Numerical P Systems (After Ten Years)

Gheorghe Păun
Romanian Academy, București,
gpaun@us.es, curteadelaarges@gmail.com

http://ppage.psystems.eu

Summary:

- quick introduction to membrane computing
  1. from cells to models
  2. references
  3. (types of) results
  4. (types of) applications
- Numerical P systems:
  - the sequential case
    1. the idea; two examples
    2. formal definition
    3. results
  - the parallel case
    1. considering "enzymes"
    2. two types of parallelism
    3. computing power results
    4. computational complexity results
  - applications

MEMBRANE COMPUTING

= branch of natural computing, aiming to abstract computing models from the living cell structure and functioning, and from cells cooperation in tissues, organs, brain, etc.

= modeling framework, general and versatile (not a single model)

= basically, a framework for handling multisets in a compartmentalized spatial structure

= possibilities to trade-off (exponential) space (created in a biological-like way) with time

Framework: Natural Computing (Unconventional Computing)

Preface of *Handbook of Natural Computing*, 4 vols., Springer, Berlin, 2012, edited by G. Rozenberg, Th. Bäck, J.N. Kok:

Natural Computing is the field of research that investigates human-designed computing inspired by nature as well as computing taking place in nature, that is, it investigates models and computational techniques inspired by nature, and also it investigates, in terms of information processing, phenomena taking place in nature.

A. Adamatzky, ed., *Advances in Unconventional Computing*, 2 vols., Springer, Berlin, 2016

H. Zenil, ed., *A Computable Universe. Understanding and Exploring Nature as Computation*, World Scientific, 2013

J. Gruska, etc.

Why natural computing?

1. limits of Turing – von Neumann paradigm
2. limits of current technology (Moore law, communication complexity)
3. adaptation, (deep!)learning, self-healing, robustness, nondeterminism, etc.
4. energy efficiency
5. challenge for matematics/informatics
   (what means to compute $in\ a\ natural\ way?$)
6. by-products (biology, medicine)
7. a new framework for understanding biology/nature

Everything started about 20 years ago, in Turku, Finland...

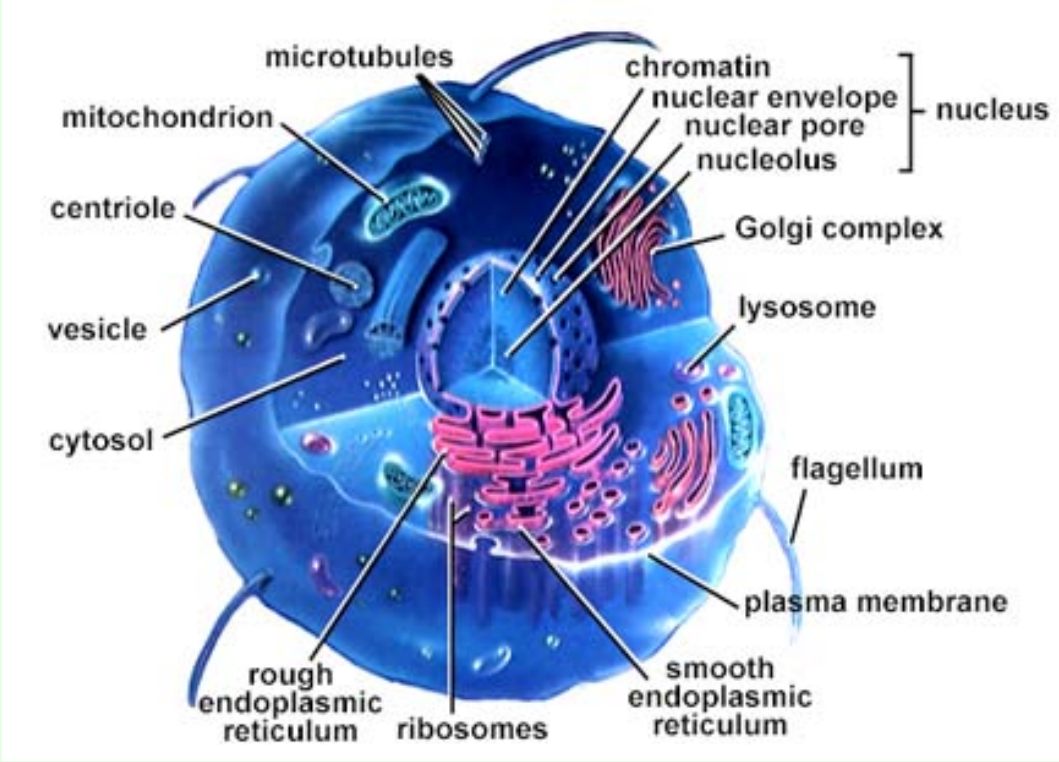
...after DNA computing

Let's go to the cell!

