# Mobility in Computer Science and in Membrane Systems

Gabriel Ciobanu

"A.I.Cuza" University of Iași, and
Romanian Academy, Institute of Computer Science, Iași
gabriel@info.uaic.ro

- joint work with Bogdan Aman -

Jena, 24th August 2010

In memory of Robin Milner (1934-2010)

# Moving Links in a Virtual Space of Links

When expressing mobility, we should mention what entities move and in what space they move.

## Several possibilities:

- processes moving in a physical space of computing locations,
- processes moving in a virtual space of linked processes,
- links moving in a virtual space of linked processes ...

– The $\pi$-calculus is a formalism where links are the moving entities, and they move in a virtual space of linked processes (the network of web pages is a good example for this approach).

– This option can express moving processes both in a physical space of computing locations and in a virtual space of linked processes [Milner99].

# Pi-Calculus

Computational world of the $\pi$-calculus:

- processes (also called agents);
- channels (also called names or ports).

The $\pi$-calculus models networks in which messages are sent from one site to another, and may contains links to active processes or to other sites.

- channels are passed as data along other channels, and this provides the changing configurations and connectivity among processes;
- this mobility increases the expressive power enabling the description of many high-level concurrent features.

## General Model of Computation

- widely accepted model of interacting systems with dynamically evolving communication topology (mobility);
- a general model of computation taking interaction as primitive (it extends the Church-Turing model by extending the $\lambda$-calculus with "elements of interaction").

$\pi$-calculus has a simple semantics and a tractable algebraic theory.

## Syntax

$$P ::= 0 \mid \overline{x}\langle z\rangle.P \mid x(y).P \mid P \mid Q \mid P + Q \mid !P \mid \nu x\ P$$

0 is the empty process, guarded processes $\overline{x}\langle z\rangle.P$ and $x(y).P$, parallel composition $P \mid Q$, nondeterministic choice $P + Q$, replication $!P$, restriction $\nu x\ P$ creating a local fresh channel $x$ for the process $P$.

## Processes interact by using names (channels) they share

A name received in one interaction can be used in another; by receiving a name, a process can interact with processes which are unknown to it, but now they share the same channel name.

## Semantics

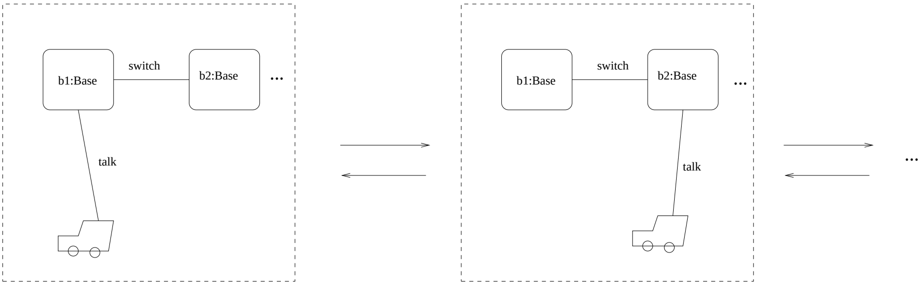
The *reduction* relation over processes is defined as the smallest relation $\rightarrow$ satisfying the following rules

| | |
|---|---|
| $(com)$ | $(\overline{x}\langle z\rangle.P + R_1) \mid (\,x(y).Q + R_2) \rightarrow P \mid Q\{z/y\}$ |
| $(par)$ | $P \rightarrow Q$ implies $P \mid R \rightarrow Q \mid R$ |
| $(res)$ | $P \rightarrow Q$ implies $(\nu x)P \rightarrow (\nu x)Q$ |
| $(str)$ | $P \equiv P',\ P' \rightarrow Q'$ and $Q' \equiv Q$ implies $P \rightarrow Q$ |

where $\equiv$ is a structural congruence relation defined as the smallest congruence over the set of processes which satisfies

- $P \equiv Q$ if $P =_\alpha Q$
- $P + 0 \equiv P,\ P + Q \equiv Q + P,\ (P + Q) + R \equiv P + (Q + R),$
- $P \mid 0 \equiv P,\ P \mid Q \equiv Q \mid P,\ (P \mid Q) \mid R \equiv P \mid (Q \mid R),$
- $!P \equiv P \mid !P$
- $\nu x0 \equiv 0,\ \nu x\nu yP \equiv \nu y\nu xP,\ \nu x(P \mid Q) \equiv P \mid \nu xQ$ if $x \notin fn(P)$.

