Where membranes meet complexes

Luca Cardelli[*] and Sylvain Pradalier[†]

Abstract

We introduce a calculus handling complexation of molecules and membranes. The approach is based on adding dynamic interfaces to processes, which model bonds between molecules. The calculus is then extended with a notion of hierarchy to handle membranes.

Introduction In biochemistry, proteins and other molecules have a common way of interacting among themselves by forming "complexes". A complex is a combinations of two or more molecules attached together along complementary surfaces. *Complexation* is the formation of a complex and *decomplexation* is the breaking up of a complex, both of which may be actively triggered, e.g., by other complexation and decomplexation interaction.

When modeling even the simplest biological systems, the issue immediately comes up of how to model complexation. Shapiro and Regev [8, 4] proposed using a fresh private channel in π-calculus to represent complexation, and a synchronization on that channel followed by discarding the channel to represent decomplexation. This technique is flexible and allows many interesting variations, and is available within a well understood calculus. However, this technique uses some high power features (private scopes, name passing) to represent what is essentially a combinatorial operation that, like parallel composition, ought to be expressible as a binary operator.

In this paper we explore using complexation as an operator; this turns out not to be so trivial to implement. We base our approach on attaching dynamic "interfaces" to processes to more easily track the "surfaces" along which processes interact to form complexes. Our notion of dynamic interfaces is similar to the mechanisms explored in beta-binders[6, 7]. However, we preserve binary complementary interactions as the fundamental mode of interaction between processes.

Membranes are another fundamental feature in biology. They divide the space within organisms into compartments, thus establishing different biochemical conditions in separate regions. Brane Calculi[1] and Projective Brane Calculus[3] are languages specifically dealing with membranes. They

[*]Microsoft Research
[†]ENS Cachan

provide a strong framework for describing and modeling membrane systems but they lack an elegant way of representing molecules and even more so, complexation. This paper proposes a language that handles molecules *and* membranes. Because membranes can stick to each other like molecules, we represent membranes as molecules that have an inner content. This is the same step that leads from π-calculus[5] to ambient-calculus[2].

Outline First we present the formalism for complexation. Then we show examples to explain why we need the different operators. Next we extend the calculus with a notion of hierarchy that represents the nesting of membranes and we give a biologically-inspired example where both complexes and membranes are needed. We finally discuss possible enhancements to the syntax, and perspectives.

Acknowledgments A discussion with Tony Hoare provided the initial direction for this work. Vincent Danos pointed out some mistakes and helped to explain intuitions behind the calculus.

1 Complexation

In biochemistry, complementary sites (surface features) of molecules can stick to each other, joining molecules into complexes. Hence, the possible interactions of a molecule with its environment are determined by its set of sites. We call the set of sites of a molecule its interface.

This observation, when reflected into process-calculus models, leads us to endow processes with interfaces. Molecules can change their shape, and hence their set of sites, in response to biochemical interactions; so we need our interfaces to be dynamic. This is achieved by allowing processes to offer and retract sites on their interfaces dynamically.

As we show by means of examples (Section 2), a number of process operators are needed to model even simple biological systems. Not surprisingly, we use much of the power of π-calculus, including notions of synchronization and scope restriction. Before presenting formally the syntax of our calculus, we recall some basic notions about multisets.

1.1 Notation for multisets

We write $\subseteq^+$, $\bigcap^+$, $+$ and $-$ for the inclusion, intersection, sum and difference of multisets. Let Σ be a set of names n (representing sites in interfaces), with $\Sigma^- = \{n^- | n \in \Sigma\}$ and $\Sigma^+ = \{n^+ | n \in \Sigma\}$, representing complementary sites. We use S,T to denote multisets over $\Sigma^- \bigcup \Sigma^+$.

Let $\overline{S} = \{a^+ | a^- \in S\} \bigcup \{a^- | a^+ \in S\}$ and $S^* = S \bigcup \overline{S}$. The "complementary intersection" is: $S \square T = (S \bigcap^+ \overline{T})^*$. We also note $S \amalg T$ for

$S + T - S \Box T$. As $(a^+ \in S \Box T) \Leftrightarrow (a^- \in S \Box T)$, we write $a \in S \Box T$ for $(a^+ \in S \Box T \wedge a^- \in S \Box T)$.

