduced in the lively subject of natural computing by Paūn [9], mostly in the tradition of automata and formal language theory. Another language of membranes and compartments, *bio-ambients*, following the tradition of process algebras, was proposed by Regev *et al.* [10], addressing description and simulation issues, and evolved later into the *brane calculus* proposed recently by Cardelli [1].

The brane calculus stands on its own as an elegant formalism, and does well in abstracting and describing the intricate biological processes involving membranes. Membranes are represented as nested multisets of actions, representing

---

\* *Corresponding author:* Équipe PPS, Université Paris 7 Denis Diderot, Case 7014, 2 Place Jussieu 75251 PARIS Cedex 05, Vincent.Danos@pps.jussieu.fr

the fusion and fission capabilities of the branes they sit *on*. Note that this in contrast with the original Ambient calculus, from which the brane calculus is derived, where capabilities sit *in* the ambients they control. These fission and fusion capabilities are meant to abstract actual proteins, or protein complexes, inserted in a membrane and defining the membrane potential interactions with various signals and other membranes. Membrane interactions are preserving the nesting parity, a principle which Cardelli refers to as *bitonality*, and which one observes in biological systems.

The purpose of the present paper is to incorporate another comparable principle, that of *projective invariance*. Specifically, we work out a revision of the brane calculus, as suggested by Cardelli [11], which consists in replacing actions with *directed actions*. Introducing directed actions takes the language a little step further down in the details, by actually telling whether an action, that is an interaction capability, is looking inwards or outwards. While still clearly an abstraction, the concept of directed actions meshes with what is known of membrane interactions at the molecular level.

In eukaryotic cells, new membranes patches are mainly produced from the second membrane surrounding the nucleus, called ER (Endoplasmic Reticulum). Some of them are dispatched to the outer membrane of the cell, called the plasmic membrane, with specific sugar decorations that control the correct interaction of the cell with its environment. These lipids bearing sugar signatures are first grown on the outer membrane of ER, and only then flipped by a protein known as a *flippase*, so that they face the inside of ER [12]. Once this is done, a vesicle breaks off from ER and bubbles up to the surrounding plasmic brane, with the glycolipid facing the inside of the brane. When finally the vesicle fuses with the outer membrane, its own membrane is flipped inside-out, and the glycolipid faces the outside, as it should to be functional.

A similar case, is the production of transmembrane proteins designed to be receptors on the outer membrane. These proteins are inserted in ER during their translation from RNA, with their specific receptor domain facing inward, so that when they are dispatched to their final destination, they ultimately face the outer solution.

Both examples, show that biological membrane interactions operate under the strong constraint that molecular implementations of actions are directed. It seems therefore natural to enrich the original brane-calculus in order to represent them, obtaining a more accurate description of biological membranes. One could fear that the resulting calculus could become complicated, but it turns out to be structurally simpler. Indeed, the constraint generated by the fact that actions are directed reflects into the calculus as a new invariance principle, which we call *projective invariance*.

The contribution of this paper is to define properly this modified brane calculus, called the *projective brane calculus*, and to give a rigorous definition and proof of projective invariance. A few key events in protein trafficking, such as fusions, sorting, tagging and routing are then described. All events are integrated

in an example of a retrovirus infecting a cell, using the cell synthesis chain for its own replication, and then escaping.

*Outline.* The paper is organized as follows. First, an informal comparison between directed and undirected actions serves both as a motivation and a way to introduce the principle of projective invariance. Then the refined syntax is given, together with an algebraic notation of *bitonal trees* representing projective equivalence in a concrete way. With this alternative notation projective invariance is easily proved. Finally, an example is detailed and at the same time various practical ways to enhance the syntax are discussed.

*Acknowledgements.* This paper benefitted greatly from discussions with Luca Cardelli.

## 2 Why directed actions are better

Anticipating somewhat on our definitions of the next section, we discuss briefly why refining ordinary actions into directed ones. This discussion is also meant to be an introduction to the syntax.

### 2.1 An example system

Consider a solution  $\mathcal{S}$  with one big membrane containing two smaller ones, and suppose further the two smaller ones may fuse (a capability represented below by the actions  $f_2$ ), and the first may also fuse with the outer one (a capability represented below by the actions  $f_1$ ). Our notation for this situation will be:

$$\mathcal{S} := \langle \cdot; f_1 \rangle (\langle f_1, f_2; \cdot \rangle (P), \langle f_2; \cdot \rangle (Q)) \quad (1)$$

where  $P$  and  $Q$  represent the respective contents of the two inner membranes. Since we are dealing with directed actions, the notation is introducing a distinction between outward and inward actions. We use a place-value notation for this. For instance  $\langle f_1, f_2; \cdot \rangle (P)$  represents a membrane containing  $P$  with the fusion actions  $f_1$  and  $f_2$  both pointing outwards, because they stand on the left of the semi-column. All actions are pointing outwards in  $\mathcal{S}$ , except for the  $f_1$  borne by the outer membrane, which is pointing inwards.