# An Example




- $\nu\ talk\ (B_1 \mid C) \mid B_2, \quad B_1 = \overline{switch}\langle talk \rangle.B_1', \quad B_2 = switch(y).B_2'$
- if $talk \notin fn(B_1')$, then $B_1'$ will lose its link to $C$ :
- $\nu\ talk\ (B_1 \mid C) \mid B_2 \ \longrightarrow\ B_1' \mid \nu\ talk\ (C \mid B_2'')$

Mobility = scoping names + extrusion of names from their scope.

- using labeled transition system defined by the the reduction rules, several behavioural equivalences are defined based on bisimulation;
- verification technique for proving properties about the mobile concurrent systems modeled in the $\pi$-calculus (protocol verification);
- properties of finite state transition systems can be described in a powerful logic called $\mu$-calculus;
- Mobility Workbench [Victor94] supports open bisimulation checking, as well as model checking $\pi$-calculus processes.

Several variants of $\pi$-calculus: Spi, Dpi, tDpi, AppliedPi, ... bigraphs.

Regev and Shapiro use $\pi$-calculus in describing biochemical systems (representation, simulation, and analysis of metabolic pathways).
**"molecule-as-computation"**: $\pi$-calculus processes as abstractions of molecules in biomolecular systems.

# Ambient Calculus

Ambient calculus [CardelliGordon98] describe computation carried out on mobile devices (i.e. networks having a dynamic topology), and mobile computation (i.e. executable code able to move around the network).

## Ambients

- primitive of the ambient calculus is the ambient;
- defined as a bounded place in which computation can occur; ambients have names used to control access to the ambient; ambients can be nested inside other ambients.

## Computation

- ambients can be moved as a whole, changing their location by consuming certain capabilities: **in**, **out**, and **open**;
- these basic ambient operations are expressive enough to simulate name-passing channels in the $\pi$-calculus;
- computation is represented as the movement of ambients.

Considering an infinite set of names $\mathcal{N}$ $(m, n, \dots)$ we define MA-processes $(A, A', B, B' \dots)$ together with their capabilities $(C, C', \dots)$:

$$C \quad ::= \quad in\ n \quad | \quad out\ n \quad | \quad open\ n$$
$$A \quad ::= \quad \mathbf{0} \quad | \quad C.A \quad | \quad n[A] \quad | \quad A \mid B \quad | \quad (\nu n)A$$

## Axioms and Rules:

**Axioms:**

$$(In) \qquad n[in\ m.A \mid A'] \mid m[B] \Rightarrow_{amb} m[n[A \mid A'] \mid B]\ ;$$
$$(Out) \qquad m[n[out\ m.A \mid A'] \mid B] \Rightarrow_{amb} n[A \mid A'] \mid m[B]\ ;$$
$$(Open) \qquad open\ n.A \mid n[B] \Rightarrow_{amb} A \mid B\ .$$

**Rules:**

$$(Res)\ \frac{A \Rightarrow_{amb} A'}{(\nu n)A \Rightarrow_{amb} (\nu n)A'}\ ; \qquad (Comp)\ \frac{A \Rightarrow_{amb} A'}{A \mid B \Rightarrow_{amb} A' \mid B}\ ;$$

$$(Amb)\ \frac{A \Rightarrow_{amb} A'}{n[A] \Rightarrow_{amb} n[A']}\ ; \quad (Struc)\ \frac{A \equiv_{amb} A',\ A' \Rightarrow_{amb} B',\ B' \equiv_{amb} B}{A \Rightarrow_{amb} B}$$