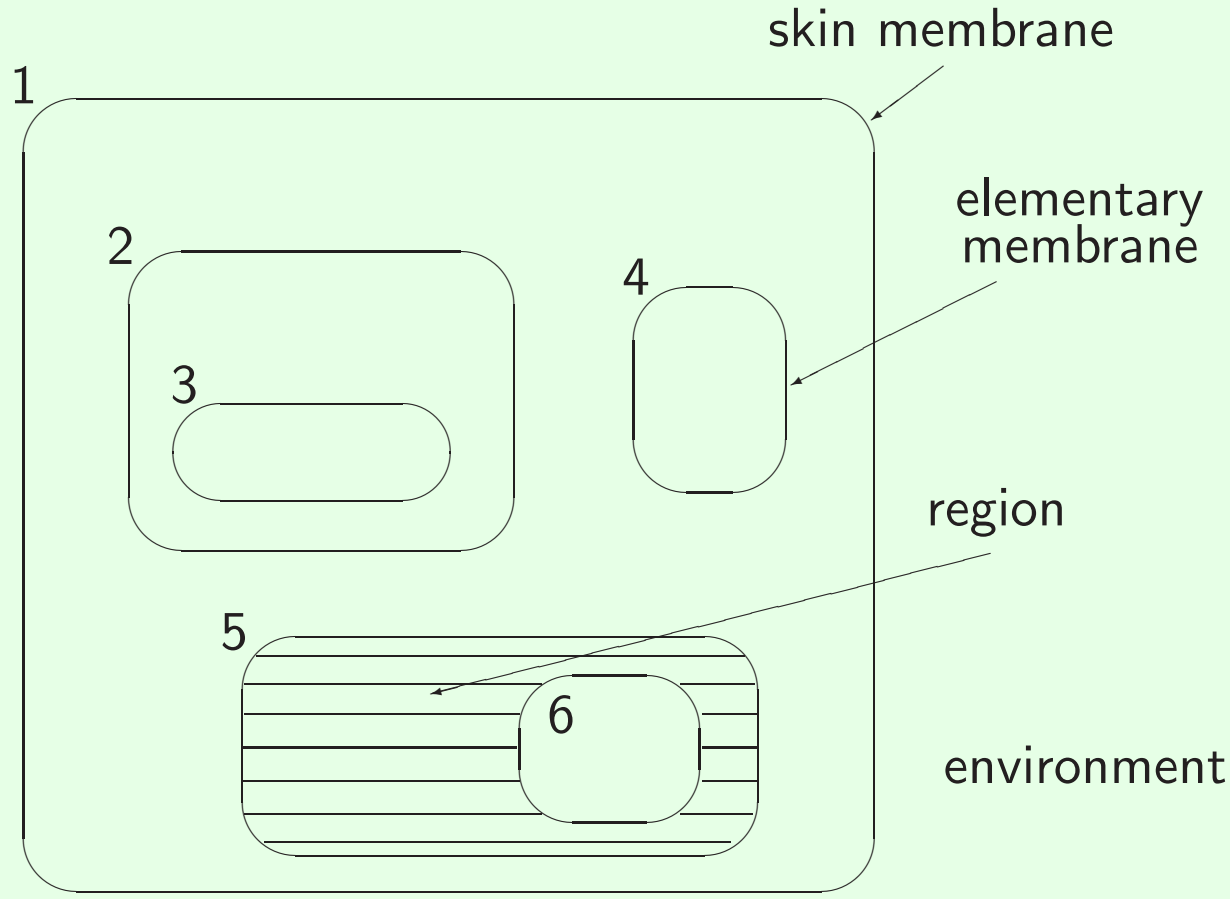
Membrane computing — Starting from the cell

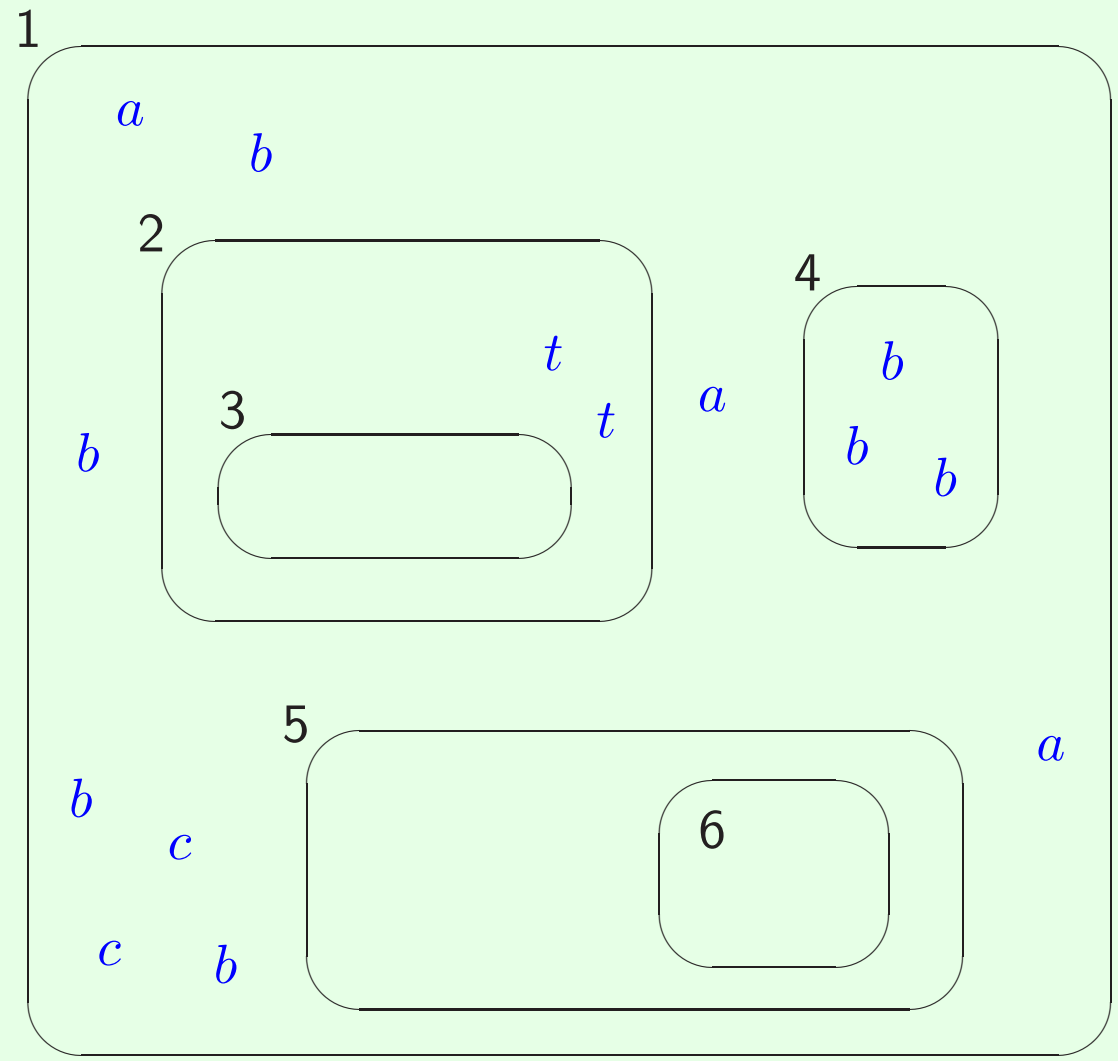
[Initial] Goal: abstracting computing models/ideas from the structure and functioning of living cells (and from their organization in tissues, organs, organisms)