Our two pairs of fusions  $f_1$  and  $f_2$  can cause either an horizontal fusion (mate) or a vertical one (exo) as long as they are “facing” each other. The system  $\mathcal{S}$  can therefore evolve in two ways depending on whether the horizontal fusion or the vertical one happens first:

$$\begin{aligned} \langle \cdot; f_1 \rangle (\langle f_1, f_2; \cdot \rangle (P), \langle f_2; \cdot \rangle (Q)) &\xrightarrow{f_2} \langle \cdot; f_1 \rangle (\langle f_1; \cdot \rangle (P, Q)) \xrightarrow{f_1} P, Q, \langle \cdot; \cdot \rangle () \\ \langle \cdot; f_1 \rangle (\langle f_1, f_2; \cdot \rangle (P), \langle f_2; \cdot \rangle (Q)) &\xrightarrow{f_1} P, \langle \cdot; f_2 \rangle (\langle f_2; \cdot \rangle (Q)) \xrightarrow{f_2} P, Q, \langle \cdot; \cdot \rangle () \end{aligned}$$

The first step of the second row is the interesting bit. The action  $f_2$  that was outwards has been *reversed* and is now pointing inwards. This is the formal counterpart of the membrane reversion happening upon vertical membrane fusion (see the flippase example discussed in the introduction). The  $f_1$  step reverses

directions just as the real thing does. As a consequence both  $f_2$  actions are still face to face, and, as said, still able to interact. No matter in which order the fusions are happening, they converge to the same end solution.

## 2.2 Same example undirected

The situation is different in the undirected case, obtained by forgetting the distinction between inward and outward pointing actions. The same example cast in the original undirected brane calculus reads now as follows:<sup>3</sup>

$$\mathcal{T} := \langle f_1 \rangle (\langle f_1, f'_2 \rangle (P), \langle f'_2 \rangle (Q)) \quad (2)$$

where  $f_1$  might cause a vertical fusion, while  $f'_2$  might cause an horizontal one. Note that since actions are now undirected, there is no longer a notion of two actions facing each other. Therefore, one has to let the action itself tell whether it is meant to trigger a vertical fusion or an horizontal one. Thus, the system evolves as follows:

$$\begin{aligned} \langle f_1 \rangle (\langle f_1, f'_2 \rangle (P), \langle f'_2 \rangle (Q)) &\xrightarrow{f'_2} \langle f_1 \rangle (\langle f_1 \rangle (P, Q)) \xrightarrow{f_1} P, Q, \langle - \rangle () \\ \langle f_1 \rangle (\langle f_1, f'_2 \rangle (P), \langle f'_2 \rangle (Q)) &\xrightarrow{f_1} P, \langle f'_2 \rangle (\langle f'_2 \rangle (Q)) \xrightarrow{f'_2} P, Q, \langle - \rangle () \end{aligned}$$

To close the square in a way similar to the directed case above, one would have either to have the first interaction, as a side-effect, change both the action  $f'_2$  in the outer part (easy) and the inner one in the membrane enclosing  $Q$ , which seems absurd, since this membrane doesn't take part in the interaction; or to suppose further that whatever can cause mate, can also cause exo, which, in the absence of directions, seems too lax a control to be expressive or even plausible.

Directed actions solve the square naturally. Now the important point is that not having this square is a violation of what we call *projective invariance*, which postulates that the nature of membrane interactions is such that the physics underlying the interaction can't tell the top from the bottom.

To see why, let us return at the example, and suppose the region (also called the lumen) containing the two membranes enclosing  $P$  and  $Q$  is projected at infinity. Under this change of viewpoint, what once was the outer space, becomes a bounded region and the corresponding term becomes ( $f_1$  is inverted purposefully):

$$\mathcal{S}' := \langle f_1; - \rangle (), \langle f_1, f_2; - \rangle (P), \langle f_2; - \rangle (Q), \quad (3)$$

and from *that* new point of view, the two fusions are of the mate sort, and they clearly commute. Therefore, if one follows the principle,  $\mathcal{S}$  and  $\mathcal{S}'$  which only differ by a change of the point of view, or the top-level of the solution, should behave the same, which in this particular case means that  $\mathcal{S}$  has to close the square explained above.

<sup>3</sup> In the original calculus, “perps” breaking the symmetry between a fusion action and a fusion co-action are used. They are not shown here.

### 2.3 Summary

To summarize this discussion, we may say that the original calculus was designed in order to respect bitonality invariance, and that the present revision is designed to further incorporate projective invariance.

As a final note before to proceed to the definitions, we may observe that both invariance constraints can be derived from Cardelli's fundamental remark that the double layer structure of biological membranes essentially enforces a single local membrane operation of *switching* [13]. There could well be a way of presenting the projective brane calculus as an abstract rendition of a more detailed calculus, incorporating double layer membrane switching as a unique primitive operation, and satisfying both invariance principles.

## 3 Membrane Interactions

It remains to give a proper definition and prove a general form of the projective invariance investigated above.

### 3.1 Actions, Branes and Solutions.

The syntax given in Table 1 defines *actions* and *solutions*. The set of actions will be denoted by  $\mathcal{A}$ . Basic actions can be of three sorts: fusions, wraps and bubbles, and actions can be combined freely by product and prefix to construct a membrane (abbreviated to brane). Solutions are products of solutions nested in membranes.

$\alpha :=$	<b>actions</b>
$f_n$	fusion
$\mathbf{b}_n\langle\sigma; \sigma\rangle$	bubble
$\mathbf{w}_n\langle\sigma; \sigma\rangle$	wrap
$\mathbf{w}_n^\perp$	co-wrap
$\sigma :=$	<b>branes</b>
-	empty brane
$\sigma, \sigma$	product
$\alpha.\langle\sigma; \sigma\rangle$	prefix
$P :=$	<b>solutions</b>
-	empty solution
$\langle\sigma; \sigma\rangle(P)$	membrane
$P, P$	product

**Table 1.** Actions, Branes and Solutions

All comma-separated expressions, branes or solutions, are to be understood up to associativity and commutativity.

