A Model for Molecular Computing: Membrane Systems

Claudio Zandron
DISCo - Universita' di Milano-Bicocca
zandron@disco.unimib.it

Summary

- Membrane Systems: introduction
- Membrane Systems: definitions
- Computational aspects of Membrane Systems
- Membrane Systems and Computational Complexity
- Application of Membrane Systems to Systems Biology

Membrane Systems: ideas

- Investigate computational properties of the cell
- Compare with other models
- Implementation (In silico? In vitro? In vivo?)
- Application to computational problems
- Application to biological problems

Membrane Systems: a Bio-inspired model

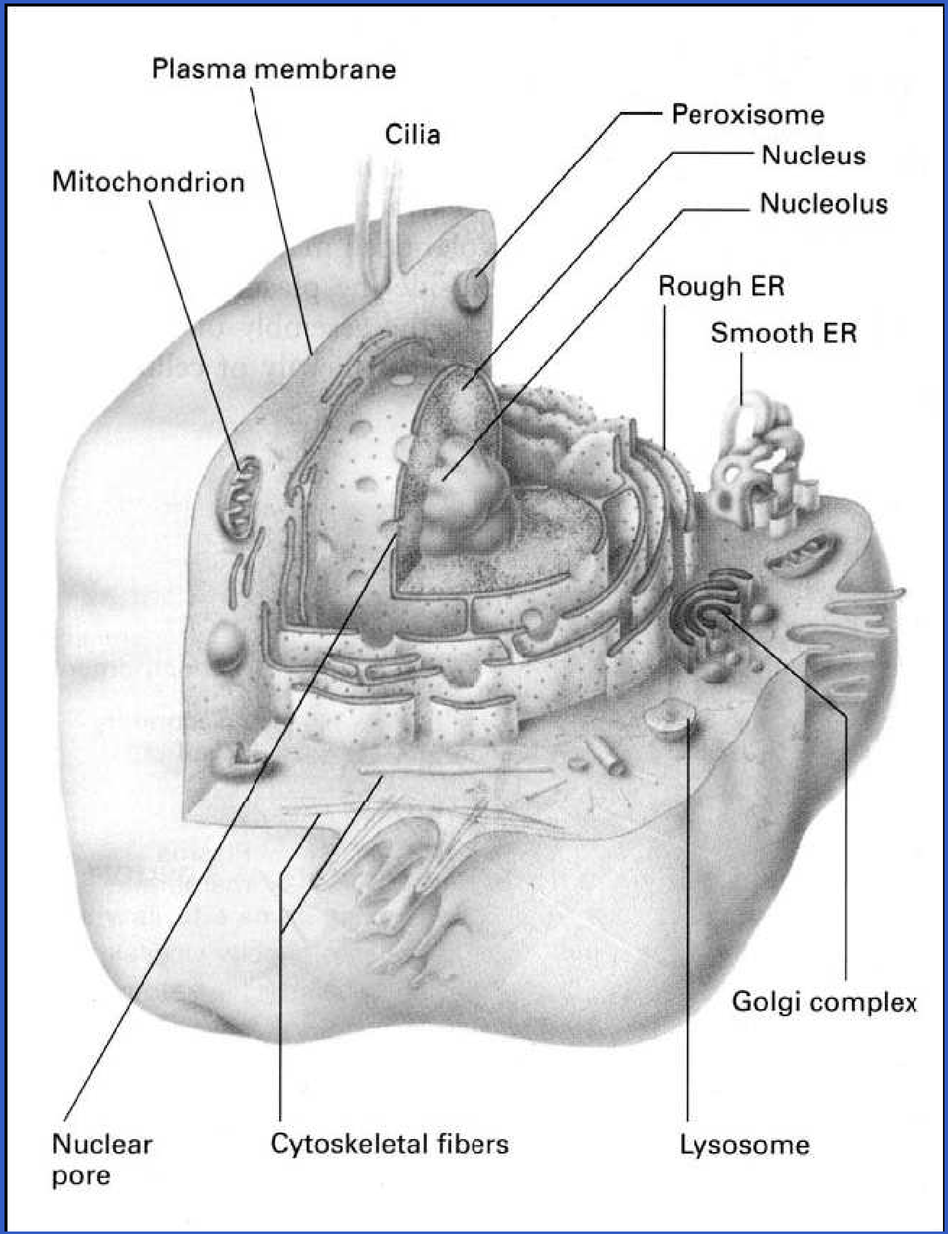
- G. Paun, 1998: computational models inspired from the structure and functioning of the cell
- Discrete model for cellular processes
- Main components:
  - Cellular structure
  - Chemical substances
  - Cellular reactions
  - Communication of substances

Membrane Systems: a Bio-inspired model

Main features:

- Discrete
- Non-Deterministic
- Maximal Parallelism

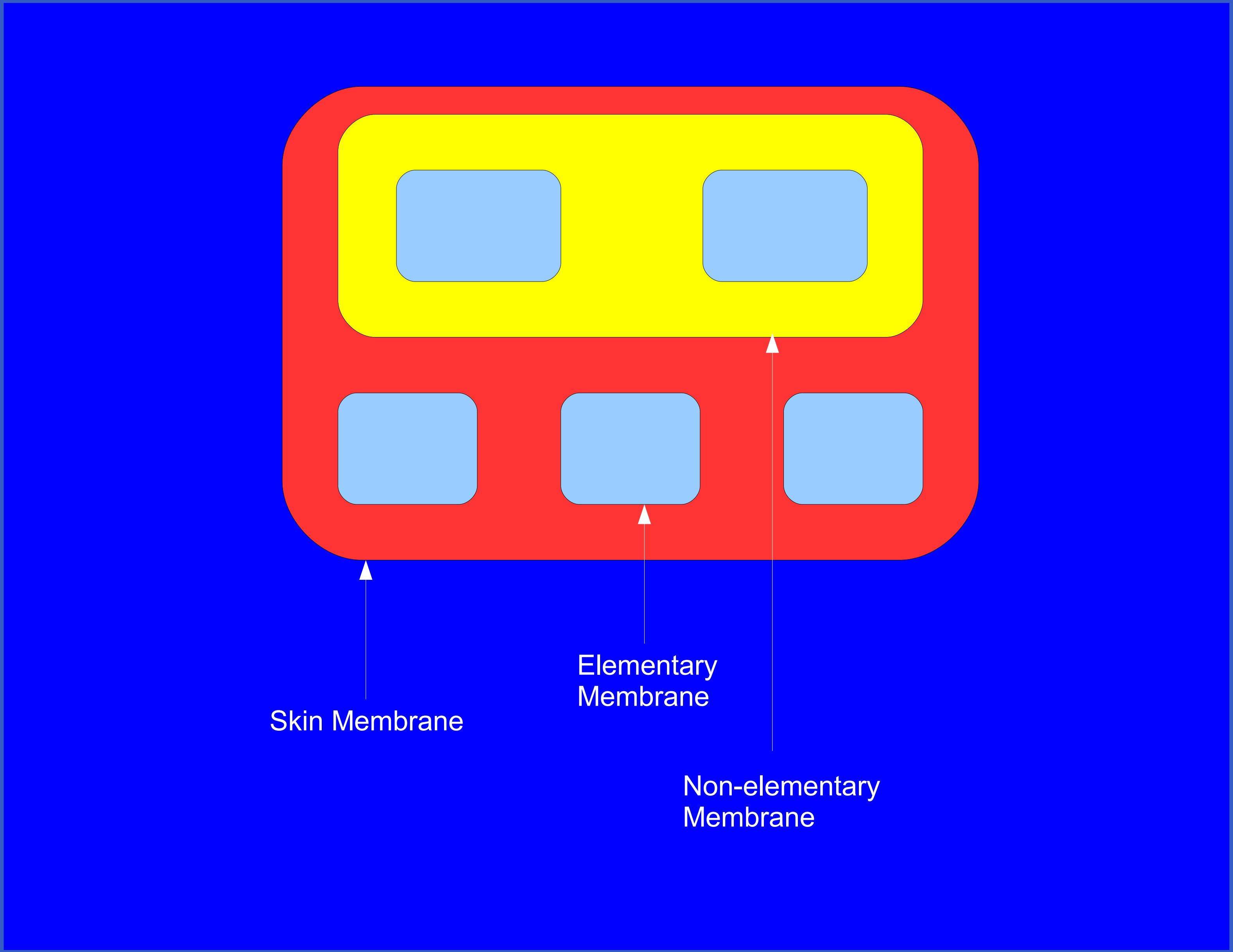
Cell Structure



Membrane structure

- Each membrane defines a region (compartment) in the membrane structure
- The most external membrane separates the system and the environment. It is called SKIN
- Some substances are communicated through the membranes
- A membrane is identified by means of a unique label
- A membrane structure can be described using matching parenthesis of a tree structure