## Syntax and Semantics [Levi03]

$$C \quad ::= \quad in\ n \mid \overline{in}\ n \mid out\ n \mid \overline{out}\ n \mid open\ n \mid \overline{open}\ n$$
$$A \quad ::= \quad \mathbf{0} \mid A \mid B \mid C.A \mid n[\,A\,] \mid (\nu n)A$$

**Axioms:**

*(In)* $n[\,in\ m.A \mid A'\,] \mid m[\,\overline{in}\ m.B \mid B'\,] \Rightarrow_{amb} m[\,n[\,A \mid A'\,] \mid B \mid B'\,]$;

*(Out)* $m[\,n[\,out\ m.A \mid A'\,] \mid \overline{out}\ m.B \mid B'\,] \Rightarrow_{amb} n[\,A \mid A'\,] \mid m[\,B \mid B'\,]$;

*(Open)* $open\ n.A \mid n[\,\overline{open}\ n.B \mid B'\,] \Rightarrow_{amb} A \mid B \mid B'$ .

**Rules:**

*(Res)* $\dfrac{A \Rightarrow_{amb} A'}{(\nu n)A \Rightarrow_{amb} (\nu n)A'}$ ; *(Comp)* $\dfrac{A \Rightarrow_{amb} A'}{A \mid B \Rightarrow_{amb} A' \mid B}$;

*(Amb)* $\dfrac{A \Rightarrow_{amb} A'}{n[\,A\,] \Rightarrow_{amb} n[\,A'\,]}$ ; *(Struc)* $\dfrac{A \equiv A',\ A' \Rightarrow_{amb} B',\ B' \equiv B}{A \Rightarrow_{amb} B}$.

# Brane Calculus

Proteins are embedded in membranes, and can act on both sides of the membrane simultaneously. Brane calculus [Cardelli04] use both sides of the membrane, emphasizing that computation happens also on the membrane surface. The new operations are inspired by biologic processes *endocytosis*, *exocytosis* and *mitosis*.

- PEP calculus: operations *pino*, *exo*, *phago*,
- MBD calculus: operations *mate*, *drip*, *bud*,
- MBD can be simulated by PEP [Cardelli04].

## Syntax

| | | |
|---|---|---|
| *Systems* | $P, Q ::= \diamond \mid P \circ Q \mid \sigma(P)$ | nests of membranes |
| *Branes* | $\sigma, \tau ::= O \mid \sigma\vert\tau \mid a.\sigma$ | combinations of actions |
| *Actions* | $a, b ::= n^{\searrow} \mid \overline{n}^{\searrow}(\sigma) \mid n^{\nwarrow} \mid \overline{n}^{\nwarrow} \mid pino(\sigma)$ | phago $\searrow$, exo $\nwarrow$ |