1.2 Syntax

- $a, b, c, \ldots \in \Sigma^- \bigcup \Sigma^+$ and $S, T, \ldots \in Multisets(\Sigma^- \bigcup \Sigma^+)$
- $P, Q, R, \ldots ::= \quad A_S \quad | \quad P\&Q \quad | \quad (\nu a)P \quad | \quad X \quad | \quad rec_X.P$
- $A, B, C, \ldots ::= \quad 0 \quad | \quad A + B \quad | \quad A|B \quad | \quad \alpha.A$
- $\alpha ::= \quad a^{\pm}\langle S \rangle \quad | \quad a^{\pm}(S) \quad | \quad \mathit{offer}(a) \quad | \quad \mathit{retract}(a) \quad | \quad \mathit{fork}(P)$

$\Sigma^- \bigcup \Sigma^+$ is the set of sites. The second line defines the syntax for molecules and complexes : it is either a sets of actions A with some interface S, or two molecules put in parallel with the parallel operator & (also used as the complexation operator), or a restriction on the name of a site, or a recursion variable or a recursion on a molecule. Action sets are defined in the third line: there is choice, parallel, and prefix. Finally we have basic actions: polyadic output, polyadic input, offer of a site, retract of a site, fork of a molecule. Further explanations will be given with the operational semantics (1.6). We use round brackets and square brackets for precedence.

1.3 Complexation and Interfaces

The main difference with π-calculus is the complexation operator & that behaves like a parallel, except for the handling of interfaces. It is this operator that makes bonds between molecules, by providing a new interface for the complex that hides the internal connections of the complex from further external interaction. Other differences are restrictions that can only occur at the top-level (they cannot be prefixed, for instance), and the duality $\pm$ of sites that represents complementarity of sites inependently of input/output direction.

Next we define the interface I(P) of a molecule or complex :

$$I(A_S) = S, \qquad I((\nu a)P) = I(P) - \{a^+, a^-\}, \qquad I(P\&Q) = I(P) \amalg I(Q)$$
$$I(X) = \emptyset, \qquad I(rec_X.P) = I(P)$$

For instance if $I(P) = \{a^+, a^+, b^-\}$ and $I(Q) = \{a^-, c^+\}$ then $I(P\&Q) = \{a^+, a^+, b^-\} + \{a^-, c^+\} - \{a^+, a^-\} = \{a^+, b^-, c^+\}$. One can see that the effect of the & operator on interfaces is to take the sum of I(P) and I(Q) and to remove an a^+ and an a^-. Intuitively & hides an a^+ on the left for each a^- on the right and vice-versa.

We have to distinguish between a^+ and a^- (that is, between two complementary sites on two molecules), because sites on molecules correspond to complementary shapes. Indeed, if molecule X can link with molecules Y and Z on a site a we do not want Y to be able to link to Z on a.

1.4 Free names

The definition of free names is quite obvious. We just detail cases that are not in the classical π-calculus:

- $fn(A_S) = fn(A) \bigcup Set(S)$ where Set(S) is the set-projection of the multiset S.
- $fn(a\langle S\rangle.A) = \{a\} \bigcup fn(A) \bigcup S$
- $fn(a(X).A) = \{a\} \bigcup (fn(A) \setminus X)$
- $fn(\mathit{offer}(a).A) = fn(\mathit{retract}(a).A) = fn(\mathit{fork}(P).A) = fn(A) \bigcup \{a\}$

1.5 Structural Congruence

The structural congruence is derived from the one of the π-calculus.

$$\begin{array}{rl}
P\&Q \equiv Q\&P, & P\&(Q\&R) \equiv (P\&Q)\&R, \\
P\&0 \equiv P, & \dfrac{P \equiv P'}{P\&Q \equiv P'\&Q} \\
(\nu x).(\nu y).P \equiv (\nu y).(\nu x).P, & (\nu x).0 \equiv 0 \\
(\nu x).P\&Q \equiv (\nu x).(P\&Q) & \quad if\ x \notin fn(Q) \\
\dfrac{P \equiv Q}{rec_X.P \equiv rec_X.Q}, & \dfrac{A \equiv B}{A_S \equiv B_S} \\
\dfrac{P \equiv_\alpha Q}{P \equiv Q}, & \dfrac{A \equiv A'}{\alpha.A \equiv \alpha.A'}, \\
A + B \equiv B + A, & A + (B + C) \equiv (A + B) + C, \\
A + 0 \equiv A, & \dfrac{A \equiv A'}{A + B \equiv A' + B} \\
A|B \equiv B|A, & A|(B|C) \equiv (A|B)|C, \\
A|0 \equiv A, & \dfrac{A \equiv A'}{A|B \equiv A'|B}
\end{array}$$

1.6 Operational Semantics

Operational semantics of the π-calculus is often presented as a labeled transitions system. We do not need it here and thus prefer the simpler unlabeled version.