Membrane Structure



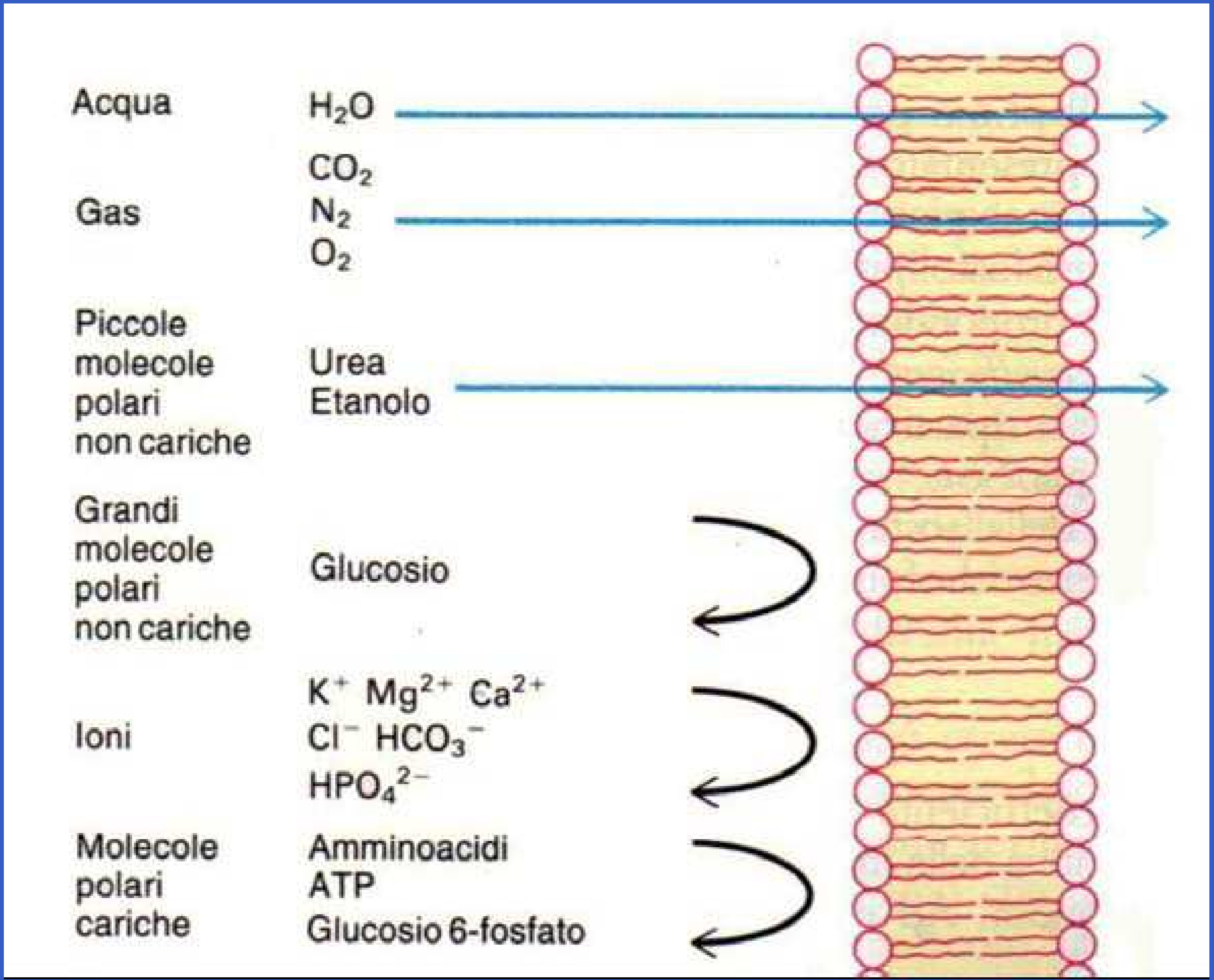
Membrane Systems: chemicals

- Chemicals - Ions, molecules, proteins: multisets of symbols (or strings) over an alphabet
- Multiset: each symbol (string) can be present in one or more copies in a region
- E.g. a^5, b^3, c means that five copies of chemical a, three copies of chemical b, and one copy of chemical c are present in a region

Membrane Systems: reactions

- Each reaction is described by means of a rewriting rule
- Chemical(s) on the left is replaced by chemicals on the right
- Examples:
  - $a \longrightarrow xy$ (Non Cooperative)
  - $ab \longrightarrow xy$ (Cooperative)
  - $ac \longrightarrow xc$ (Catalyst)