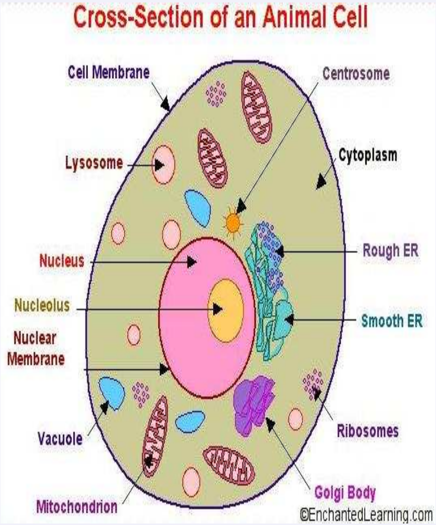
## Semantics

| | |
|---|---|
| $pino(\rho).\sigma\vert\sigma_0(P) \Rightarrow \sigma\vert\sigma_0(\rho(\diamond) \circ P)$ | Pino |
| $\overline{n}^{\nwarrow}.\tau\vert\tau_0(n^{\nwarrow}.\sigma\vert\sigma_0(P) \circ Q) \Rightarrow P \circ \sigma\vert\sigma_0\vert\tau\vert\tau_0(Q)$ | Exo |
| $n^{\searrow}.\sigma\vert\sigma_0(P) \circ \overline{n}^{\searrow}(\rho).\tau\vert\tau_0(Q) \Rightarrow \tau\vert\tau_0(\rho(\sigma\vert\sigma_0(P)) \circ Q)$ | Phago |
| $P \Rightarrow Q$ implies $P \circ R \Rightarrow Q \circ R$ | Par |
| $P \Rightarrow Q$ implies $\sigma(P) \Rightarrow \sigma(Q)$ | Mem |
| $P \equiv_b P'$ and $P' \Rightarrow Q'$ and $Q' \equiv_b Q$ implies $P \Rightarrow Q$ | Struct |

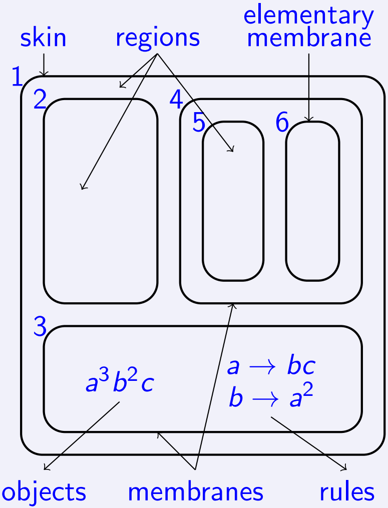
# Outline

# From Cell to Membrane Systems

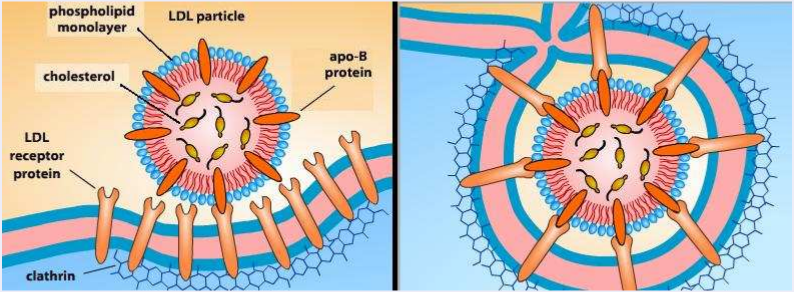
## Cell



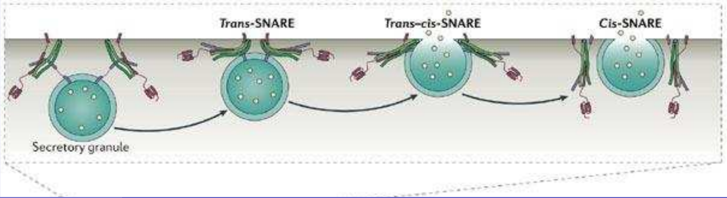

## Membrane System



## Receptor-Mediated Endocytosis

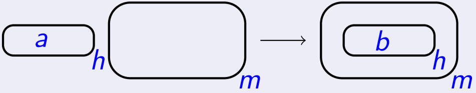



## SNARE-Mediated Exocytosis

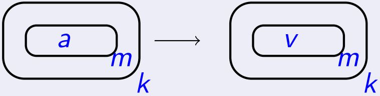


# Simple Mobile Membranes
## Evolution Rules

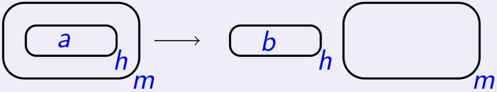
### Endocytosis: $[a]_h[\ ]_m \rightarrow [[b]_h]_m$




### Local Object Evolution: $[[a]_m]_k \rightarrow [[v]_m]_k$




### Exocytosis: $[[a]_h]_m \rightarrow [b]_h[\ ]_m$



**Theorem (KrishnaPaun05)**

*Simple mobile membranes with nine membranes using rules of types (gevol), (endo), (exo) have the same computational power as a Turing Machine.*