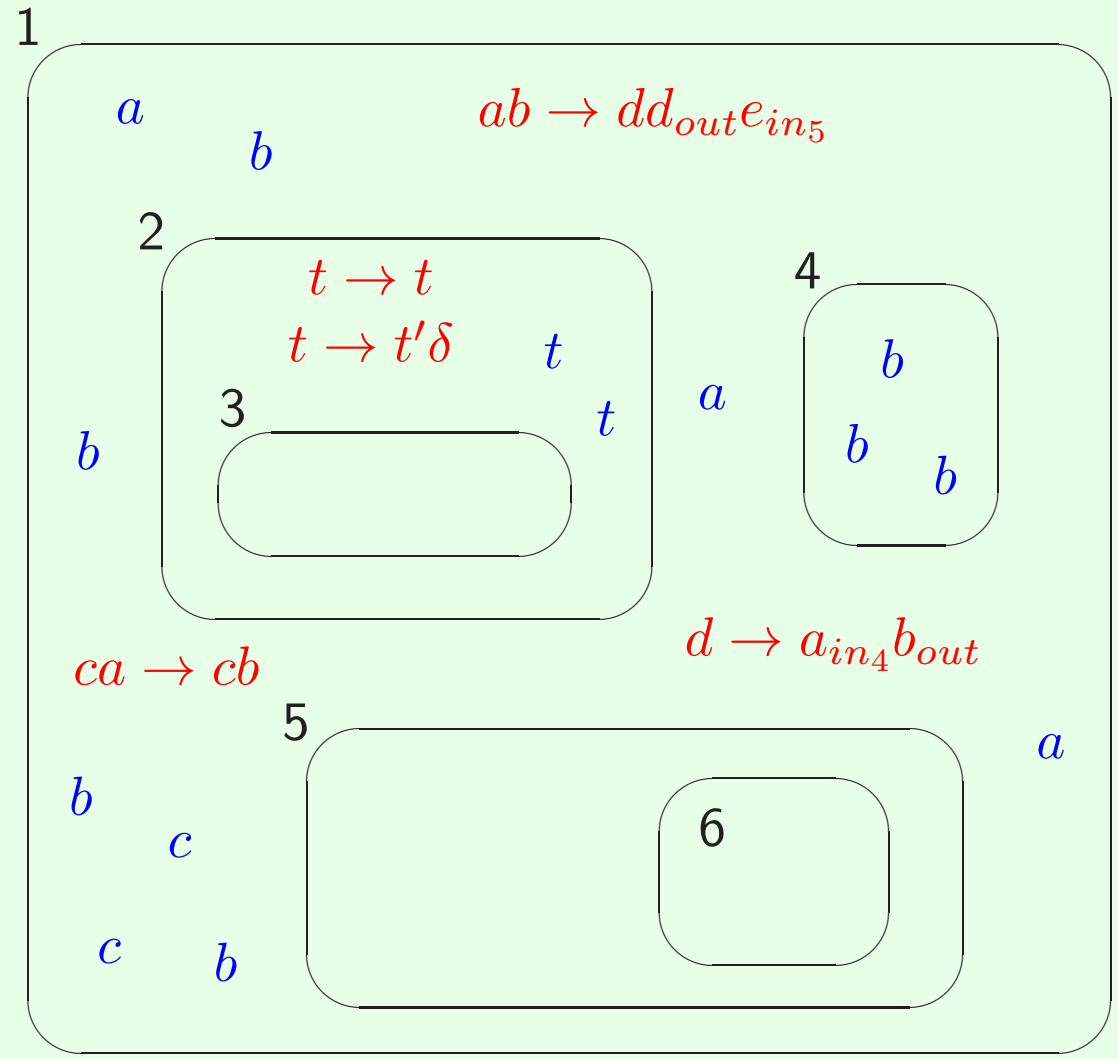
hence not producing models for biologists (although, this is now a tendency)



THE BASIC IDEA




1
a
b
2
t
t
3
a
b
4
b
b
b
5
a
b
c
6
c
b


1
a
b
$ab \to dd_{out}e_{in_5}$
2
$t \to t$
$t \to t'\delta$
t
t
3
b
a
4
b
b
b
$d \to a_{in_4}b_{out}$
$ca \to cb$
5
a
b
c
6
c
b

Functioning (basic ingredients):