So far this language only concerns membranes and their interactions capabilities and doesn't include proteins and complexes, nor any other molecules. Following Cardelli, a somewhat richer language of actions will be used in the example at the end. In particular, we will introduce molecules and actions to bind and release those, it will also be convenient to use “banged” or inexhaustible actions written  $!\sigma$ , as well as a mechanism of recruitment to specify how actions are allocated to the sub-branes when a brane is divided. But this is for later, as none of what interests us for now is depending on these choices.

### 3.2 Membrane Interactions

Interactions are sorted in three groups: bubbles, fusions and wraps. Both bubbles and wraps are dividing a membrane. This is the reason why the associated actions have as arguments the actions dispatched to the membrane which they create. Fusions are fusing two membranes in one and don't need such an allocation. To ease reading action indices are not represented, nor are prefixes.

*Bubbles: Drip & Pino*

$$\begin{aligned} \langle \mathbf{b}\langle\sigma';\tau'\rangle, \sigma; \tau \rangle \langle P \rangle &\xrightarrow{drip} \langle\sigma';\tau'\rangle \langle \_ \rangle, \langle\sigma; \tau \rangle \langle P \rangle \\ \langle\sigma; \tau, \mathbf{b}\langle\sigma';\tau'\rangle \rangle \langle P \rangle &\xrightarrow{pino} \langle\sigma; \tau \rangle \langle \langle\sigma';\tau'\rangle \langle \_ \rangle, P \rangle \end{aligned}$$

A bubble action divides the brane it is sitting on, creating a new brane, which is either outside or inside the original one, depending on whether the action points outwards (drip) or inwards (pino).

*Fusions: Mate & Exo*

$$\begin{aligned} \langle \mathbf{f}, \sigma; \tau \rangle \langle P \rangle, \langle \mathbf{f}, \sigma'; \tau' \rangle \langle Q \rangle &\xrightarrow{mate} \langle\sigma, \sigma'; \tau, \tau' \rangle \langle P, Q \rangle \\ \langle\sigma; \tau, \mathbf{f} \rangle \langle \langle \mathbf{f}, \sigma'; \tau' \rangle \langle P \rangle, Q \rangle &\xrightarrow{exo} \langle\sigma, \tau'; \tau, \sigma' \rangle \langle Q \rangle \end{aligned}$$

A pair of fusion actions facing each other induces a fusion which is either vertical (exo) or horizontal (mate) depending on where the two actions are sitting.

*Wraps: Phago, Bud & Swap*

$$\begin{aligned} \langle \mathbf{w}^\perp, \sigma; \tau \rangle \langle P \rangle, \langle \mathbf{w}\langle\sigma''; \tau''\rangle, \sigma'; \tau' \rangle \langle Q \rangle &\xrightarrow{phago} \langle\sigma'; \tau' \rangle \langle \langle\sigma''; \tau''\rangle \langle \langle\sigma; \tau \rangle \langle P \rangle \rangle, Q \rangle \\ \langle\sigma'; \tau', \mathbf{w}\langle\sigma''; \tau''\rangle \rangle \langle \langle \mathbf{w}^\perp, \sigma; \tau \rangle \langle P \rangle, Q \rangle &\xrightarrow{bud} \langle\sigma''; \tau''\rangle \langle \langle\sigma; \tau \rangle \langle P \rangle \rangle, \langle\sigma'; \tau' \rangle \langle Q \rangle \\ \langle\sigma; \mathbf{w}^\perp, \tau \rangle \langle \langle \mathbf{w}\langle\sigma''; \tau''\rangle, \sigma'; \tau' \rangle \langle P \rangle, Q \rangle &\xrightarrow{swap} \langle\sigma; \tau \rangle \langle \langle\tau''; \sigma''\rangle \langle P, \langle\tau'; \sigma' \rangle \langle Q \rangle \rangle \end{aligned}$$

A pair of a wrap and a co-wrap actions facing each other induces a division of the brane carrying the wrap action, which encloses the brane carrying the co-wrap action. The wrapped brane is either outside the wrapping brane (phago),

$$\begin{array}{ccc}
\langle \mathfrak{b}(\sigma'; \tau'), \sigma; \tau \rangle(P), Q & \xrightarrow{\text{drip}} & \langle \sigma'; \tau' \rangle(-), \langle \sigma; \tau \rangle(P), Q \\
\updownarrow \sim & & \updownarrow \sim \\
P, \langle \tau; \sigma, \mathfrak{b}(\sigma'; \tau') \rangle(Q) & \xrightarrow{\text{pino}} & P, \langle \tau; \sigma \rangle(\langle \sigma'; \tau' \rangle(-), Q) \\
\\
\langle \mathfrak{f}, \sigma; \tau \rangle(P), \langle \mathfrak{f}, \sigma'; \tau' \rangle(Q), R & \xrightarrow{\text{mate}} & \langle \sigma, \sigma'; \tau, \tau' \rangle(P, Q), R \\
\updownarrow \sim & & \updownarrow \sim \\
P, \langle \tau; \mathfrak{f}, \sigma \rangle(\langle \mathfrak{f}, \sigma'; \tau' \rangle(Q), R) & \xrightarrow{\text{exo}} & P, Q, \langle \tau, \tau'; \sigma, \sigma' \rangle(R)
\end{array}$$

**Table 2.** Bubble and Fusions symmetries.

or inside (bud). These two steps are similar to respectively pino and drip, except that the other brane (carrying the co-action) is wrapped in the process.