- $\frac{P \equiv P' \ ,\ P' \to Q' \ ,\ Q' \equiv Q}{P \to Q}$: (Cong)
- $(a^{\pm}\langle R\rangle.A)_S \quad \& \quad (a^{\mp}(X).B)_T \quad \to \quad A_S \quad \& \quad B\{R/X\}_T$
  if $a \in S\Box T$: (Com)

- $\dfrac{P \quad \rightarrow \quad P'}{P\&Q \quad \rightarrow \quad P'\&Q}$: (&) , $\dfrac{P \quad \rightarrow \quad P'}{(\nu a)P \quad \rightarrow \quad (\nu a)P'}$: (ν)
  and $\dfrac{P\{rec_x.P/X\} \quad \rightarrow \quad Q}{rec_X.P \quad \rightarrow \quad Q}$: (Rec)
- $\dfrac{A_S\&P \quad \rightarrow \quad A'_{S'}\&P'}{(A{+}B)_S\&P \quad \rightarrow \quad A'_{S'}\&P'}$ if $A \neq A'$: (Sum)
- $\dfrac{A_S\&P \quad \rightarrow \quad A'_{S'}\&P'}{(A|B)_S\&P \quad \rightarrow \quad (A'|B)_{S'}\&P'}$: (Par)
- $(\mathit{offer}(a).A)_S \quad \rightarrow \quad A_{S+a}$: (Offer)
- $(\mathit{retract}(a).A)_S \quad \rightarrow \quad A_{S-a}$ if $a \in S$: (Retract)
- $(\mathit{fork}(P).A)_S \quad \rightarrow \quad A_S \quad \& \quad P$: (Fork)

Rules Cong, &, ν and Rec are classical rules. Sum and Par are structural rules to handle choice and parallel operator on actions. Note that the & and Par rules are similar : both operators are parallel operators. The difference is that & is at the level of molecules and permits communication while | is in a molecule and just permits shuffle of actions. This is because there is no equivalent of the rule Com for the operator |.

Offer and Retract rules are basic modifications of interfaces. Fork is the synthesis of a new molecule.

The Com rule is remarkable. We ask that $a \in S \square T$ because we want to restrict communication to linked molecules. Only these molecules should be able to exchange information. This implies that communication occurs between an a^+ and a^-.

One can imagine fusing the two dualities input/output and a^+/a^- into one, and saying that outputs always come from an a^+, for instance. But this will restrict communication on a bond to only one direction, which is not acceptable.

2 Use of different connectors and examples

We present different examples of modeling basic biological systems in order to show why the different operators of the syntax are needed and to illustrate different mechanisms of the calculus.

2.1 Decomplexation : the graph structure of the complex is not forgotten

In order to model decomplexation, remembering which molecules are part of the complex is not sufficient. Indeed, if two parts of the complex are linked by only one bond and if this bond is removed, one obtain two separate complexes. Then, to determine which molecules are part of which complex we need to know which molecule was linked with which other molecule within the previous complex. We also have to prevent the decomplexation to add bonds within the complex. Thus, we need to remember the graph structure of the complex. The graph of a complex is built by taking molecules of the complex as nodes and bonds between molecules within the complex as arcs.