**Theorem (Krishna05)**

*Simple mobile membranes with four membranes using rules of types (gevol), (endo), (exo) have the same computational power as a Turing Machine.*

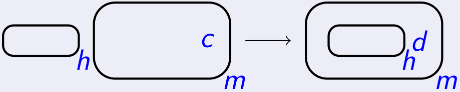
**Theorem (AmanCiobanu08)**

*Simple mobile membranes with three membranes using rules of types (levol), (endo), (exo) have the same computational power as a Turing Machine.*

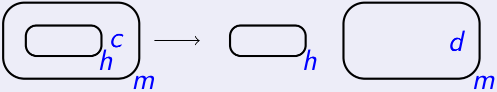
# Enhanced Mobile Membranes

## Evolution Rules

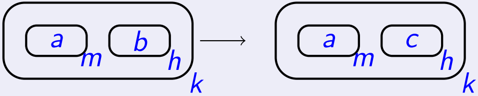
### Enhanced Endocytosis: $[\ ]_h[c]_m \rightarrow [[\ ]_h d]_m$




### Enhanced Exocytosis: $[[\ ]_h c]_m \rightarrow [\ ]_h[d]_m$




### Contextual Evolution: $[[a]_m[b]_h]_k \rightarrow [[a]_m[c]_h]_k$



**Theorem (KrishnaCiobanu08)**

*Simple mobile membranes with three membranes using rules of types (cevol) have the same computational power as a Turing Machine.*

**Theorem (KrishnaCiobanu08)**

*Enhanced mobile membranes with twelve membranes using rules of types (endo), (exo), (fendo), (fexo) have the same computational power as a Turing Machine.*

**Theorem (AmanCiobanu08)**

*Enhanced mobile membranes with nine membranes using rules of types (endo), (exo), (fendo), (fexo) have the same computational power as a Turing Machine.*

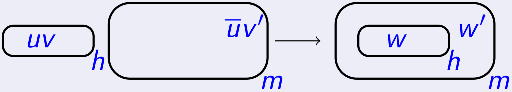
# Mutual Mobile Membranes

## Evolution Rules

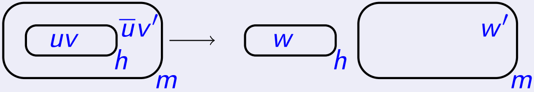
### Theorem (KrishnaCiobanu08)

*Enhanced mobile membranes with three membranes using rules of types (endo), (exo) have the same computationally power as enhanced mobile membranes with three membranes using rules of types (fendo), (fexo).*

### Mutual Endocytosis: $[uv]_h[\overline{u}v']_m \rightarrow [\ [w]_h w']_m$




### Mutual Exocytosis: $[[uv]_h\overline{u}v']_m \rightarrow [w]_h[w']_m$



**Theorem (AmanCiobanu09)**

*Mutual mobile membranes with three membranes using rules of types (mendo), (mexo) have the same power as a Turing Machine.*

**Proposition**

*Mutual mobile membranes with three membranes using rules of types (mendo), (mexo) subsume the families of languages generated by extended tabled 0L systems (ET0L).*

# Outline

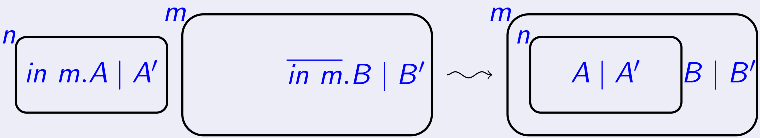
# Safe Mobile Ambients [Levi&Sangiorgi03]

## Syntax

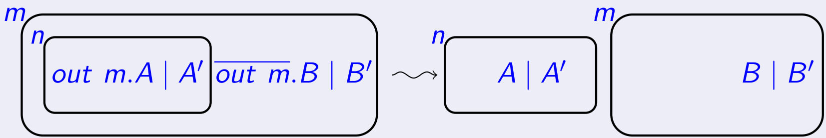
$$C \quad ::= \quad in\ n \quad | \quad \overline{in}\ n \quad | \quad out\ n \quad | \quad \overline{out}\ n$$

$$A \quad ::= \quad \mathbf{0} \quad | \quad A \mid B \quad | \quad C.A \quad | \quad n[\,A\,] \quad | \quad (\nu n)A$$