However the swap interaction is different, and doesn't look like corresponding to any actual biological transformation. What it really is, is a phago step, but seen from the point of view of  $P$ , that is to say from the inside. It is there to close our set of interactions in the case when  $P$  is taken as the region at infinity.

Closing these basic interactions under any context results in a transition system over solutions, written  $\rightarrow$  and called thereafter *reduction*.

### 3.3 Projective equivalence

Projective equivalence, written  $\sim$ , is the least equivalence relation such that:

$$P, \langle \sigma; \tau \rangle(Q) \sim \langle \tau; \sigma \rangle(P), Q \quad (4)$$

One has to be careful about two things here:

- $\sigma$  and  $\tau$  are exchanged above, because inward actions become outward actions and conversely;
- changing one's point of view is a global transformation and therefore not compatible with product:

$$\begin{aligned}
R, P, \langle \sigma; \tau \rangle(Q) &\sim \langle \tau; \sigma \rangle(R, P), Q \\
&\not\sim R, \langle \tau; \sigma \rangle(P), Q
\end{aligned}$$

**Proposition 1** *Reduction respects projective equivalence, that is to say, if  $P \sim Q$  and  $P \rightarrow P'$ , then there exists a  $Q'$ , such that  $P' \sim Q'$  and  $Q \rightarrow Q'$ .*

$$\begin{array}{ccc}
P, \langle \tau; \mathbf{w}^\perp, \sigma \rangle \langle \langle \mathbf{w} \langle \sigma''; \tau'' \rangle, \sigma'; \tau' \rangle \langle Q \rangle, R \rangle & \xrightarrow{\text{swap}} & P, \langle \tau; \sigma \rangle \langle \langle \tau''; \sigma'' \rangle \langle Q, \langle \tau'; \sigma' \rangle \langle R \rangle \rangle \rangle \\
\uparrow \sim & & \uparrow \sim \\
\langle \mathbf{w}^\perp, \sigma; \tau \rangle \langle P \rangle, \langle \mathbf{w} \langle \sigma''; \tau'' \rangle, \sigma'; \tau' \rangle \langle Q \rangle, R & \xrightarrow{\text{phago}} & \langle \sigma'; \tau' \rangle \langle \langle \sigma''; \tau'' \rangle \langle \langle \sigma; \tau \rangle \langle P \rangle \rangle, Q \rangle, R \\
\uparrow \sim & & \uparrow \sim \\
\langle \tau'; \mathbf{w} \langle \sigma''; \tau'' \rangle, \sigma' \rangle \langle \langle \mathbf{w}^\perp, \sigma; \tau \rangle \langle P \rangle, R \rangle, Q & \xrightarrow{\text{bud}} & \langle \sigma''; \tau'' \rangle \langle \langle \sigma; \tau \rangle \langle P \rangle \rangle, Q, \langle \tau'; \sigma' \rangle \langle R \rangle
\end{array}$$

**Table 3.** Wrap symmetries.

This can be proved by induction on the number of steps of  $\sim$  applied to go from  $P$  to  $Q$ . Key lemmas are given by Tables 2 and 3 and express the internal symmetry relating interactions of a same sort. However, a more perspicuous argument can be given if one changes notation. This new notation is developed in the next section. Of course it is more abstract and doesn't have the direct biological interpretation that the first syntax supports. It is nevertheless very convenient and in some sense, as we shall see, equivalent.

## 4 Bitonal Tree Representation

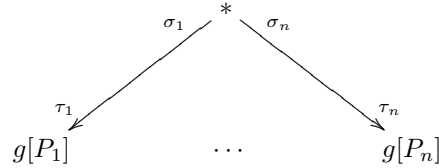
A graph is a pair  $(V, E)$ , where  $V$  is a set of vertices (or nodes), and  $E \subseteq V \times V$  is a set of edges. Such graphs are undirected, and have at most one edge between any two given vertices.

A *bitonal graph* consists of the following data:

- an undirected graph  $G = (V, E)$ ;
- a labeling function  $\lambda$  mapping  $E$  to  $\mathcal{A} \times \mathcal{A}$ ;
- a 2-colouring of  $G$ , that is a map  $\kappa$  from  $V$  to  $\{0, 1\}$ , such that  $(x, y) \in E \Rightarrow \kappa(x) \neq \kappa(y)$ .

Being 2-colourable  $G$  can only have even length cycles. When  $G$  is acyclic and connected one says  $G$  is a bitonal tree. A *pointed bitonal tree*  $(x, V, E, \lambda, \kappa)$  is a bitonal tree together with a distinguished vertex  $x \in V$ .