Here is a basic example illustrating how communications permit us to synchronyze the activity of molecules, and that we do not loose the graph structure of a complex. Let:

- $P = (\mathit{offer}(1^+).1^+\langle\rangle.\mathit{offer}(2^+).2^+\langle\rangle.2^+\langle\rangle.\mathit{retract}(1^+))\{\}$
- $Q = (\mathit{offer}(1^-).1^-\langle\rangle.\mathit{offer}(3^+).3^+\langle\rangle)\{\}$
- $R = ((\mathit{offer}(2^-)|\mathit{offer}(3^-)).(2^-\langle\rangle|3^-\langle\rangle).2^-\langle\rangle.\mathit{retract}(3^-))\{\}$

(P|Q).(P'|Q') is an abbreviation for P.Q.(P'|Q') + Q.P.(P'|Q')).
First P and Q offer and link on 1, then they synchronized so that they know they can try to link R. Then R offers 2 and 3 and synchronized with P and Q. Thanks to these synchronizations all the three molecules know that they are linked. P and R synchronize once again to agree on the release of Q.

P and R can release Q and stay connected. If they had not been connected within the complex (i.e. before Q is released) they would not have been connected after. This illustrates how the graph structure of a complex is remembered.

2.2 Associativity and NaCl

Let:

- $Na = (\mathit{offer}(e^+).Na')\{\}$
- $Cl = (\mathit{offer}(e^-).Cl')\{\}$
- $S = Na_1 \quad \& \quad Na_2 \quad \& \quad Cl_1 \quad \& \quad Cl_2$

We indexed molecules because we want to emphasize the variations permitted by the associativity of the &.
S can evolve by linking Na and Cl molecules. This models the ionization $Na + Cl \rightarrow Na^+ + Cl^-$:

$$\begin{array}{lllllll} S & \rightarrow^* & (& (Na'_1)_{e^+}\&(Na'_2)_{e^+} &) \;\&\; (& (Cl'_1)_{e^-}\&(Cl'_2)_{e^-} &) \\ & \equiv & (& (Na'_1)_{e^+}\&(Cl'_1)_{e^-} &) \;\&\; (& (Na'_2)_{e^+}\&(Cl'_2)_{e^-} &) \\ & \equiv & (& (Na'_1)_{e^+}\&(Cl'_2)_{e^-} &) \;\&\; (& (Na'_2)_{e^+}\&(Cl'_1)_{e^-} &) \end{array}$$

The difference between the last two lines is the pairs of linked molecules. In the last one we have (Na_1, Cl_2) and (Na_2, Cl_1) while in the other one we have (Na_1, Cl_1) and (Na_2, Cl_2). This is permitted by the rule of associativity in the structural congruence. In the first line one do not detail which Na^+ is linked with which Cl^-.

In theory these three systems are different, but in practice they are not distinguishable. Such links as $Na^+ - Cl^-$ are indeed always shifting from a molecule to another one. All different configurations are actually coexisting. Thus we do not want to distinguish them in our calculus.

2.3 Communication and strong links: Use of νa

Now we want to model stable links, corresponding to covalent bonds, for instance. When such a link is made, no rearrangement with other molecules is possible. The only way to remove such a link is to retract one of the linked sites. Let:

- $P = (\nu c)(\quad offer(p^-).p^-\langle c^+\rangle.retract(p^-).offer(c^-).$ $Work.retract(c^-).offer(p^-)\quad)_{\{\}}$
- $Q = (\quad offer(p^+).p^+(x).retract(p^+).offer(x).$ $Work'.retract(x).offer(p^+)\quad)_{\{\}}$
- $S = P\&Q$

Work and Work' are some actions to perform when the strong link is established. They begin and end by synchronization on the site c for P and x for Q (which are actually the same).

$$\begin{array}{rcl}
S & = & (\nu c)[\quad (offer(p^-).p^-\langle c^+\rangle.retract(p^-).offer(c^-). \\
 & & \quad Work.retract(c^-).offer(p^-))_{\{\}}\quad] \\
 & \& & (offer(p^+).p^+(x).retract(p^+).offer(x). \\
 & & \quad Work'.retract(x).offer(p^+))_{\{\}} \\
\xrightarrow{\tau,\tau} & & (\nu c)[\quad (p^-\langle c^+\rangle.retract(p^-).offer(c^-). \\
 & & \quad Work.retract(c^-).offer(p^-))_{\{p^-\}}\quad] \\
 & \&_{p^+,p^-} & (p^+(x).retract(p^+).offer(x). \\
 & & \quad Work'.retract(x).offer(p^+))_{\{p^+\}} \\
\xrightarrow{\tau} & & (\nu c)[\quad (retract(p^-).offer(c^-). \\
 & & \quad Work.retract(c^-).offer(p^-))_{\{p^-\}} \\
 & \&_{p^+,p^-} & (retract(p^+).offer(x). \\
 & & \quad Work'.retract(x).offer(p^+))_{\{p^+\}}\quad] \\
\xrightarrow{\tau,\ldots} & & (\nu c)[\quad (Work.retract(c^-).offer(p^-))_{\{c^-\}} \\
 & \&_{c^+,c^-} & (Work'.retract(x).offer(p^+))_{\{c^+\}}\quad]
\end{array}$$