## In: $n[in\ m.A|A']|m[\overline{in}\ m.B|B'] \rightsquigarrow m[n[A|A']|B|B']$




## Out: $m[n[out\ m.A|A']|\overline{out}\ m.B|B'] \rightsquigarrow n[A|A']|m[B|B']$



# Encoding Safe Ambients into Mutual Mobile Membranes

## Definition

A translation $\mathcal{T} : \mathcal{SA} \to \mathcal{M}^3$ is given by $\mathcal{T}(A) = dlock\ \mathcal{T}_1(A)$, where $\mathcal{T}_1 : \mathcal{SA} \to \mathcal{M}^3$ is

$$\mathcal{T}_1(A) = \begin{cases} cap\ n[\ ]_{cap\ n} & \text{if } A = cap\ n \\ cap\ n[\ \mathcal{T}_1(A_1)\ ]_{cap\ n} & \text{if } A = cap\ n.\, A_1 \\ [\ \mathcal{T}_1(A_1)\ ]_n & \text{if } A = n[\ A_1\ ] \\ [\ ]_n & \text{if } A = n[\ ] \\ \mathcal{T}_1(A_1), \mathcal{T}_1(A_2) & \text{if } A = A_1 \mid A_2 \end{cases}$$

## Theorem (Operational Correspondence)

1. *If $A \rightsquigarrow B$, then $\mathcal{T}(A) \to \mathcal{T}(B)$.*
2. *If $\mathcal{T}(A) \to M$, then exists $B$ such that $A \rightsquigarrow B$ and $M = \mathcal{T}(B)$.*

# Decidability Result

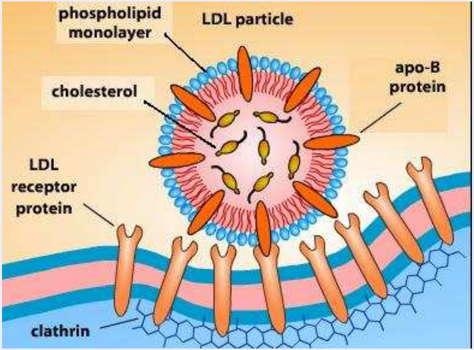
**Theorem (AmanCiobanu08)**

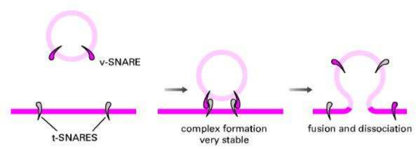
*For two arbitrary mobile membranes $M_1$ and $M_2$, it is decidable whether $M_1$ reduces to $M_2$.*

The main steps of the proof are as follows:

1. mobile membranes systems are reduced to a fragment of mobile ambients;
2. the reachability problem for two arbitrary mobile membranes can be expressed as the reachability problem for the corresponding mobile ambients;
3. the reachability problem is decidable for a fragment of mobile ambients by reducing it to the reachability problem in Petri nets.

# Outline

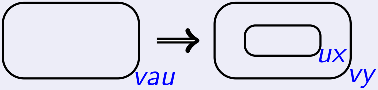


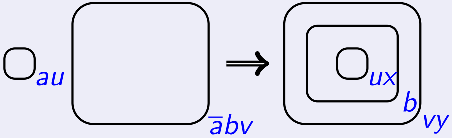

SNAREs and vesicle fusion

- Some proteins on the surface of the cell membrane serve as "markers" that identify a cell to other cells.
- The interaction of these markers with their respective receptors forms the basis of cell-cell interaction in the immune system.