There is an obvious map  $g[\_]$  from solutions to pointed bitonal trees. Given a solution  $\langle \sigma_1; \tau_1 \rangle \langle P_1 \rangle, \dots, \langle \sigma_n; \tau_n \rangle \langle P_n \rangle$  one represents it as:

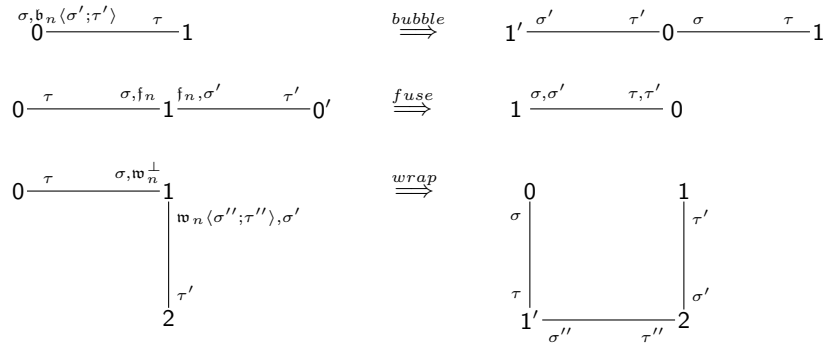


where the distinguished vertex is denoted  $*$ . Up to associativity and commutativity on the side of solutions, this map is a bijection. In the vocabulary of graph theory,  $g[P]$  is the dual graph of  $P$ , its nodes are regions (or lumens) and its edges represent region adjacency. Changing one's “point of view” amounts simply in this dual representation to changing the distinguished vertex in the dual world of bitonal trees.

**Proposition 2** *Let  $P$  and  $P'$  be two solutions, then  $P \sim P'$  if and only if  $g[P]$  and  $g[P']$  differ only in their distinguished vertex.*

This is easily seen, and is what we wanted. Bitonal trees are giving a concrete representation of projective equivalence classes. Furthermore, membrane reductions can be defined directly on this representation.

#### 4.1 Tree interactions



**Table 4.** Rewriting bitonal trees.

Tree interactions are presented in Table 4. Even numbered vertices corresponding to one colour and odd numbered ones to the other. One sees that for each type of membrane interaction, there remains only one interaction in the world of trees. It is easy to verify that:

- these transformations preserve bitonality,
- that they are definable with general 2-colourable graphs,<sup>4</sup>
- that they preserve acyclicity and connectivity, and
- most importantly, that once one chooses a particular point of view and therefore an orientation of the tree, they correspond exactly to the membrane interactions of the preceding section.

<sup>4</sup> Fuse and wrap only apply when the two left hand side edges are distinct and then even nodes must be distinct as well, since multi arcs are not allowed.

To illustrate the last point, consider the case of the wrap rewriting. Depending on where the tree is rooted, wrap will correspond to one of phago, swap and bud. Specifically, if in the partial order induced by the choice of a root,  $1 > 0$  and  $2$ , then it corresponds to a phago, if  $2 > 1$  and  $0$  then it is a bud, and if  $0 > 1$  and  $2$ , then it is a swap.

Proposition 1 follows now easily.

## 5 Enriching the language

Next, we would like to test on a concrete and reasonably large example how well our modified brane language does in expressing biological interactions involving membranes. Yet, for the example to be interesting we need to flesh in a bit our spartan syntax and adjust some notations. This is what we do in this section.

### 5.1 Recruitment

The first natural modification, already suggested by Cardelli, is to have membrane dividing actions (wraps and bubbles) explicitly recruit the actions allocated to the new membrane. This is particularly useful when there might be very few copies of some action of interest.

For instance a pino allocating actions  $\sigma$  to the inner bubble becomes:

$$\langle \sigma; \mathbf{b}\langle \cdot; \sigma \rangle \rangle (P) \xrightarrow{\text{pino}} \langle \cdot; \cdot \rangle (\langle \cdot; \sigma \rangle (\cdot), P)$$

The recruited actions  $\sigma$  are consumed and no longer part of the outer membrane they were taken from.

Adding a recruitment mechanism has the side effect to make even clearer which membranes are turned upside down and when. Pino above has to flip  $\sigma$  because the inner part of the new membrane corresponds to the outer part of the former membrane it was pinched off. This is not new and we knew it from the exo case. The same considerations applies to phago, but not to the “direct” interactions drip, mate and bud where the branes orientation are unchanged.

In the example we will use it specifically to describe the trip of the action corresponding to the *escape receptor* that the virus uses to exit the cell once it is properly replicated.

### 5.2 Protein-membrane interactions

As natural as our place-value syntactic convention for actions (having outward action and inward action separated by a semi-column) may be, it is not so readable in examples. So to ease reading, and starting from now, actions will have an explicit superscript indicating whether they are directed inward or outward. For instance an outward fusion action will be written  $f^e$ , while an inward fusion action will be written  $f^i$ .

The other modification that we need is a mechanism to control the insertion in membranes of various molecules such as proteins. Introducing the corresponding two actions, B and R, we keep Cardelli's elegant *bind-and-release* rules [1]:

$$\begin{array}{l}
m, \langle \mathbf{B}^e(m). \sigma, \tau \rangle \langle P \rangle \xrightarrow{Bind(out)} \langle \sigma, \tau \rangle \langle P \rangle \\
\langle \sigma; \mathbf{B}^i(m). \tau \rangle \langle m, P \rangle \xrightarrow{Bind(in)} \langle \sigma, \sigma' \rangle \langle P \rangle \\
\langle \mathbf{R}^e(m). \sigma, \sigma' \rangle \langle P \rangle \xrightarrow{Release(out)} m, \langle \sigma, \sigma' \rangle \langle P \rangle \\
\langle \mathbf{R}^i(m). \sigma, \sigma' \rangle \langle P \rangle \xrightarrow{Release(in)} \langle \sigma, \sigma' \rangle \langle m, P \rangle
\end{array}$$

We used the superscript notation to distinguish whether an action is directed inward or outward. In this simple model, inserted molecules are not considered to be directed.