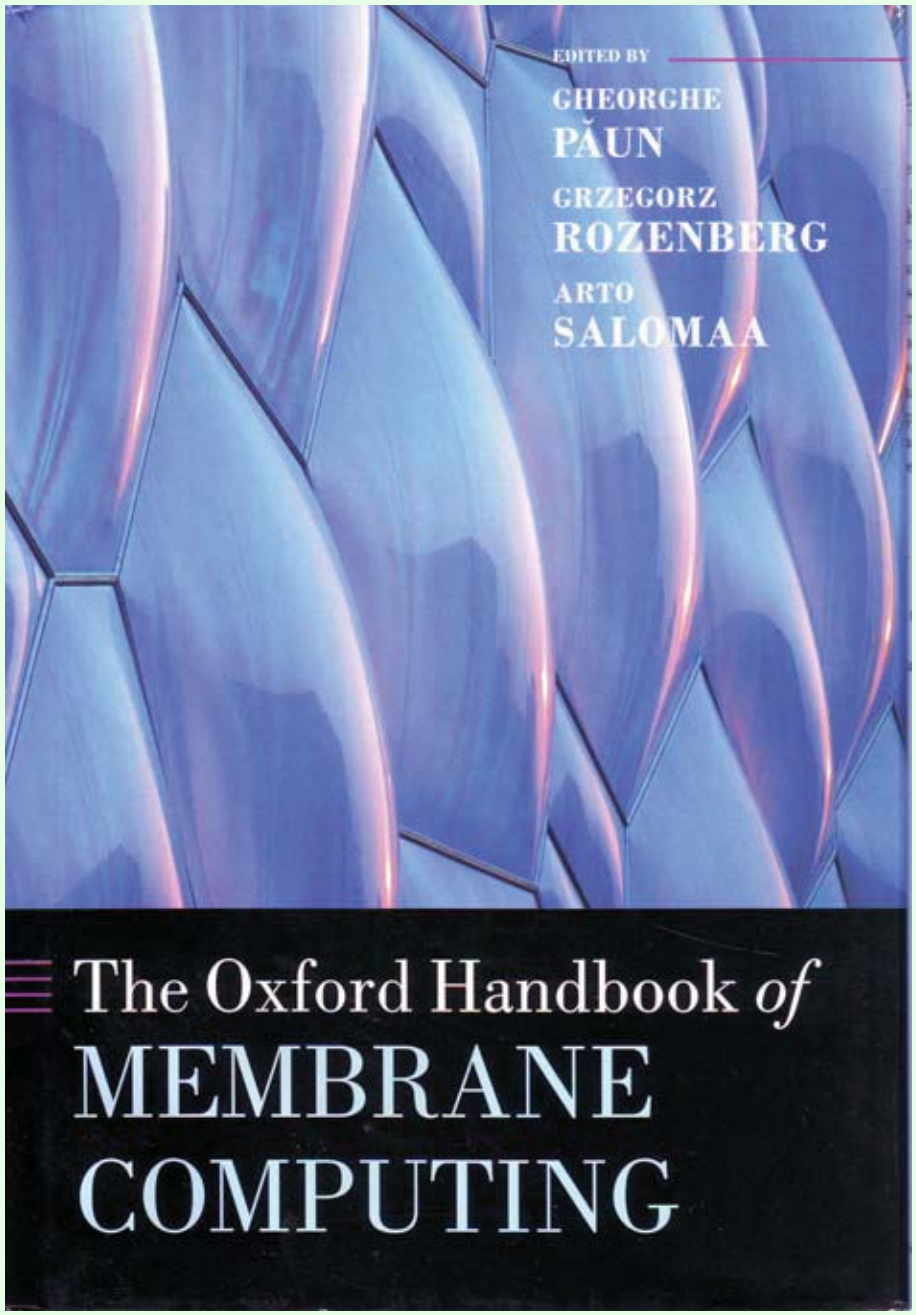
- nondeterministic choice of rules and objects
- maximal parallelism
- transition, computation, halting
- various ways to define the output (generative, accepting, computing)

Some references (branches, classes of P systems):

- Gh. Păun: Computing with Membranes. *Journal of Computer and System Sciences*, 61, 1 (2000), 108–143, and *Turku Center for Computer Science-TUCS Report* No 208, 1998 (www.tucs.fi)
- Gh. Păun: P systems with active membranes: Attacking NP-complete problems, *J. Automata, Languages, and Combinatorics*, 6, 1 (2001), 75–90
- A. Păun, Gh. Păun: The power of communication: P systems with symport/antiport, *New Generation Computing*, 20, 3 (2002), 295–306
- C. Martin-Vide, J. Pazos, Gh. Păun, A. Rodriguez-Paton: Tissue P systems, *Theoretical Computer Sci.*, 296, 2 (2003), 295–326
- Gh. Păun, R. Păun: Membrane computing and economics: Numerical P systems, *Fundamenta Informaticae*, 73, 1-2 (2006), 213–227
- M. Ionescu, Gh. Păun, T. Yokomori: Spiking neural P systems, *Fundamenta Informaticae*, 71, 2-3 (2006), 279–308
- Gh. Păun, M.J. Pérez-Jiménez: Solving problems in a distributed way in membrane computing: dP systems, *Int. J. of Computers, Communication and Control*, 5, 2 (2010), 238–252.
  (other branches: conformon, population, kernel, metabolic, etc.)

Monographs/volumes in membrane computing

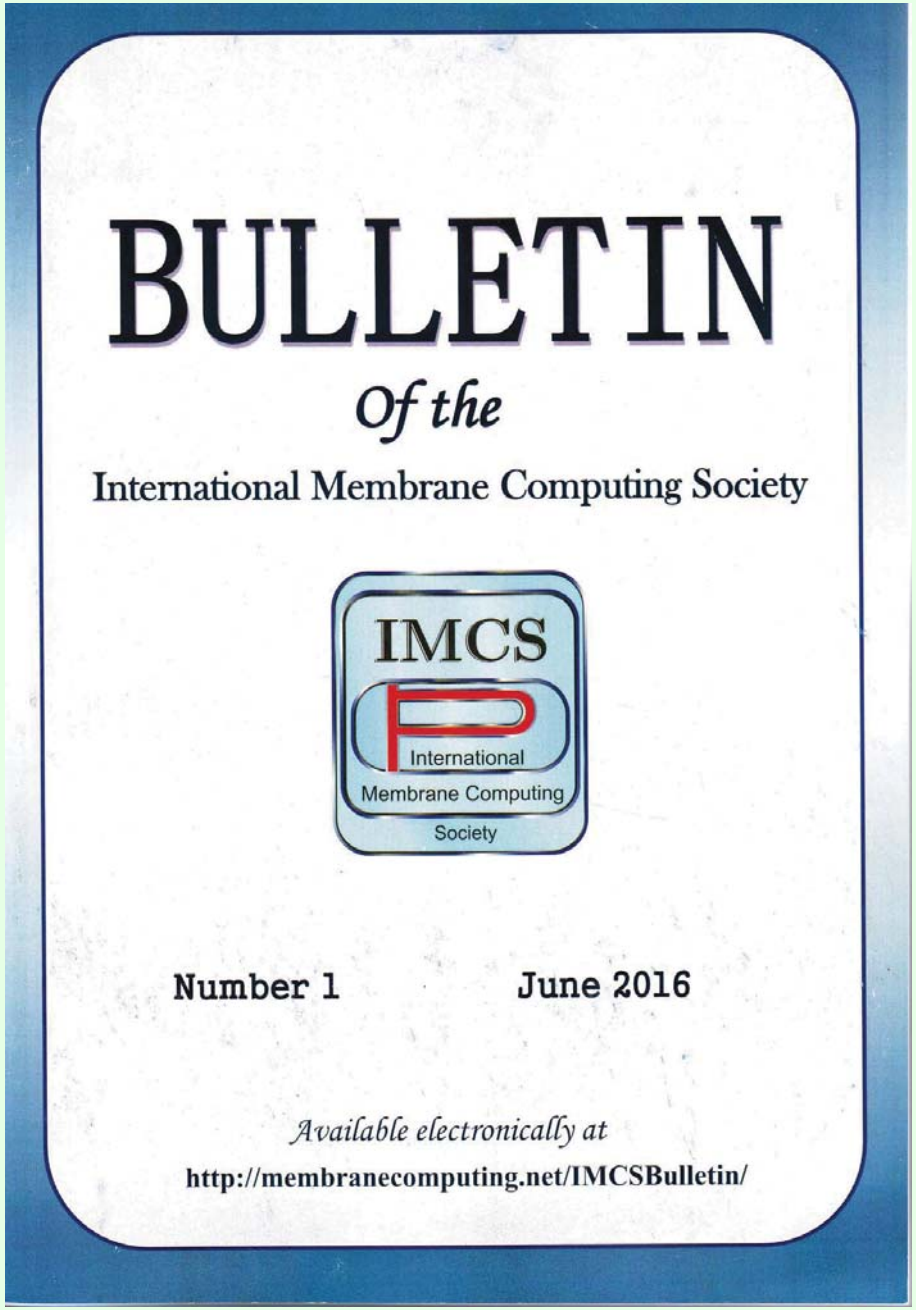
1. Gh. Păun: *Membrane Computing. An Introduction*. Springer, 2002 (translated in Chinese in 2012)
2. G. Ciobanu, Gh. Păun, M.J. Pérez-Jiménez, eds.: *Applications of Membrane Computing*. Springer, 2006
3. P. Frisco: *Computing with Cells. Advances in Membrane Computing*. Oxford Univ. Press, 2009
4. G. Ciobanu: *Membrane Computing. Biologically Inspired Process Calculi*. The Publ. House of Al.I. Cuza Univ., Iași, 2010
5. Gh. Păun, G. Rozenberg, A. Salomaa, eds.: *The Oxford Handbook of Membrane Computing*. Oxford Univ. Press, 2010
6. V. Manca: *Infobiotics. Information in Biotic Systems*. Springer, 2013
7. P. Frisco, M. Gheorghe, M.J. Pérez-Jiménez, eds.: *Applications of Membrane Computing in Systems and Synthetic Biology*. Springer, 2014
8. Gexiang Zhang, Jixiang Cheng, Tao Wang, Xueyuan Wang, Jie Zhu: *Membrane Computing: Theory and Applications*. Science Press, Beijing, 2015
9. NEW Gexiang Zhang, M.J. Pérez-Jiménez, M. Gheorghe: *Real-Life Applications with Membrane Computing*. Springer, 2017