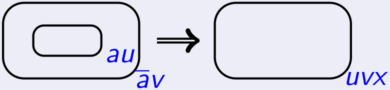
**Pino:** $[\,]_{vau} \Rightarrow [[\,]_{ux}]_{vy}$




**Phago:** $[\,]_{au}[\,]_{\bar{a}bv} \Rightarrow [[[\,]_{ux}]_{b}]_{vy}$




**Exo:** $[[\,]_{au}]_{\bar{a}v} \Rightarrow [\,]_{uvx}$



The number of objects from the right-hand of a rule is called its *weight*.

## Summary of Results

| Number $n$ of membranes | Operations $(op_1, op_2)$ | Weights $(w_1, w_2)$ | Article |
|---|---|---|---|
| 8 | Pino, exo | 4,3 | Theorem 6.1 [Krishna07] |
| 3 | Pino, exo | 5,4 | Theorem 1 [AmanCiobanu09] |
| 9 | Phago, exo | 5,2 | Theorem 6.2 [Krishna07] |
| 9 | Phago, exo | 4,3 | Theorem 6.2 [Krishna07] |
| 4 | Phago, exo | 6,3 | Theorem 2 [AmanCiobanu09] |

## Theorem (A Family of Results)

*Mutual membranes with objects on surface using n membranes and operations $(op_1, op_2)$ of weights $(w_1, w_2)$ have the same computational power as a Turing Machine.*

## Syntax of PEP

| | | | |
|---|---|---|---|
| *Systems* | $P, Q ::= \diamond \mid P \circ Q \mid \sigma(P)$ | | nests of membranes |
| *Branes* | $\sigma, \tau ::= O \mid \sigma\vert\tau \mid a.\sigma$ | | combinations of actions |
| *Actions* | $a, b ::= n^{\searrow} \mid \overline{n}^{\searrow}(\sigma) \mid n^{\nwarrow} \mid \overline{n}^{\nwarrow} \mid pino(\sigma)$ | | phago $\searrow$, exo $\nwarrow$ |

## Reductions of PEP

| | |
|---|---|
| $pino(\rho).\sigma\vert\sigma_0(P) \Rightarrow \sigma\vert\sigma_0(\rho(\diamond) \circ P)$ | Pino |
| $\overline{n}^{\nwarrow}.\tau\vert\tau_0(n^{\nwarrow}.\sigma\vert\sigma_0(P) \circ Q) \Rightarrow P \circ \sigma\vert\sigma_0\vert\tau\vert\tau_0(Q)$ | Exo |
| $n^{\searrow}.\sigma\vert\sigma_0(P) \circ \overline{n}^{\searrow}(\rho).\tau\vert\tau_0(Q) \Rightarrow \tau\vert\tau_0(\rho(\sigma\vert\sigma_0(P)) \circ Q)$ | Phago |
| $P \Rightarrow Q$ implies $P \circ R \Rightarrow Q \circ R$ | Par |
| $P \Rightarrow Q$ implies $\sigma(P) \Rightarrow \sigma(Q)$ | Mem |
| $P \equiv_b P'$ and $P' \Rightarrow Q'$ and $Q' \equiv_b Q$ implies $P \Rightarrow Q$ | Struct |

## Definition

A translation $\mathcal{T} : \mathcal{PEP} \to \mathcal{M}^2\mathcal{OS}$ is given by

$$\mathcal{T}(P) = \begin{cases} [\mathcal{T}(P)]_{\mathcal{S}(\sigma)} & \text{if } \sigma(P) \\ \mathcal{T}(Q)\ \mathcal{T}(R) & \text{if } P = Q \,|\, R \end{cases}$$

where $\mathcal{S} : \mathcal{PEP} \to \mathcal{M}^2\mathcal{OS}$ is defined as:

$$\mathcal{S}(\sigma) = \begin{cases} \sigma & \text{if } \sigma = n^{\searrow} \text{ or } \sigma = n^{\nwarrow} \text{ or } \sigma = \overline{n}^{\nwarrow} \\ \overline{n}^{\searrow}\ S(\rho) & \text{if } \sigma = \overline{n}^{\searrow}(\rho) \\ pino\ S(\rho) & \text{if } \sigma = pino(\rho) \\ \mathcal{S}(a)\ \mathcal{S}(\rho) & \text{if } \sigma = a.\rho \\ \mathcal{S}(\tau)\ \mathcal{S}(\rho) & \text{if } \sigma = \tau \,|\, \rho \end{cases}$$