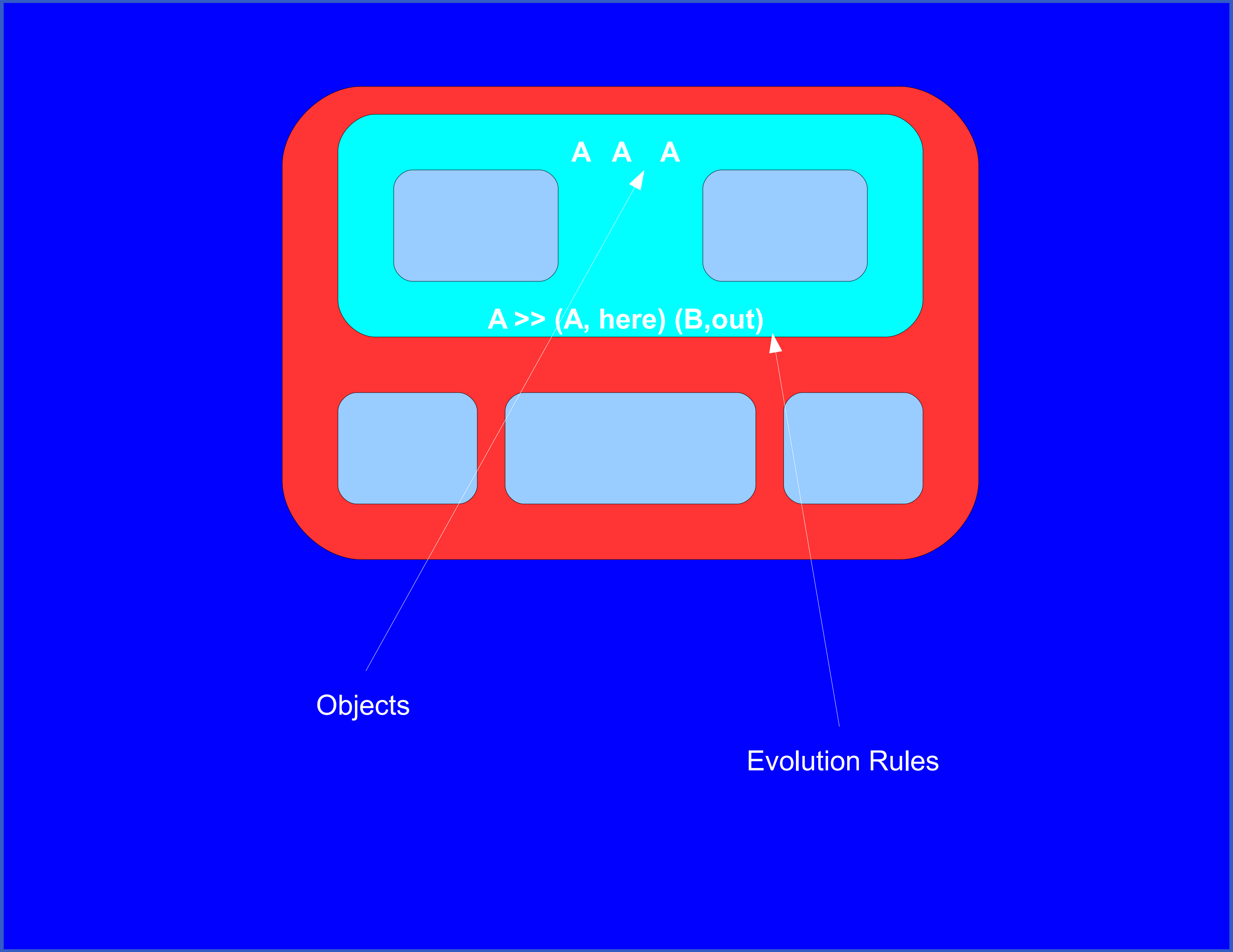
Passive Communication



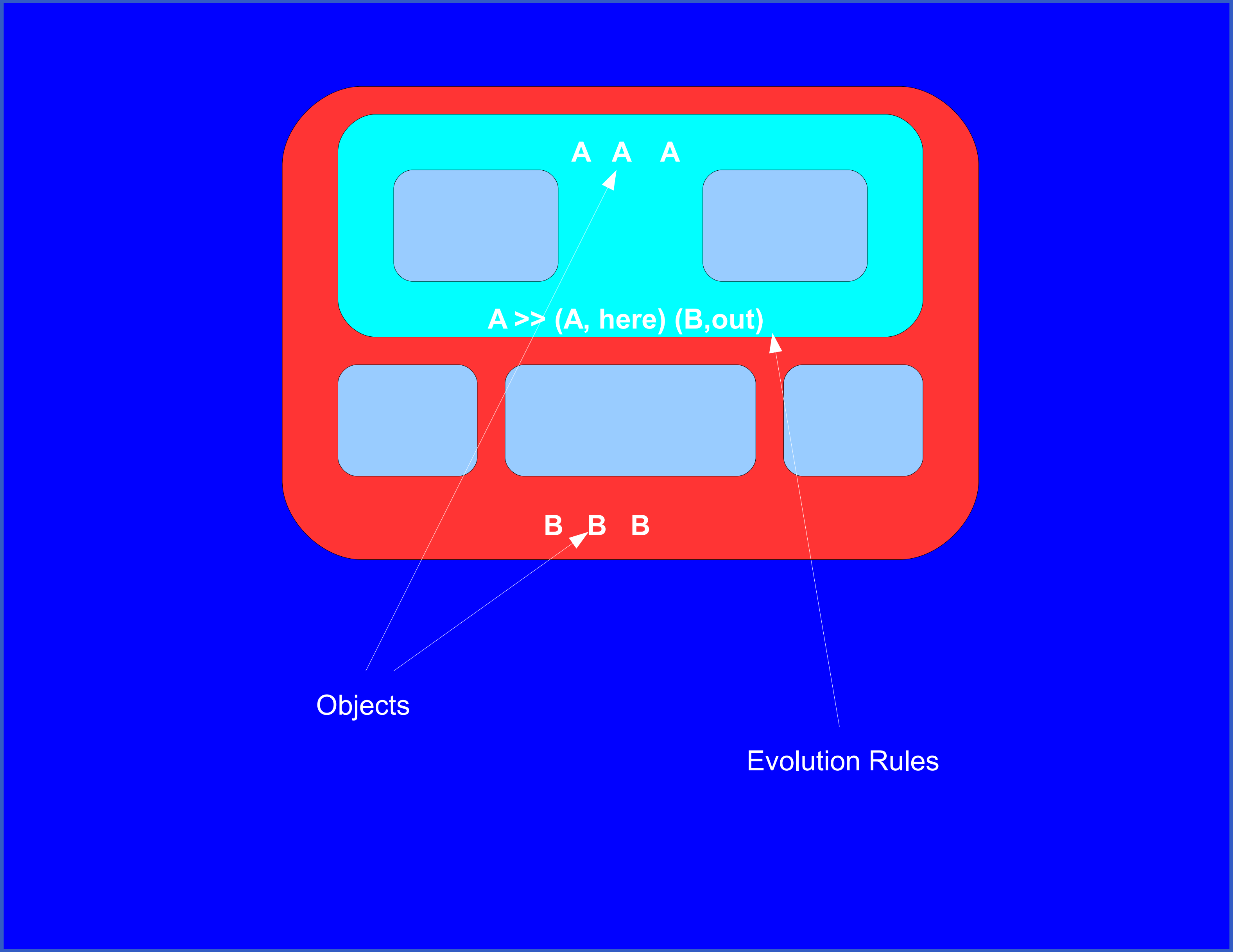
Membrane Systems: reactions and communication

- Each reaction is described by means of a rewriting rule and target indication
- Target: $here$, out, in_j
- Chemical(s) on the left is replaced by chemicals on the right
- Obtained chemicals are communicated according to target indication
- Examples:
  - $a \longrightarrow (x, here)(y, out)(z, in_3)$
  - $ab \longrightarrow x(y, in_5)^2$
  - $ac \longrightarrow (x, out)^3 c$