EDITED BY
GHEORGHE
PĂUN
GRZEGORZ
ROZENBERG
ARTO
SALOMAA
The Oxford Handbook of
MEMBRANE
COMPUTING

News in Membrane Computing:

creation (2016) of IMCS = International MC Society (web page under construction)

1. three yearly meetings: CMC, ACMC, BWMC
2. *Bulletin of the IMCS*: `http://membranecomputing.net/IMCSBulletin/`
3. three yearly prizes (PhD, theory, application of the year)
4. *International Journal of MC*
5. a series of books
6. others ("connecting people")


BULLETIN
Of the
International Membrane Computing Society
IMCS
International
Membrane Computing
Society
Number 1
June 2016
Available electronically at
http://membranecomputing.net/IMCSBulletin/

Formal definitions:

Catalytic P systems: $\Pi = (O, C, \mu, w_1, \ldots, w_m, R_1, \ldots, R_m, i_{in}, i_{out})$

rules: $u \to v, \quad a \to v, \quad ca \to cv$

(v can have target indications, $(a, here), (a, in), (a, out)$)

Symport/antiport P systems: $\Pi = (O, \mu, w_1, \ldots, w_m, E, R_1, \ldots, R_m, i_{in}, i_{out})$

rules: $(u, out; v, in)$ (antiport), $(u, out), (u, in)$ (symport)

Spiking neural P systems: $\Pi = (O, \sigma_1, \ldots, \sigma_m, syn, i_{in}, i_{out})$

neurons: $\sigma_i = (n_i, R_i)$

rules: $E/a^c \to a^p; d$ (spiking), $a^s \to \lambda$ (forgetting)

(Types of) Results:

- characterization of Turing computability (RE, NRE, $PsRE$)
  Examples: by catalytic P systems (2 catalysts) [Sosik, Freund, Kari, Oswald]
  by (small) symport/antiport P systems [many]
  by spiking neural P systems [many]
- polynomial solutions to NP-complete problems (by using an exponential workspace created in a "biological way": membrane division, membrane creation, string replication, etc.) [Sevilla team], [Madras team], [Milano team], [Alhazov, Pan, Sosik, Murphy, Woods] etc.
  even characterizations of **PSPACE**
- other types of mathematical results (normal forms, hierarchies, determinism versus nondeterminism, complexity) [Ibarra group]
- connections with ambient calculus, Petri nets, X-machines, quantum computing, lambda calculus, brane calculus, etc. [many]
- simulations and implementations
- applications

Some FAQ:

- computing beyond Turing? (no, but ...acceleration)
- what kind of implementation? (none, but ...Sevilla, Adelaide, Madrid, Technion-Haifa)
- why so many variants?
- why so powerful? (RE = CS + erasing)

(Types of) Applications:

- biology, medicine, ecosystems (continuous versus discrete mathematics) [Sevilla, Verona, Milano, Sheffield, Ruston-Lousiana, Trento, etc.]