### 5.3 Routing molecules

The bind and release constructions can be combined to route molecules in the solution. A basic combination of actions is the following (in superscript notation):

$$\mathbf{B}^i(m). \mathbf{b}^e \langle f_{dest}^e, \mathbf{R}^i(m) \rangle$$

the binding mechanism is looking inwards for an  $m$ , then pinches off a bubble with  $m$  inserted in the bubble, a bubble which is then sent to its destination, that is any membrane offering complementary fuse action.

A typical evolution would be then:

$$\begin{array}{l}
\langle \mathbf{B}^i(m). \mathbf{b}^e \langle f_{dest}^e, \mathbf{R}^i(m) \rangle \rangle \langle m \rangle, \langle f_{dest}^e \rangle \langle - \rangle \xrightarrow{Bind(in)} \\
\langle \mathbf{b}^e \langle f_{dest}^e, \mathbf{R}^i(m) \rangle \rangle \langle - \rangle, \langle f_{dest}^e \rangle \langle - \rangle \xrightarrow{drip} \\
\langle - \rangle \langle - \rangle, \langle f_{dest}^e, \mathbf{R}^i(m) \rangle \langle - \rangle, \langle f_{dest}^e \rangle \langle - \rangle \xrightarrow{Release(in)} \\
\langle - \rangle \langle - \rangle, \langle f_{dest}^e \rangle \langle m \rangle, \langle f_{dest}^e \rangle \langle - \rangle \xrightarrow{mate} \\
\langle - \rangle \langle - \rangle, \langle \rangle \langle m \rangle
\end{array}$$

The way the macro is written above,  $m$  can be freed from the bubble membrane either before or after it has fused with the destination. Other combinations are possible by prefixing. For instance if one insists that  $m$  is freed in the bubble before the latter is fused, then one writes  $\mathbf{B}^i(m). \mathbf{b}^e \langle \mathbf{R}^i(m). f_{dest}^e \rangle$ .

Bubble trafficking in the example will include actions with their sorts chosen so as to remind of their destination, *e.g.*,  $f_G$  will denote fuse actions recognized by the Golgi apparatus.

## 6 A virus invasion formalized

As said, our example will be the *modus operandi* of a virus. The system below is pictured roughly after the influenza virus, a retrovirus that carries genetic information as RNA and contains a retrotranscriptase, written DNAp below,

converting it back to DNA so that all the virus components can be synthesized by the host cell.

One thing to keep track of, in relation to our choice of directed actions, is that the action corresponding to the escape receptor prefixes an action of the form  $B(C)$ , meant to bind the final virus complex  $C$ , and help him escape. It therefore has to point ultimately inwards. Since the patch of membrane hosting it is fused with the outer membrane by an *exo*, that particular action has to point outwards by the time it is synthesized.

## 6.1 Description

The virus presents itself to the outer plasmic membrane of the cell and fools the cell in engulfing it as in the ordinary endocytotic pathway. Properly wrapped the virus is transported to a close organelle termed the early endosome, from which it escapes to deliver its cargo in the cytosol. The cargo, consists in genetic material that codes for the capsid (envelope), and various other proteins needed to reconstruct the virus. Among these proteins there is a receptor that is going to allow the virus, when reconstructed in the cell (possibly in many copies) to escape to the outer space.

We first define all the organelles involved in this process together with the virus itself:

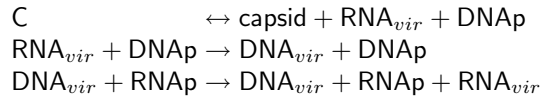
$$\begin{aligned}
\text{Virus} &:= \langle \mathfrak{w}^e, \mathfrak{f}_{exo}^e \rangle (C) \\
\text{Cell} &:= \langle !\mathfrak{w}^e(\mathfrak{f}_{endo}^e), !\mathfrak{f}_{endo}^e, !\mathfrak{f}_{out}^i \rangle (\text{Nucleus}, \text{Endosome}, \text{ER}, \text{Golgi}) \\
\text{Endosome} &:= \langle \mathfrak{f}_{endo}^e, \mathfrak{f}_{exo}^i \rangle (-) \\
\text{Nucleus} &:= \langle !B^e(\text{RNA}_{vir}).R^i(\text{RNA}_{vir}), !B^e(\text{DNAP}).R^i(\text{DNAP}), \\
&\quad !B^i(\text{RNA}_{vir}).R^e(\text{RNA}_{vir}) \rangle (\text{RNAP}) \\
\text{ER} &:= \langle !B^e(\text{RNA}_{vir}).(R^e(\text{RNA}_{vir}), R^e(\text{capsid}), \\
&\quad R^e(\text{DNAP}), \text{send\_receptor}) \rangle (-) \\
\text{Golgi} &:= \langle !\mathfrak{f}_G^e, !\mathfrak{b}^e \langle \mathfrak{f}_{out}^e, \text{action\_receptor} \rangle \rangle (-)
\end{aligned}$$

where we used the ‘!’ to represent inexhaustible actions, and the actions:

$$\begin{aligned}
\text{send\_receptor} &:= \mathfrak{b}^e \langle \mathfrak{f}_G^e, \text{action\_receptor} \rangle \\
\text{action\_receptor} &:= B^e(C). \mathfrak{b}^i \langle \mathfrak{w}^e, \mathfrak{f}_{exo}^e, R^i(C) \rangle
\end{aligned}$$

The actions sitting on the cell brane (the plasmic membrane) represent the cell endocytotic and exocytotic pathways, while the actions on top of the nucleus brane represent the workings of the nuclear pore complexes forming the only passageways between the nuclear lumen, where transcription take place, and the cytosolic one, where translation takes place. Translation is here modeled indirectly at the level of ER.