Membrane System



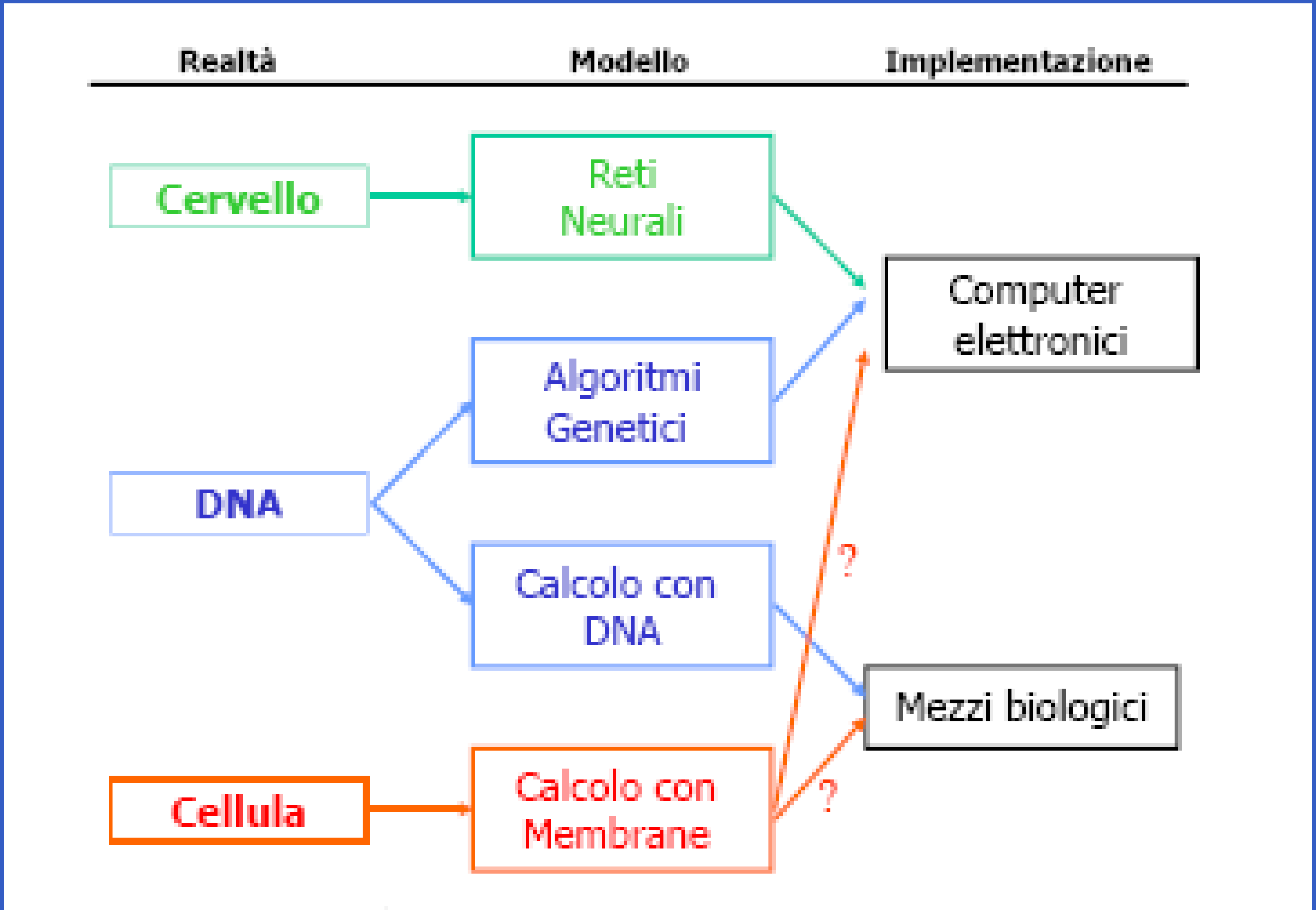
Membrane System



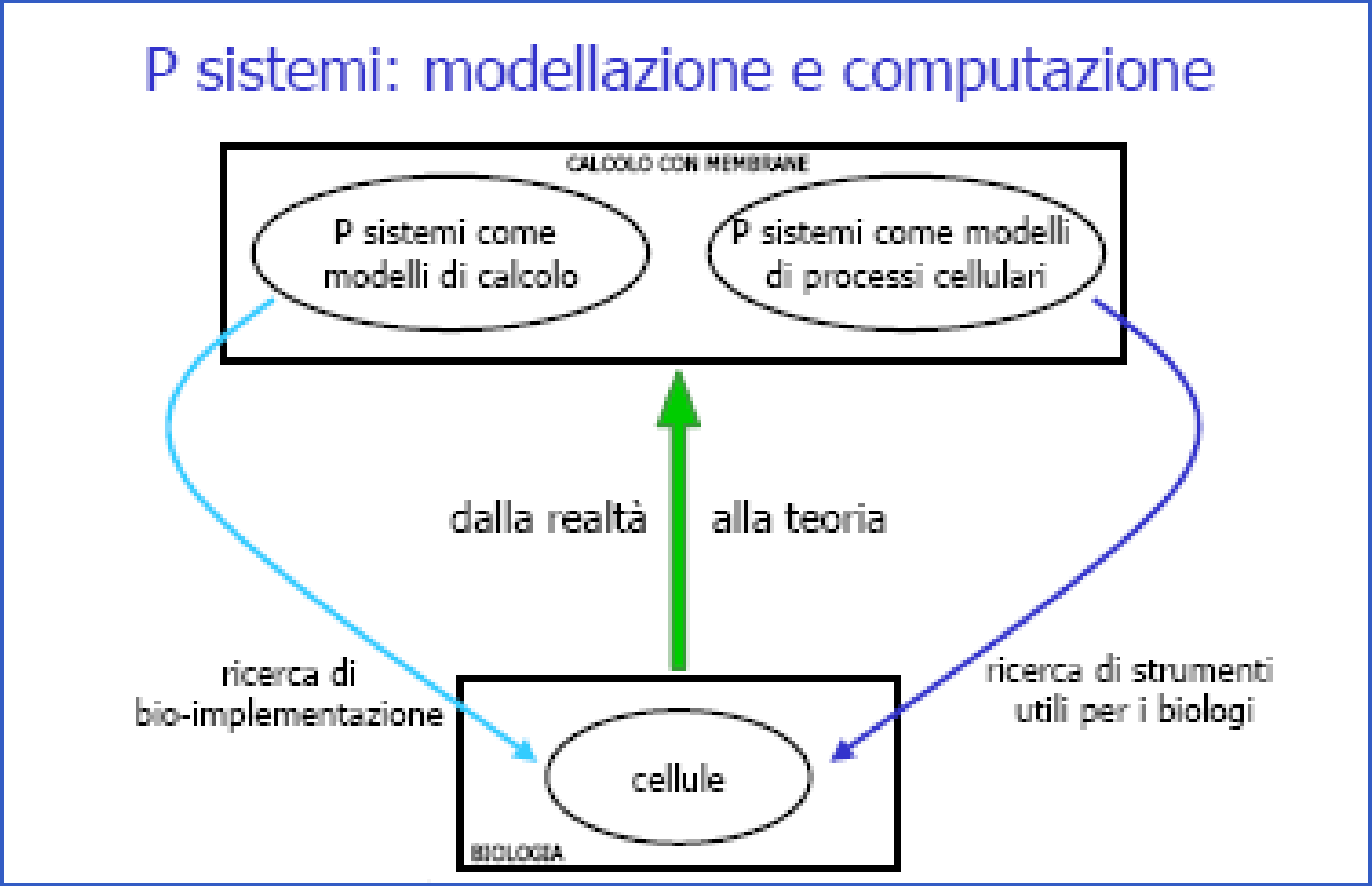
Further (basic) rules features

- δ: dissolving membrane action
  - Membrane disappears
  - Evolution rules disappear
  - Skin Membrane cannot be dissolved
- Priority relation among rewriting rules
  - Rules are applied following partial priority order
  - A rule can be applied only if no rule of higher priority can be applied
  - If different choices of rules can be applied at the same time, then non-deterministic choice

Implementation?



Use of membrane systems



Definition

$$\Pi = (V, \mu, M_1, \ldots, M_n, (R_1, \rho_1), \ldots, (R_n, \rho_n), i_0)$$

- V: Alphabet
- μ: Membrane structure (Ex. $[\,[\,]_2\,[\,]_3\,[\,[\,]_5\,[\,]_6\,]_4\,]_1$)
- M_i: Multisets of symbols (or strings) in V
- R_i: Finite sets of evolution rules $x \rightarrow y(tar)$, $x \in V^*, y \in V^* \cup V^*\delta(\delta \notin V), tar \in \{here, out, in_j\}$
- ρ_i: Partial order relations over R_i
- i_0: Output Membrane. If empty, then the output region is the environment

Evolution

- $M_1, \ldots, M_n$: initial configuration
- Rules are applied following the given priorities
- Rules are applied in a non-deterministic way
- All objects evolve in parallel
- All regions evolve in parallel
- Rules can move objects through membranes
  - *here*: the object is not moved
  - *out*: the object is sent to the adjacent external region
  - in_j: the object is sent to the inner membrane with label j
- δ: membrane is dissolved

Computation

- **Computation**: Sequence of transitions. A computation halts when no further rule can be applied.
- **Output**:
  - Objects in i_0 when the computation halts