- computer science (computer graphics, sorting/ranking, 2D languages, cryptography, general model of distributed-parallel computing) [many]

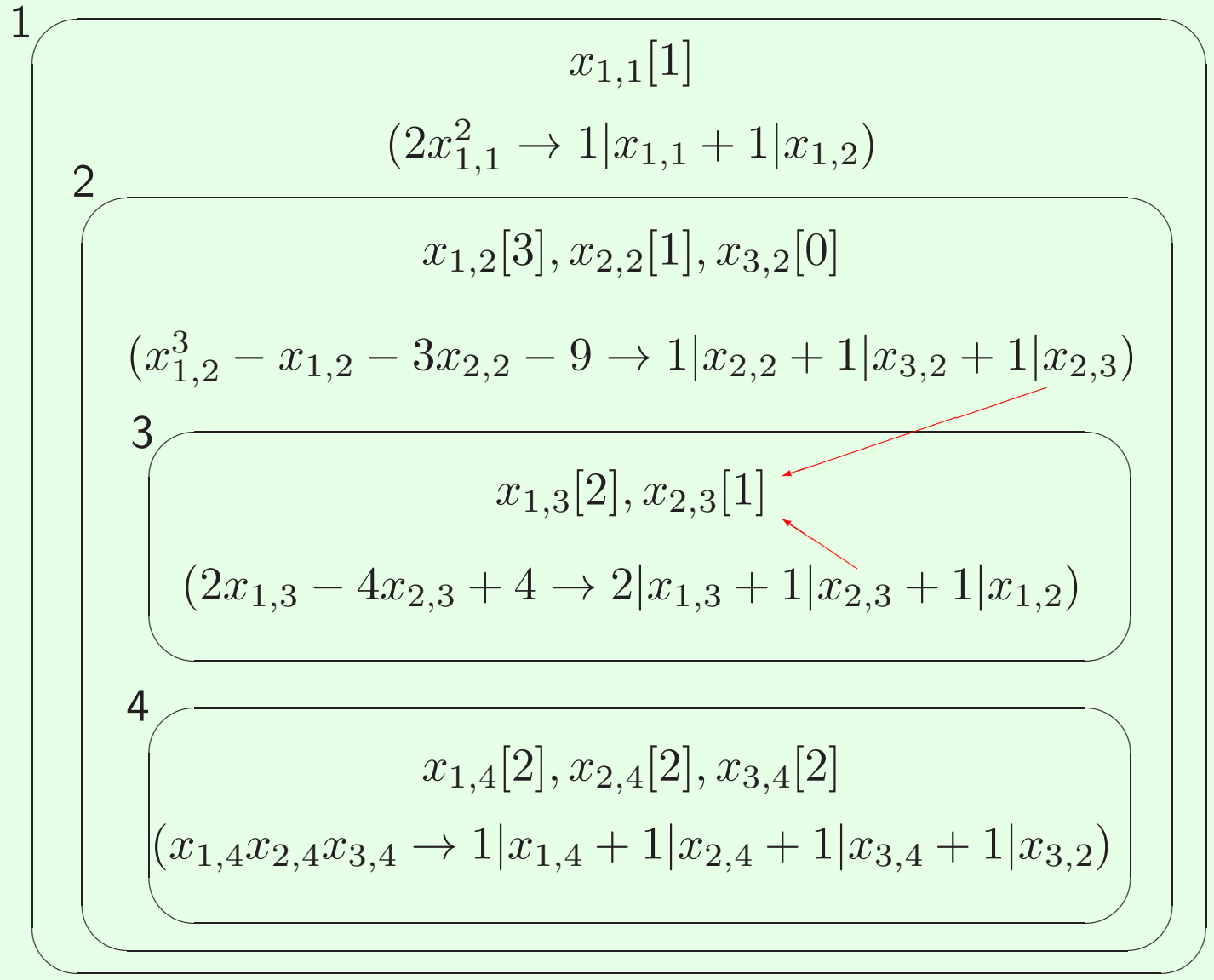
- linguistics (modeling framework, parsing) [Tarragona, Chișinău]

- optimization (membrane algorithms [Nishida, 2004], [many])

- economics ([Warsaw group], [R. Păun], [Vienna group])

- robot control ([Politehnica University Bucharest], [Opava])

Numerical P systems – directly, by an example:



Transition from $t = 0$ to $t = 1$:
Sum of distribution coefficients: $C_1 = 2, C_2 = 3, C_3 = 4, C_4 = 4$
Productions:

$$F_1(1) = 2, \quad F_2(3, 1, 0) = 12, \quad F_3(2, 1) = 4, \quad F_4(2, 2, 2) = 8.$$

"Unitary portions" in each compartment: 1, 4, 1, 2, respectively.
After repartitions:

$$\begin{aligned}
&x_{1,1}(1) = 1, \\
&x_{1,2}(1) = 2, \quad x_{2,2}(1) = 4, \quad x_{3,2}(1) = 6, \\
&x_{1,3}(1) = 2, \quad x_{2,3}(1) = 5, \\
&x_{1,4}(1) = 2, \quad x_{2,4}(1) = 2, \quad x_{3,4}(1) = 2.
\end{aligned}$$

Now, the productions are

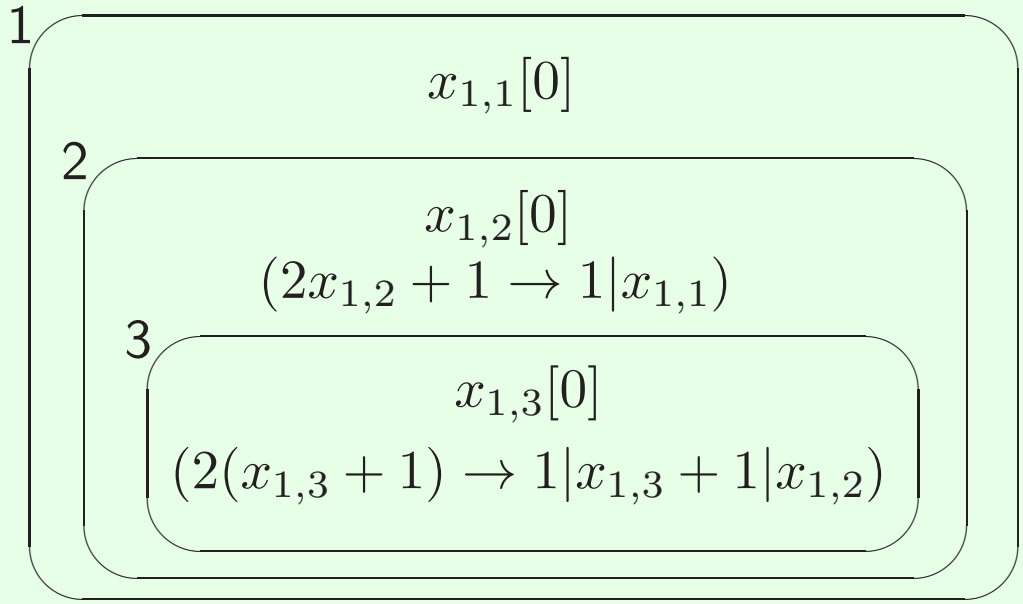
$$F_1(1) = 2, \quad F_2(2,4,6) = -15, \quad F_3(2,5) = -12, \quad F_4(2,2,2) = 8,$$

with the portions $1, -5, -3, 2$, respectively, hence we get

$$\begin{aligned}
&x_{1,1}(2) = 1,\\
&x_{1,2}(2) = -2, \quad x_{2,2}(2) = -5, \quad x_{3,2}(2) = 3,\\
&x_{1,3}(2) = -6, \quad x_{2,3}(2) = -8,\\
&x_{1,4}(2) = 2, \quad x_{2,4}(2) = 2, \quad x_{3,4}(2) = 2.
\end{aligned}$$