The first step is achieved by offer of p^+ and p^-. The bond on site p is automatically made. The second one is due to a communication on p. The molecules exchange the new site c. Then they retract sites p^+ and p^- and offer c^+ and c^-. Even if there is some associativity rearrangement between the second and the third step, that is some P or Q link on site p with these molecules, the privacy of c ensures that these two molecules will eventually link. Indeed, they will eventually retract p^+ and p^- and offer c^+ and c^-. As c is private, no other molecule can make a bond on c. Thus, this bond on site c is strong and won't be broken unless one of them retracts.

2.4 Synthesis of Transcription Factors: $fork(P)$ and $rec_X.P$

When adding interfaces to processes to support complexation, we end up with processes "inside" interfaces. How then can an inner process trigger the creation of a new entity at the outer level ? This cannot be done by simple parallel composition, as usual, because now we are at the wrong level. Therefore we use a fork primitive to allow an "inner" process to create an "outer" entity.

We use a very basic model of transcription factors synthesis showing how $fork(P)$ models synthesis to a new molecule and how $rec_X.P$ models infinite behavior of molecules. Let:

- $dna_1 = rec_X. \quad (\quad tf_2^+().fork(TF_1 \& X) \quad)_{tf_2^+}$
- $TF_i = rec_Y. \quad (\quad tf_i^- \langle\rangle.Y + 0 \quad)_{tf_i^-}$ for i = 1,2.
- and $S = dna_1 \& TF_2$

After dna_1 has synchronized with TF_2 on the site tf_2, the fork action releases a TF_1 molecule which is the product of the reaction, and dna_1 molecule (because dna is not destroyed in the process).

3 Complexation of Membranes

We now want to represent membranes. The main step we take is to generalize molecules A_S to membranes $A[\![P]\!]_{S,T}$ where P is the contents inside the membrane, A is the activity of the membrane, and S, T are two interfaces: S for the outside and T for the inside of the membrane. The idea is that interactions can be offered to the outside ($a^\uparrow$) or to the inside ($a^\downarrow$) of a membrane. As in the case of molecules, the complexation operator, &, can create complexes of membranes by merging their (external) interfaces, while hiding the internal connections from view.

We have to design an oriented version of the calculus as in [3] because we do not want the shifting effects of the associativity congruence rule (see part 2.2) to occur between two sides of a membrane. Such a case would correspond to a receptor that keeps changing sides on its membrane and this is not relevant for biology. Moreover [3] has shown how oriented actions are better from both the biological and mathematical points of view.

3.1 Syntax

We take the notation of [3] for action of membranes: f stands for fuse (with f and $f^{\perp}$ as complementary actions) w for wrap and b for bubble. We use $a^{\updownarrow}$ to stand for either $a^{\uparrow}$ or $a^{\downarrow}$.

- $a, b, c, \ldots \in \Sigma^{-} \bigcup \Sigma^{+}$ and $S, T, \ldots \in Multisets(\Sigma^{-} \bigcup \Sigma^{+})$
- $P, Q, R, \ldots ::= \quad A[\![P]\!]_{S,T} \quad | \quad P\&P \quad | \quad (\nu a)P \quad | \quad X \quad | \quad rec_X.P$
- $A, B, C, \ldots ::= \quad 0 \quad | \quad A+B \quad | \quad A|B \quad | \quad \alpha^{\uparrow}.A \quad | \quad \alpha^{\downarrow}.A$
- $\alpha ::= \quad a\langle S\rangle \quad | \quad a(S) \quad | \quad \mathit{offer}(a) \quad | \quad \mathit{retract}(a) \quad | \quad [S]$
  $| \quad f \quad | \quad f^{\perp} \quad | \quad w \quad | \quad w(A[\![]\!]_{S,T}) \quad | \quad b(A[\![]\!]_{S,T})$