  - $\emptyset$ if the computation never stops
- $NOP_m(\alpha, \beta, ...)$: family of numbers generated by object P systems with at most m membranes
- $RP_m(\alpha, \beta, ...)$: family of languages generated by Rewriting P systems with at most m membranes

- $\alpha \in \{Pri, nPri\}$ (with or without priority)
- $\beta \in \{\delta, n\delta\}$ (with or without dissolving membrane action)

Example: $6n$

$$\Pi_1 = (V, \mu, M_1, M_2, (R_1, \rho_1), (R_2, \rho_2), 2)$$

$$\begin{aligned}
V &= \{a, b, c\} \\
\mu &= [_1 [_2]_2]_1 \\
M_1 &= a^2 \\
M_2 &= \lambda \\
R_1 &= \{a \longrightarrow a(b, in_2)(c, in_2)^2, a^2 \longrightarrow (a, out)\} \\
R_2 &= \emptyset \\
\rho_1, \rho_2 &= \emptyset
\end{aligned}$$

Example: $6n$

- Step 0: $[aa[\,]_2]_1$

Example: $6n$

- Step 0: $[aa[]_2]_1$
- Step 1: $[aa[bccbcc]_2]_1$

Example: $6n$

- Step 0: $[aa[]_2]_1$
- Step 1: $[aa[bccbcc]_2]_1$
- Step 2: $[aa[bccbccbccbcc]_2]_1$

Example: $6n$

- Step 0: $[aa[\,]_2]_1$
- Step 1: $[aa[bccbcc]_2]_1$
- Step 2: $[aa[bccbccbccbcc]_2]_1$
- Step 3: $[aa[b^6c^{12}]_2]_1$

Example: $6n$

- Step 0: $[aa[]_2]_1$
- Step 1: $[aa[bccbcc]_2]_1$
- Step 2: $[aa[bccbccbccbcc]_2]_1$
- Step 3: $[aa[b^6c^{12}]_2]_1$
- Step 4: $[aa[b^8c^{16}]_2]_1$

Example: $6n$

- Step 0: $[aa[]_2]_1$
- Step 1: $[aa[bccbcc]_2]_1$
- Step 2: $[aa[bccbccbccbcc]_2]_1$
- Step 3: $[aa[b^6c^{12}]_2]_1$
- Step 4: $[aa[b^8c^{16}]_2]_1$
- ...
- Step n: $aa[[b^{2n}c^{4n}]_2]_1$

Symbol-object systems: main results

$NOP_k(feat)$= family of natural numbers generated by P systems using k membranes and the features specified in $feat$ (cooperative/non-cooperative rules, dissolving membrane action, etc.)