One also has the respective reactions of decomplexation, retrotranscription and forward transcription:



We assume the first reaction to be reversible and suppose also that the last two reactions are happening in the nucleus.

## 6.2 Running the system

With the system definition in place, we can play our model. To keep a manageable notation we haven't systematically represented actions recruited and consumed during bubble formation, and only relevant actions, membranes and molecules are represented at each step.

$$\begin{aligned} \text{Virus, Cell} &\rightarrow_{\text{phago}} \langle !\mathfrak{w}^e \langle f_{\text{endo}}^e \rangle, !f_{\text{out}}^i \rangle \langle \langle f_{\text{endo}}^e \rangle \langle \langle f_{\text{exo}}^e \rangle \langle \text{C} \rangle \rangle, \text{Endosome}, \dots \rangle \\ &\rightarrow_{\text{mate}} \langle \dots \rangle \langle \langle f_{\text{exo}}^i \rangle \langle \langle f_{\text{exo}}^e \rangle \langle \text{C} \rangle \rangle, \dots \rangle \\ &\rightarrow_{\text{exo}} \langle \dots \rangle \langle \text{C, Nucleus}, \dots \rangle \end{aligned}$$

The virus just broke in and delivered the cargo in the cytosol. We don't write the outer membrane anymore. The C complex dissociates and after transfer to the nucleus, backward and forward transcription take place.

$$\begin{aligned} &\rightarrow_{\text{decomplexation}} \text{ capsid, RNA}_{\text{vir}}, \text{DNAP, Nucleus}, \dots \\ &\rightarrow_{\text{Bind}^2, \text{Release}^2} \langle \dots \rangle \langle \text{RNA}_{\text{vir}}, \text{DNAP, RNAP} \rangle, \dots \\ &\rightarrow_{\text{retrotranscription}} \langle \dots \rangle \langle \text{DNA}_{\text{vir}}, \text{RNAP} \rangle, \dots \\ &\rightarrow_{\text{transcription}} \langle \dots \rangle \langle \text{RNA}_{\text{vir}}, \text{DNA}_{\text{vir}}, \text{RNAP} \rangle, \dots \\ &\rightarrow_{\text{Bind, Release}} \text{ Nucleus, RNA}_{\text{vir}}, \text{ER}, \dots \end{aligned}$$

The viral RNA has been duplicated and sent out again from the nucleus. It then binds to ER where it is translated. Products of this translation, capsid, DNAP, and RNA<sub>vir</sub> are released in the cytosol, while the receptor, modeled here as an action, remains inserted in ER:

$$\begin{aligned} &\rightarrow_{\text{Drip}} \text{ER, } \langle f_G^e, \text{action\_receptor} \rangle \langle \_ \rangle, \text{Golgi}, \dots \\ &\rightarrow_{\text{Mate}} \langle \text{action\_receptor}, !\mathfrak{b}^e \langle f_{\text{out}}^e, \text{action\_receptor} \rangle, \dots \rangle \langle \_ \rangle, \dots \end{aligned}$$

The drip action needs to recruit a copy of the receptor. Now that one is around, it can be triggered. We write again the plasmic outer membrane:

$$\begin{aligned} &\rightarrow_{\text{Drip}} \langle \langle f_{\text{out}}^i, \dots \rangle \langle \text{Golgi}, \langle f_{\text{out}}^e, \text{action\_receptor} \rangle \langle \_ \rangle, \dots \rangle \\ &\rightarrow_{\text{Exo}} \langle \overline{\text{action\_receptor}}, \dots \rangle \langle \text{Golgi}, \dots \rangle \end{aligned}$$

with the following notation for the flipped action:

$$\overline{\text{action\_receptor}} := \text{B}^i(\text{C}).\mathfrak{b}^e \langle \mathfrak{w}^e, f_{\text{exo}}^e, \text{R}^i(\text{C}) \rangle$$

Observe that the suffix is flipped too in the exo. So now the receptor is bound to the outer membrane, and facing inwards as it should, waiting for the viral complex to escape.

Meanwhile the other components, capsid, DNAP and RNA<sub>vir</sub>, have assembled again in the cytosol, and may now exit the cell with the help of the receptor:

$$\begin{aligned} &\rightarrow_{\text{complexation}} \langle \text{B}^i(\text{C}).\mathfrak{b}^e \langle \mathfrak{w}^e, f_{\text{exo}}^e, \text{R}^i(\text{C}) \rangle, \dots \rangle \langle \text{C}, \dots \rangle \\ &\rightarrow_{\text{Bind}} \langle \mathfrak{b}^e \langle \mathfrak{w}^e, f_{\text{exo}}^e, \text{R}^i(\text{C}) \rangle, \dots \rangle \langle \dots \rangle \\ &\rightarrow_{\text{Drip}} \text{Virus, Cell} \end{aligned}$$

and the overall system is back to its original state, except that the cell contains now many copies of the virus components and is actively manufacturing more.