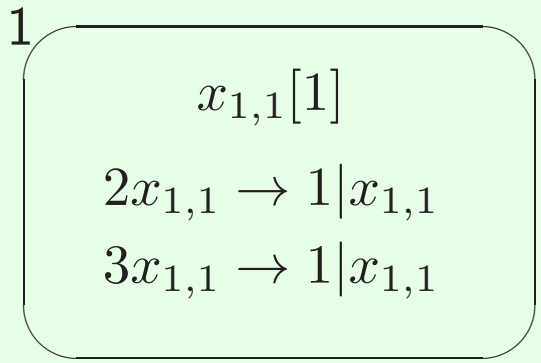
and so on

A simple (deterministic) example:



Variable $x_{1,3}$ increases by 1 at each step, also transmitting its value to $x_{1,2}$. In turn, compartment 2 transmits the value $2x_{1,2}+1$ to $x_{1,1}$, which is never consumed, hence its value increases continuously. We start with all variables equal to 0. Thus, $x_{1,1}$ starts from 0 and continuously receives $2i+1$, for $i = 0, 1, 2, 3, \ldots$, which implies that in n steps we get for $x_{1,1}$ the value $\sum_{i=0}^{n-1}(2i+1) = n^2$, that is, $Set(\Pi, x_{1,1}) = \{n^2 \mid n \geq 0\}$.

A simple (non-deterministic) example:



$Set(\Pi, x_{1,1}) = \{2^i 3^j \mid i \geq 0, j \geq 0\}.$

References:

1. Gh. Păun, R. Păun: Membrane computing and economics: Numerical P systems. *Fundamenta Informaticae*, 73 (2006), 213–227.

2. O. Arsene, C. Buiu, N. Popescu: SNUPS – A simulator for numerical membrane computing. *Intern. J. of Innovative Computing, Information and Control*, 7, 6 (2001), 3509–3522.

3. C. Buiu, O. Arsene, C. Cipu, M. Pătrascu: A software tool for modeling and simulation of numerical P systems. *BioSystems*, 103, 3 (2011), 442–447.

4. C. Buiu, C.I. Vasile, O. Arsene: Development of membrane controllers for mobile robots. *Information Sciences*, 187 (2012), 33–51.

5. A.B. Pavel, O. Arsene, C. Buiu: Enzymatic numerical P systems - a new class of membrane computing systems. *Proc. IEEE Fifth Intern. Conf. on Bio-Inspired Computing: Theories and Applications (BIC-TA 2010)* Liverpool, 2010, 1331–1336.

6. A.B. Pavel, C. Buiu: Using enzymatic numerical P systems for modeling mobile robot controllers. *Natural Computing*, 11, 3 (2012), 387–393.

References (cont.):

7. C.I. Vasile, A.B. Pavel, I. Dumitrache, Gh. Păun: On the power of enzymatic numerical P systems. *Acta Informatica*, 49, 6 (2012), 395–412.

8. C.I. Vasile, A.B. Pavel, I. Dumitrache: Universality of enzymatic numerical P systems. *Inter. J. Computer Math.*, 90, 4 (2013).

9. A. Leporati, A.E. Porreca, C. Zandron, G. Mauri: Improving universality results on parallel enzymatic numerical P systems. *Proc. 11th Brainstorming Week on Membrane Computing*, Sevilla, 2013, 177–200.

10. A. Leporati, G. Mauri, A.E. Porreca, C. Zandron: Enzymatic numerical P systems using elementary arithmetic operations. *Pre-proc. 14th Intern. Conf. on Membrane Computing*, Chișinău, 2013, 225–240.

11. Gh. Păun: Some open problems about numerical P systems. *Proc. 11th Brainstorming Week on Membrane Computing*, Sevilla, 2013, 233–242.

12. Zhiqiang Zhang, Tingfang Wu: A Bibliography of Numerical P Systems, *Bull. IMCS*, 1 (June 2016), 59–61 (34 titles)

Software and implementation:

- P-lingua – Sevilla, Spain:
  `http://www.p-lingua.org/wiki/index.php/Main_Page`

- Parallel simulators for MC on GPU Project – Sevilla, Spain:
  `http://sourceforge.net/p/pmcgpu`

M. García-Quismondo, L. F. Macías-Ramos, M. J. Pérez-Jiménez. Implementing enzymatic numerical P systems for AI applications by means of graphic processing units. *Beyond Artificial Intelligence*, volume **4** of *Topics in Intelligent Engineering and Informatics*, 2013, pp. 137–159.

(Relatively) Formal definition:

$$\Pi = (\mu, (Var_1, Pr_1, Var_1(0)), \ldots, (Var_m, Pr_m, Var_m(0)), x_{j_0,i_0}),$$

where: μ is a membrane structure with m membranes labeled injectively by $1, 2, \ldots, m$, $Var_i = \{x_{j,i} \mid 1 \leq j \leq k_i\}$ is the set of variables from region i, Pr_i is the set of programs from region i, $Var_i(0)$ is the set of initial values for the variables in region i, and x_{j_0,i_0} is the output variable

Program (l from region i):

$$pr_{l,i} = (F_{l,i}(x_{1,i}, \ldots, x_{k_i,i}) \rightarrow c_{l,1}|v_1 + c_{l,2}|v_2 + \ldots + c_{l,n_i}|v_{n_i})$$

"production function" $\rightarrow$ "repartition protocol"

The "production" at time t in compartment i is divided by $C_{l,i} = \sum_{s=1}^{n_i} c_{l,s}$, and the "unitary portions" are distributed according to the coefficients $c_{l,s}$. The programs are nondeterministically chosen. $Set(\Pi, x_{j_0,i_0})$ is the generated set.