- $NOP_*(coo, tar) = NOP_1(coo, tar) = NRE$ - Cooperative rules, universality
- $NOP_*(ncoo, tar) = NOP_1(ncoo, tar) = NCF$ - Non-Cooperative rules, context free
- $NCF = NOP_*(ncoo, tar) \subset (NE0L \subseteq)NOP_2(ncoo, tar, \delta)$ - Dissolving rules increase power
- $NOP_*(ncoo, tar, \delta)(\subseteq ET0L) \subset NCS$ - Dissolving rules not universal

Symbol-object systems: main results

- $NOP_*(ncoo, tar, \delta) \subseteq NOP_*(cat, tar, \delta) \subseteq NOP_*(coo, tar, \delta)$ - Catalysts are in between cooperative and non-cooperative rules
- $NOP_*(2cat, tar) = NRE$ - Bistable catalysts, universality
- $NOP_*(ncoo, tar, \delta)(\subseteq ET0L) \subseteq NOP_*(ncoo, tar, pri)$ - Priority increase power (?)
- $NOP_2(cat, tar, pri) = NRE$ - Priority increase power (?)

Structuring the objects: string-objects

- Complex molecules can be represented by strings
- CF rewriting rule are applied in parallel to all strings, but ONE RULE PER STRING
- After replacing a simbol in the string with a set of simbols, the whole string is sent to the target destination
- Further features (sissolving membrane action, priorities, etc.) can still be considered

String-object systems: main results

$RP_k(feat)$= family of languages generated by P systems using k membranes and the features specified in $feat$ (cooperative/non-cooperative rules, dissolving membrane action, etc.)

- $RP_1(CS) = CS, RP_1(RE) = RE$
- $RP_1(CF, nPri, n\delta) = CF$
- $CF \subset RP_4(nPri, n\delta) = RP_*(nPri, n\delta)(= MAT) \subset RE$ - Membrane structure increases power
- $RP_3(Pri, n\delta) = RE$ - Priority increases power

Some basic variants: promoters and inhibitors

Substances participate in reaction to allow or forbid certain reactions

- Inhibitor conditions: a rule can be applied iff a multiset/string does not contain certain symbols:
  $NOP_*(Inh, feat) = ?$
  $RP_7(Inh, n\delta) = RE$
- Promoter conditions: a rule can be applied iff a string does contain certain symbols
  $NOP_6(ncoo, pro) = NRE$
  $RP_*(Perm, n\delta)(= MAT) \subset RE$

Variable Thickness of Membranes

Rules are of the form:

$$A \to xT(tar)$$

where $A \in V, x \in V^*, tar \in \{here, out, in_j\}, T \in \{\lambda, \delta, \tau\}$

symbol	**thickness**	**Action**
δ	1	Membrane is dissolved
δ	2	Thickness is reduced
τ	1	Thickness is increased
τ	2	No changes

Variable Thickness of Membranes

- If a membrane has thickness 2, then no object can pass through it
- When a membrane is dissolved
  - all the rules of the membrane are lost
  - all objects remain free in the membrane immediately outside