Arrows on actions give orientation to actions. $[S]$ is the action corresponding to Bind&Release of [1]: it permits molecules to cross membranes. Here, however, we do not match on names of molecules but on interfaces. A whole complex of molecules, for example, can be transported across the membrane by an $[S]$ action that recognizes its external interface. Actions *f, w and b* are for membranes reactions. They are precisely described in [1, 3]. We no longer need the fork action because of the pino action that plays its role. Interfaces are now defined by:

$$I(A[\![P]\!]_{S,T}) = S, \qquad I((\nu a)P) = I(P) - a, \qquad I(P\&Q) = I(P) \amalg I(Q)$$
$$I(X) = \emptyset, \qquad I(rec_X.P) = I(P)$$

3.2 Free names

We have to define free names for the new operators:

- $fn(A[\![P]\!]_{S,T}) = fn(A) \bigcup fn(P) \bigcup Set(S) \bigcup Set(T)$
- $fn(f^{\updownarrow}.A) = fn(f^{\perp\updownarrow}.A) = fn(w^{\updownarrow}.A) = fn(A)$
- $fn(w^{\updownarrow}(B[\![]\!]_S).A) = fn(b^{\updownarrow}(B[\![]\!]_S).A) = fn(A) \bigcup fn(B) \bigcup Set(S)$
- $fn([S]^{\updownarrow}.A) = fn(A) \bigcup Set(S)$

3.3 Structural Congruence

Some rules need to be added to or changed in the structural congruence:

- $\frac{A \equiv B \quad P \equiv Q}{A[\![P]\!]_S \equiv B[\![Q]\!]_S}$
- $\frac{A \equiv B \quad \alpha \equiv \beta}{\alpha^{\updownarrow}.A \equiv \beta^{\updownarrow}.B}$
- $\alpha \equiv \alpha$, $\frac{A \equiv B}{w(A[\![]\!]_S) \equiv w(B[\![]\!]_S)}$ and $\frac{A \equiv B}{b(A[\![]\!]_S) \equiv b(B[\![]\!]_S)}$

3.4 Operational Semantics

The list of reactions is a bit long. This is partially because orientation doubles reactions.

These are the two rules for communication. Com1 is just the extension to the oriented case of the rule Com of the first calculus. Com2 is the case of communication between nested membranes. Of course for Com2 there is also the version where input and output are exchanged.

- $(a^{\pm\uparrow}[\![R]\!].A)[\![P]\!]_{S,S'} \quad \& \quad (a^{\mp\uparrow}(X).B)[\![Q]\!]_{T,T'}$
  $\rightarrow \quad A[\![P]\!]_{S,S'} \quad \& \quad B\{R/X\}[\![Q]\!]_{T,T'}$ if $a \in S \Box T$: (Com1)
- $(a^{\pm\downarrow}\langle R\rangle.A)[\![\quad (a^{\mp\uparrow}(U).B)[\![Q]\!]_{T,T'}, Q' \quad]\!]_{S,S'}$
  $\rightarrow \quad A[\![\quad B\{R/U\}[\![Q]\!]_{T,T'}, Q' \quad]\!]_{S,S'}$ if $a \in S'\Box T$: (Com2)

The following rules are just extensions to the oriented case of the Offer and Retract rules.

- $(\mathit{offer}(a)^{\uparrow}.A)[\![P]\!]_{S,T} \quad \rightarrow \quad A_{S+a,T}$: (Off-Out)
- $(\mathit{offer}(a)^{\downarrow}.A)[\![P]\!]_{S,T} \quad \rightarrow \quad A_{S,T+a}$: (Off-In)
- $(\mathit{retract}(a)^{\uparrow}.A)[\![P]\!]_{S,T} \quad \rightarrow \quad A_{S-a,T}$ if $a \in S$: (Retract-Out)
- $(\mathit{retract}(a)^{\downarrow}.A)[\![P]\!]_{S,T} \quad \rightarrow \quad A_{S,T-a}$ if $a \in T$: (Retract-In)

The following rules just look for a molecule with the proper interface and push it across the membrane. The content of A is empty because we want it to be a molecule.