Computing power:

Families:
$NNP_m(poly^n(r), nneg, \alpha)$
$m \geq 1$ (membranes), $n \geq 0$ (degree), $r \geq 0$ (variables), $\alpha \in \{div, lost, carry, stop\}$
(nneg = non-negative coefficients, D for deterministic, * for unbounded)

Theorem 1. *(i)* $DNN^+P_1(poly^1(1), nneg, div) - SLIN_1^+ \neq \emptyset$.
(ii) $SLIN_1^+ \subset DNN^+P_*(poly^1(1), nneg, div)$.

Theorem 2. $N^+RE = NN^+P_8(pol^5(5), div) = NN^+P_7(poly^5(6), div)$.

The proof [R. Păun, Gh. Păun, 2006] is based on the characterization of RE sets of numbers as positive values of polynomials with integer values, [Matijasevitch].
Open problems: (i) universality for deterministic systems; (ii) improve the parameters.

Enzymatic numerical P systems: also programs of the form

$$F_{l,i}(x_{1,i},\ldots,x_{k_i,i})|_{e_{j,i}} \to c_{l,1}|v_1 + c_{l,2}|v_2 + \ldots + c_{l,n_i}|v_{n_i},$$

where $e_{j,i}$ is a variable from Var_i different from $x_{1,i},\ldots,x_{k_i,i}$, and from $v_1,\ldots,v_{n_i}$. Applicable at a time t only if $e_{j,i}(t) > \min(x_{1,i}(t),\ldots,x_{k_i,i}(t))$.

Theorem 3. $NRE = NNP_7(poly^5(5), enz, seq)$.

[Vasile, Pavel, Dumitrache, Păun, 2012]

Parallelism:

- $oneP =$ all programs in a region (which can be applied, under enzymes control) are applied, but each variable is used only once

- $allP =$ as above, with each variable appearing in as many times as necessary

Families:
$NNP_m(poly^n(r), enz, nneg, \alpha, \beta)$,
$m \geq 1$ (membranes), $n \geq 0$ (degree), $r \geq 0$ (variables),
$\alpha \in \{div, lost, carry, stop\}$, $\beta \in \{seq, allP, oneP\}$

always $\alpha = div$ (hence omitted)

Theorem 4. $NRE = NNP_*(poly^1(2), enz, oneP) = NNP_4(poly^1(6), enz, allP)$.

[C.I. Vasile, A.B. Pavel, I. Dumitrache, Gh. Păun, 2012],

[C.I. Vasile, A.B. Pavel, I. Dumitrache, 2013]

Theorem 5. $NRE = NNP_1(poly^1(1), enz, allP) = NNP_1(poly^1(1), enz, oneP)$.

Theorem 6. *There are universal enzymatic numerical P systems:*

– *with 31 variables and 61 programs, working in the allP mode*

– *with 156 variables and 105 programs, working in the oneP mode*

– *with 137 variables and 108 programs, working in the oneP mode*

[A. Leporati, A.E. Porreca, C. Zandron, G. Mauri, 2013]

Open problems: (i) what about the non-enzymatic systems? (ii) improvements also for seq? (ii) improve the parameters

Complexity results: [A. Leporati, A.E. Porreca, C. Zandron, G. Mauri, 2013]

Definition: $L \subseteq \{0,1\}^*$ is decided by Π (enzymatic, allP) in polynomial time, if Π contains two variables acc, rej and, after introducing $1x$ (for all $x \in L$) in a specified variable, Π halts in $O(|x|^k)$ time, and:

- if $x \in L$, then $acc = 1, rej = 0$
- if $x \notin L$, then $acc = 0, rej = 1$

Notation: $\mathbf{P} - \mathbf{ENP}(X), X \subseteq \{+, -, \times, \div\}$ (X = operations used in production functions)

Theorem 7. *(i)* $\mathbf{P} - \mathbf{ENP}(+, -) = \mathbf{P}$, *(ii)* $\mathbf{P} - \mathbf{ENP}(+, -, \times, \div) = \mathbf{PSPACE}$.

Open problems: what about $oneP$, seq, non-enzymatic?

Many open problems and research topics:

- considering migrating variables $(F_{j,i}(x_{p_1}, \ldots, x_{p_k}) \rightarrow c_{j,1}|(x_{r_1}, tar_1) + \ldots + c_{j,q}|(x_{r_q}, tar_q))$
- generating strings (symbols are associated with numbers sent to the environment – variable out there)
- controlling the computation by means of thresholds (for variables, for production,...)
- borrowing ideas from SN P systems to numerical P systems, and conversely
- etc.

Applications:

- economics (recent): E. Sánchez Karhunen, L. Valencia-Cabrera, MC Applications in Computational Economics, *BWMC, February 2017*
- robot controlling: Polytechnical University of Bucharest, as well as Opava, Czech Republic:
  C.I. Vasile, A.B.Pavel, I. Dumitrache, J. Kelemen: Implementing obstacle avoidance and follower behaviors on Koala robots using numerical P systems, *Proc. 10th Brainstorming Week on Membrane Computing*, Sevilla, 2012, Fenix Editora, Sevilla, 2012, vol. II, 215–227.

  NEW Andrei George Florea, Cătălin Buiu: *Membrane Computing for Distributed Control of Robotic Swarms: Emerging Research and Opportunities*, IGI Global, 2017
- engineering (the pole balancing problem): D. Llorente-Rivera, M.A. Gutiérrez-Naranjo (BWMC, 2-6 February, 2015, Sevilla, Spain)
- (recent) multi-dimensional decision making: S. Raghavan, K. Chandrasekaran, ACMC 2017 (21-25 September, 2017, Chengdu, China)
- what else? (complexity results are encouraging)

Koala robot

Thank you!

...and please do not forget: 19th Brainstorming Week on MC, February 2018

Bulletin of the IMCS:
`http://membranecomputing.net/IMCSBulletin/`

`http://ppage.psystems.eu`