The movement of objects can be controlled by means of such feature

Result: $RE = RP_5(nPri, \delta, \tau)$

Communication using Electrical Charges

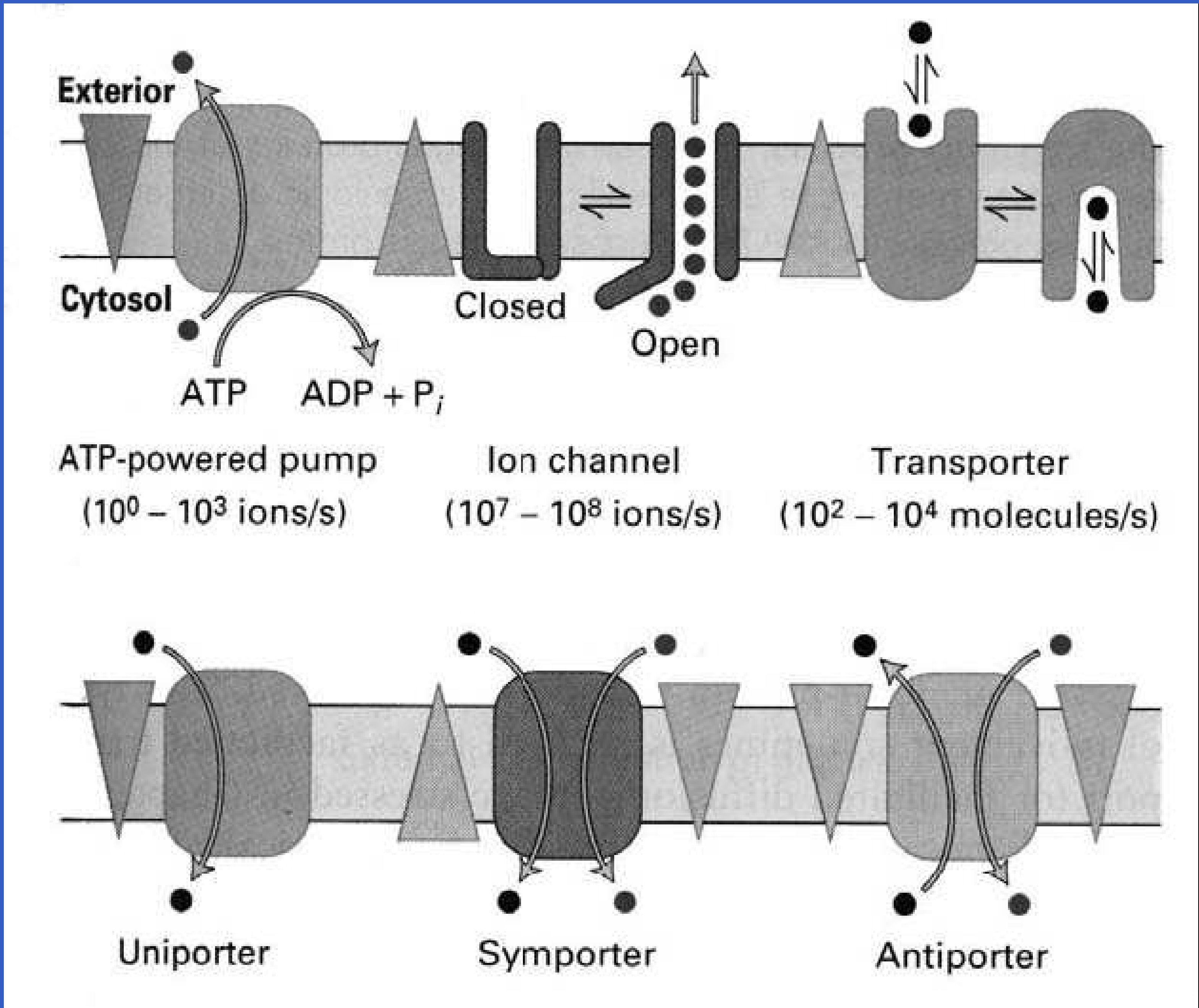
- Electrical charges are associated with both the objects and the membranes
- A rule marks the object with $t \in \{out, 0, +, -\}$

Out: The object is sent to the external region

0: The object remains in the same region

$+$ ($-$): The object is sent to an inner membrane marked with $-$ ($+$), if any

Symport/Antiport communication



Membrane Systems with Symport/Antiport

Complex communication rules:

- Uniport: (a,in)
- Symport: (ab,in)
- Antiport: (a,in)(b,out)

Universality for systems using only communication (no evolution). Objects imported from the environment.

(Some) Research Directions

- Computational complexity aspects for Membrane systems
- Biological processes description (p53 signalling pathways, photosynthesis, etc.)
- Stochastic modelling and simulation algorithms for complex systems
- Approximation algorithms for optimization problems
- Natural languages and parsers
- Comparisons with similar models (e.g. Cardelli's Brane Calculi)

Creation of membranes by membrane division.
Two types of division:

- Division for elementary membranes

$$[_h\ A\ ]_h^{\alpha} \rightarrow [_h\ B\ ]_h^{\beta}\ [_h\ C\ ]_h^{\gamma}$$

- Division for non-elementary membranes

$$[_{h_0}\ [_{h_1}\ ]_{h_1}^{+} \ldots [_{h_k}\ ]_{h_k}^{+}\ [_{h_{k+1}}\ ]_{h_{k+1}}^{-} \ldots [_{h_n}\ ]_{h_n}^{-}\ ]_{h_0}^{\alpha} \longrightarrow$$
$$[_{h_0}\ [_{h_1}\ ]_{h_1}^{+} \ldots [_{h_k}\ ]_{h_k}^{+}\ ]_{h_0}^{\beta}\ [_{h_0}\ [_{h_{k+1}}\ ]_{h_{k+1}}^{-} \ldots [_{h_n}\ ]_{h_n}^{-}\ ]_{h_0}^{\gamma}$$

Attacking computationally complex problems

- Satisfiability can be solved in polynomial time (exponential space) using membrane systems with active membranes
- Hamiltonian Path Problem can be solved in polynomial time (exponential space) using membrane systems with active membranes

Result: SAT and HPP can be solved in polynomial time (exponential space) using membrane systems with active membranes
<u>without division for non-elementary membranes</u>

Systems without Membrane Division

Question: Is it possible to solve NP Complete problems in polynomial time using DETERMINISTIC or CONFLUENT P systems without membrane division?

Result: Every deterministic (or confluent) membrane system Π, without membrane division and working in time t, can be simulated by a DTM working in time $O(t \times A \times B \times C \times log(A \times B \times C^t))$, where

- $A = \#$ of membranes in Π
- $B = \#$ of symbols in V
- $C = max\{k \mid k$ is the size of a rule in $\Pi\}$

Complexity Classes in P-Systems

- $NP \subseteq PMC_{eam}$
- $coNP \subseteq PMC_{eam}$
- $PSPACE \subseteq PMC_{am}$
- $PMC_{am} \subseteq EXPSPACE$
- $PMC_{am} \subseteq EXPTIME$

Main resources

- BOOKS:
  - G. Paun, Membrane Computing - An introduction, Springer-Verlag, Berlin, 2002
  - G. Ciobanu, M.J. Perez-Jimenez, G. Paun (Eds), Applications of Membrane Computing, Springer-Verlag, Berlin 2006
  - P. Frisco, Computing with Cells. Advances in Membrane Computing, Oxford University Press, 2009
  - G. Paun (ed.), Handbook of Membrane Computing, in press
- INTERNET: P systems web page: http://ppage.psystems.eu