## Rules of M$^2$OS

$[\ ]_{S(n^{\searrow}\sigma|\sigma_0)}[\ ]_{S(\overline{n}^{\searrow}(\rho).\tau|\tau_0)} \to [[[\ ]_{S(\sigma|\sigma_0)}]_{S(\rho)}]_{S(\tau|\tau_0)}$

$[[\ ]_{S(n^{\nwarrow}.\sigma|\sigma_0)}]_{S(\overline{n}^{\nwarrow}.\tau|\tau_0)} \to [\ ]_{S(\sigma|\sigma_0|\tau|\tau_0)}$

$[\ ]_{S(pino(\rho).\sigma|\sigma_0)} \to [[\ ]_{S(\rho)}]_{S(\sigma|\sigma_0)}$

# Encoding PEP into $M^2OS$

**Proposition**

1. *If* $P \equiv_b Q$ *then* $\mathcal{T}(P) \equiv_m \mathcal{T}(Q)$.
2. *If* $\mathcal{T}(P) \equiv_m M$ *then there exists* $Q$ *such that* $M = \mathcal{T}(Q)$.

**Theorem (Operational Correspondence)**

1. *If* $P \Rightarrow Q$ *then* $\mathcal{T}(P) \to \mathcal{T}(Q)$.
2. *If* $\mathcal{T}(P) \to M$ *then there exists* $Q$ *such that* $P \to_b Q$ *and* $M \equiv_m \mathcal{T}(Q)$.

# Conclusion

- various notions of mobility in computer science: $\pi$-calculus, distributed $\pi$-calculus, mobile ambients, brane calculi;
- various classes of Mobile Membranes inspired from different biological features, and study their computational and modelling power;
- provide a translation between Mobile Membranes and Mobile Ambients, two formalisms used in describing biological systems;
- extend Membranes with Objects on Surface with biological inspired co-objects, studying their computational power, and relate them to the PEP fragment of Brane Calculus.
- other aspects like time and types in mobile membranes were studied; e.g., we define mobile membranes in which each membrane and each object has a lifetime, and show that by adding explicit lifetime we do not obtain a more powerful formalism.

# Some Papers


- G. Ciobanu, M. Rotaru. A $\pi$-calculus Machine. *Journal of Universal Computer Science* vol.6, Springer, 39-59, 2000.
- G. Ciobanu, M. Rotaru. Molecular Interaction. *Theoretical Computer Science*, vol.289, 801-827, Elsevier, 2002.
- G.Ciobanu, C.Prisacariu. Timers for Distributed Systems, *Electronic Notes in Theoretical Computer Science*, vol.164, 81-99, 2006.
- G.Ciobanu, V.Zakharov. Encoding Mobile Ambients into $\pi$-calculus. *Lecture Notes in Computer Science* vol.4378, 148-161, 2006.
- G.Ciobanu, M.Koutny. Modelling and Verification of Timed Interaction and Migration, *Lecture Notes in Computer Science* vol.4961, 215-229, 2008.
- B. Aman, G. Ciobanu. Timed Mobile Ambients for Network Protocols. Lecture Notes in Computer Science vol.5048, 234-250, 2008.
- B.Aman, G.Ciobanu. On the Relationship Between Membranes and Ambients. *Biosystems* vol.91, 515-530, 2008.
- B.Aman, G.Ciobanu. Turing Completeness Using Three Mobile Membranes. *Lecture Notes in Computer Science* vol.5715, 42–55, 2009.
- B.Aman, G.Ciobanu. Simple, Enhanced and Mutual Mobile Membranes. *Transactions on Computational Systems Biology* vol.XI. 26–44, 2009.
- G.Ciobanu, S.N.Krishna. Enhanced Mobile Membranes: Computability Results, *Theory of Computing Systems*, accepted, to appear.

Thank you!