## 7 Conclusion

Cardelli’s original brane calculus was designed to express biological interactions involving membranes. By design, its syntax reflected the bitonality invariant. We have presented here a refined brane calculus introducing directed actions, where the original calculus only used undirected ones. Thus our variant also incorporates a new notion of projective invariance. Together with the standard syntax, we gave an alternative syntax of bitonal trees, which reflects directly the new invariant, and where the number of primitive interactions downs to three: fusion, wrap and bubble. At the same time this revision of the original calculus is more convincing biologically in that it gives richer means to express how proteins are inserted in the membrane. The viral example developed and specifically the way the virus “escape” receptor is handled makes a good case for this refined language of actions.

While not completely understood, some observable membrane interactions within a cell are beginning to be explained at the protein level by today’s molecular biology. Internal routing mechanisms, involving proteins such as *t*-snares and *v*-snares, are believed to be a pretty good account of how specific routing is handled in the cell. Membrane fission, as in endocytosis, is also described in some cases at the level of proteins, for instance in clathrin-coated vesicle formation [14, pp.514–516]. It therefore seems tempting to pursue the effort and explore whether one can actually reduce membrane actions, which here were taken to be abstract atomic properties of a given membrane, to the properties of proteins (and glycolipids) actually inserted in the membrane. In other words one would like to merge a protein calculus such as the  $\kappa$ -calculus [15], and a brane calculus, in a bigger and more comprehensive language to be able to handle additional detail when additional detail is known. This is one interesting subject for further investigation.

Another question which might be worth exploring is that of endowing the brane-calculus with a quantitative operational semantics. If one thinks again about the virus infection scenario, a non-deterministic semantics is clearly not enough to describe what is happening. Whether an infection will succeed or not hinges on the kinetics of the various processes involved, and if one is willing to analyse this, it is important to incorporate quantitative aspects in the semantics. Using Gillespie’s algorithm [16, 17], a non-deterministic transition system can be made to generate a continuous-time stochastic process over the system state space. This was done already in the calculus of bio-ambients [10], and it should be easy to adapt to the present framework. However, the challenge is not only to have a quantitative semantics, but also to reproduce some known regulation phenomena, and some more work has to be done here to get some interesting examples.

Finally, as said at the end of the second section, there is also the question of whether there is a more detailed calculus based on the single local operation of membrane switching described by Cardelli [13], from which one could reconstruct the projective brane calculus.

## References

1. Luca Cardelli. Brane calculi. In *Proceedings of BIO-CONCUR'03, Marseille, France*, volume ? of *Electronic Notes in Theoretical Computer Science*. Elsevier, 2003. To appear.
2. Aviv Regev, William Silverman, and Ehud Shapiro. Representation and simulation of biochemical processes using the  $\pi$ -calculus process algebra. In R. B. Altman, A. K. Dunker, L. Hunter, and T. E. Klein, editors, *Pacific Symposium on Biocomputing*, volume 6, pages 459–470, Singapore, 2001. World Scientific Press.
3. Corrado Priami, Aviv Regev, Ehud Shapiro, and William Silverman. Application of a stochastic name-passing calculus to representation and simulation of molecular processes. *Information Processing Letters*, 2001.
4. Aviv Regev and Ehud Shapiro. Cells as computation. *Nature*, 419, September 2002.
5. Vincent Danos and Cosimo Laneve. Core formal molecular biology. In *Proceedings of the 12th European Symposium on Programming (ESOP'03, Warsaw, Poland)*, volume 2618 of *LNCS*, pages 302–318. Springer, April 2003.
6. Vincent Danos and Cosimo Laneve. Graphs for formal molecular biology. In *Proceedings of the First International Workshop on Computational Methods in Systems Biology (CMSB'03, Rovereto, Italy)*, volume 2602 of *LNCS*, pages 34–46. Springer, February 2003.
7. Vincent Danos and Jean Krivine. Formal molecular biology done in CCS. In *Proceedings of BIO-CONCUR'03, Marseille, France*, volume ? of *Electronic Notes in Theoretical Computer Science*. Elsevier, 2003. To appear.
8. Marc Chiaverini and Vincent Danos. A core modeling language for the working molecular biologist. In *Proceedings of CMSB'03*, volume 2602 of *LNCS*, page 166. Springer, 2003.
9. Gh. Paun. *Membrane Computing. An Introduction*. Springer-Verlag, Berlin, 2002.
10. Aviv Regev, Ekaterina M. Panina, William Silverman, Luca Cardelli, and Ehud Shapiro. Bioambients: An abstraction for biological compartments. *Theoretical Computer Science*, 2003. To Appear.
11. Luca Cardelli. Brane calculi (slides). Slides., 2003.
12. John W. Kimball. [http://users.rcn.com/jkimball.ma.ultranet/BiologyPages/Kimball's Biology Pages](http://users.rcn.com/jkimball.ma.ultranet/BiologyPages/Kimball's%20Biology%20Pages). Online biology textbook, 2003.
13. Luca Cardelli. Bitonal membrane systems. Draft, 2003.
14. Bruce Alberts et al. *Essential Cell Biology*. International Series on Computer Science. Garland Science, New-York, 2004.
15. Vincent Danos and Cosimo Laneve. Formal molecular biology. *Theoretical Computer Science*, 325(1):69–110, September 2004.
16. Daniel T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. Comp. Phys.*, 22:403–434, 1976.
17. Daniel T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem*, 81:2340–2361, 1977.