- $A[\![]\!]_{U,U'}\&([S]^{\uparrow}.B)[\![R]\!]_{T,T'} \quad \rightarrow \quad (B)[\![A[\![]\!]_{U,U'}\&R]\!]_{T,T'}$ if $S = U$: (Through-out)
- $([S]^{\downarrow}.B)[\![A[\![]\!]_{U,U'}\&R]\!]_{T,T'} \quad \rightarrow \quad A[\![]\!]_{U,U'}\&(B)[\![R]\!]_{T,T'}$ if $S = U$: (Through-in)

Emission of a new membrane inside or outside. These are Drip and Pino reactions.

- $(b^{\uparrow}(A[\![]\!]_{S,S'}).B)[\![P]\!]_{T,T'} \quad \rightarrow \quad B[\![P]\!]_{T,T'} \quad \& \quad A[\![]\!]_{S,S'}$: (Drip)
- $(b^{\downarrow}(A[\![]\!]_{S,S'}).B)[\![P]\!]_{T,T'} \quad \rightarrow \quad B[\![P \quad \& \quad A[\![]\!]_{S,S'}]\!]_{T,T'}$: (Pino)

Reactions Phago, Bud and Swap. In each of them a membrane engulfs itself in another membrane, wrapping itself in a piece of new membrane. The Swap reaction could seem strange. Actually Swap is Phago or Bud reaction seen from the point of view of P. See [3] for more details.

- $(w^{\uparrow}.A)[\![P]\!]_{S,S'} \quad \& \quad (w^{\uparrow}(C[\![]\!]_{T,T'}).B)[\![Q]\!]_{U,U'}$
  $\rightarrow \quad B[\![C[\![A[\![P]\!]_{S,S'}]\!]_{T,T'} \quad \& \quad Q]\!]_{U,U'}$: (Phago)
- $(w^{\downarrow}(C[\![]\!]_{T,T'}).B)[\![(w^{\uparrow}.A)[\![P]\!]_{S,S'} \quad \& \quad Q]\!]_{U,U'}$
  $\rightarrow \quad B[\![Q]\!]_{U,U'} \quad \& \quad C[\![A[\![P]\!]_{S,S'}]\!]_{T,T'}$: (Bud)
- $(w^{\downarrow}.A)[\![R \quad \& \quad (w^{\uparrow}(C[\![]\!]_{U,U'}).B)[\![Q]\!]_{T,T'}]\!]_{S,S'}$
  $\rightarrow \quad A[\![C[\![B[\![R]\!]_{T,T'} \quad \& \quad Q]\!]_{U,U'}]\!]_{S,S'}$: (Swap)

The following Exo and Mate reactions are both fusions of membranes either vertical (Exo) or horizontal (Mate). Note that Exo reverses orientations of actions in B and adds T' to S and T to S'.

- $(f^{\perp\downarrow}.A)[\![(f^{\uparrow}.B)[\![P]\!]_{T,T'}\&Q]\!]_{S,S'} \quad \rightarrow \quad P \quad \& \quad (A|\overline{B})[\![Q]\!]_{S+T',S'+T}$
  where $\overline{B}$ just reverses orientations of actions in B : (Exo)
- $(f^{\perp\uparrow}.A)[\![P]\!]_{S,S'}\&(f^{\uparrow}.B)[\![Q]\!]_{T,T'} \quad \rightarrow \quad (A|B)[\![P\&Q]\!]_{S+T,S'+T'}$:(Mate)

The structural rules Sum and Par extend to:

- $\dfrac{A[\![P]\!]_{S,S'} \ \& \ R \quad \rightarrow \quad A'[\![P']\!]_{T,T'} \ \& \ R'}{(A|C)[\![P]\!]_{S,S'} \ \& \ R \quad \rightarrow \quad (A'|C)[\![P']\!]_{T,T'} \ \& \ R'}$: (Par)
- $\dfrac{A[\![P]\!]_{S,S'} \ \& \ R \quad \rightarrow \quad A'[\![P']\!]_{T,T'} \ \& \ R'}{(A+C)[\![P]\!]_{S,S'} \ \& \ R \quad \rightarrow \quad A'[\![P']\!]_{T,T'} \ \& \ R'}$ if $A \neq A'$: (Sum)

3.5 Modeling of a molecule repaired in a Golgi apparatus

We next present a small example to illustrate interactions between membranes and molecules. Let:

- $M = (\quad damaged^{+\uparrow}\langle\rangle.retract(damaged^{+}).$
  $offer(undamaged^{+}) \quad)[\![]\!]_{damaged^{+},\_}$
- $G = (\quad golgi|[damaged^{+}]^{\uparrow}|[undamaged^{+}]^{\downarrow} \quad)[\![$
  $(damaged^{-}()^{\uparrow})[\![]\!]_{damaged^{-},\_} \quad]\!]_{\_,\_}$
- $S = M\&G$

M is a damaged molecule. G is a golgi apparatus containing a molecule, which can repair M. So M has to go inside G to be repaired. Since M has the interface $damaged^{+}$, G can use its action $[damaged]^{\uparrow}$ to capture M and put it into its content. The content of G is then:
$(damaged^{+\uparrow}\langle\rangle.retract(damaged^{+}).offer(undamaged^{+}))[\![]\!]_{damaged^{+},\_}$
$\& \quad (damaged^{-}()^{\uparrow})[\![]\!]_{damaged^{-},\_}$

The synchronization on the site *damaged*, followed by the retracting of $damaged^{+}$ by M and the offering of $undamaged^{+}$, represent the repairing of M. The interface of M is now $undamaged^{+}$ and it can be released outside of G.

4 Possible extensions and Perspectives

4.1 Possible extensions

Different extensions are possible to enhance the syntax. A first interesting feature consists of test actions $?(a)$ and $\neg?(a)$ with theses rules:

- $(?(a).A)[\![P]\!]_{S,S'}\&Q \quad \rightarrow \quad (A)[\![P]\!]_{S,S'}\&Q \quad : \quad \text{if } a \in S\Box I(Q)$
- $(\neg?(a).A)[\![P]\!]_{S,S'}\&Q \quad \rightarrow \quad (A)[\![P]\!]_{S,S'}\&Q \quad : \quad \text{if } a \notin S\Box I(Q)$

Tests can be useful because synchronization does not a molecule permit a molecule to check if some site is not connected. It can also be a simpler way to check if a site is connected without requiring one to "program" both sides of the link.

Another extension would be to have molecules on membranes. In this work, a molecule can only be between membranes. So we loose the possibility of representing transmembranal molecules. A simple solution is to permit terms of the first calculus between membranes and on membranes. To compute the interface of a membrane we then have to add all interfaces of molecules sitting on this membrane.

4.2 Perspectives

An interesting perspective is to design a projective version of the calculus as in [3]; this means building a "projective" equivalence relation that identifies molecules representing the same system under different points of view. This equivalence has to be compatible with reduction; this implies that each reaction must have a "symmetric" reaction with respect to the projective equivalence. This is almost achieved in the calculus we presented here. For instance, the two communication rules are "symmetric" : they correspond to the same system seen from two different points of view. The only problem is the ν operator. Taking a point of view inside a ν requires a symmetric operator for ν. If one thinks of ν as the beginning of a scope, its symmetrical operator would end the scope.

Another perspective is to reduce the calculus to just what we need. For instance, examples do not use the full power of name passing communications. Our conjecture is that bounded outputs are sufficient to express biological interactions.

Finally a stochastic semantics is needed as a basis for simulations.

References

[1] Luca Cardelli. Brane calculi. *Computational Methods in Systems Biology: Second International Workshop, CMSB'04*, 3082:257–280, April 2004.

[2] Luca Cardelli and Andrew Gordon. Mobile ambients. *Theoritical Computer Science*, 240/1:177–213, 2000.

[3] Vincent Danos and Sylvain Pradalier. Projective Brane-calculus. *Computational Methods in Systems Biology: Second International Workshop, CMSB'04*, 3082:134–148, April 2004.

[4] A.Regev W.Silverman E.Shapiro. Representation and simulation of biochemical processes using the pi-calculus process algebra. 2001.

[5] Robin Milner. *Communicating and Mobile Systems*. Cambridge University Press.

[6] Corrado Priami and Paola Quaglia. Beta binders for biological interactions. *Computational Methods in Systems Biology: Second International Workshop, CMSB'04*, April 2004.

[7] Corrado Priami and Paola Quaglia. Operational patterns in beta binders. 2005.

[8] C.Priami A.Regev E.Shapiro W.Silverman. Application of a stochastic name-passing calculus to representation and simulation of molecular processes. 2001.