Romanian Journal of Information Science and Technology
Vol. 10, nr. 4, 2007

Tracing Some Open Problems in Membrane Computing

Gheorghe PĂUN

Institute of Mathematics of the Romanian Academy
PO Box 1-764, 014700 Bucureşti, Romania
E-mail: george.paun@imar.ro, gpaun@us.es

Abstract. Membrane computing is a branch of natural computing aiming to abstract computing models from the structure and functioning of the living cell, and from the way cells cooperate in tissues, organs, or other populations of cells. This research area developed very fast, both at the theoretical level and in what concerns the applications. During the almost ten years since membrane computing was initiated, several open problems were circulated, sometimes in comprehensive lists prepared for meetings in this area. The present notes intend to survey the research related to some of these problems, mentioning both progresses made in solving them and questions which still wait for research efforts.

1 Introduction

Membrane computing was initiated in 1998 (with the seminal paper published in 2000, [39]) and the literature of this area has grown very fast (already in 2003, Thompson Institute for Scientific Information, ISI, has qualified the initial paper as "fast breaking" and the domain as "emergent research front in computer science" – see http://esi-topics.com). Here, we start by a quick and informal introduction to this area, introducing the basic ideas and types of models, the main types of theoretical results and applications. For details, in particular, for technical definitions and proofs, we refer to the large bibliography of membrane computing, as available at http://psystems.disco.unimib.it. One can find there many papers to download, in particular, pre-proceedings volumes of the yearly Workshop on Membrane Computing (the eight edition took place in June 2007 in Thessaloniki, Greece, the next one will be in Edinburgh, UK, and the tenth edition will be organized in Curtea de Argeş, Romania, where this series of meetings started in 2000), as well as the volumes of the Brainstorming Week on Membrane Computing (organized each year at the beginning of February, in Sevilla, Spain).

Membrane computing is a branch of natural computing, the broad area of research concerned with computation taking place in nature and with human-designed computing inspired by nature. Membrane computing abstracts computing models from the architecture and the functioning of living cells, as well as from the organization of cells in tissues, organs (brain included) or other higher order structures such as colonies of cells (e.g., bacteria). The initial goal was to learn from cell biology something possibly useful to computer science, and the area quickly developed in this direction. Several classes of computing models – called *P systems* – were defined in this context, inspired from biological facts or motivated from mathematical or computer science points of view. A number of applications were reported in the last few years in several areas – biology, bio-medicine, linguistics, computer graphics, economics, approximate optimization, cryptography, etc. Several software products for simulating P systems and attempts of implementing P systems on a dedicated hardware were reported; also an attempt towards an implementation in bio-chemical terms is in progress.

The main ingredients of a P system are (i) *the membrane structure*, delimiting compartments where (ii) *multisets of objects* evolve according to (iii) *(reaction) rules* of a biochemical inspiration. The rules can process both objects and membranes. Thus, membrane computing can be defined as a framework for devising cell-like or tissue-like computing models which process multisets in compartments defined by means of membranes. These models are (in general) distributed and parallel. When a P system is considered as a computing device, hence it is investigated in terms of (theoretical) computer science, the main issues studied concern the *computing power* (in comparison with standard models from computability theory, especially Turing machines/Chomsky grammars and their restrictions) and the *computing efficiency* (the possibility of using parallelism for solving computationally hard problems in a feasible time). Computationally and mathematically oriented ways of using the rules and of defining the result of a computation are considered in this case. When a P system is constructed as a model of a bio-chemical process, it is examined in terms of dynamical systems, with the evolution in time being the issue of interest, not a specific output.

From a theoretical point of view, P systems are both powerful (most classes are Turing complete, even when using ingredients of a reduced complexity – a small number of membranes, rules of simple forms, ways of controlling the use of rules directly inspired from biology are sufficient for generating/accepting all sets of numbers or languages generated by Turing machines) and efficient (many classes of P systems, especially those with enhanced parallelism, can solve computationally hard problems – typically **NP**-complete problems, but also harder problems, e.g., **PSPACE**-complete problems – in feasible time – typically polynomial). Then, as a modeling framework, membrane computing is rather adequate for handling discrete (biological) processes, having many attractive features: easy understandability, scalability and programmability, inherent compartmentalization and non-linearity, etc.

At this moment, there are three main types of P systems: (i) cell-like P systems, (ii) tissue-like P systems, and (iii) neural-like P systems.

The first type imitates the (eukaryotic) cell, and its basic ingredient is the *membrane structure*, a hierarchical arrangement of membranes (understood as three dimensional vesicles), i.e., delimiting compartments where multisets of objects are placed; the objects are in general described by symbols from a given alphabet, but also string-objects can be considered; rules for evolving these objects are provided, also localized, acting in specified compartments or on specified membranes. The most common types of rules are multiset rewriting rules (similar to chemical reactions) and transport rules, e.g., symport or antiport rules, inspired by biological processes. The objects not only evolve, but they also pass through membranes (we say that they are "communicated" among compartments). The rules can have several forms, and their use can be controlled in various ways: promoters, inhibitors, priorities, etc. Also the hierarchy of membranes can evolve, e.g., by creating and destroying membranes, by division, by bio-like operations of exocytosis, endocytosis, phagocytosis, and so on.

In tissue-like P systems, several one-membrane cells are considered as evolving in a common environment. They contain multisets of objects, while also the environment contains objects. Certain cells can communicate directly (channels are provided between them) and all cells can communicate through the environment. The channels can be given in advance or they can be dynamically established – this latter case appears in so-called *population P systems*. In the case when the cells are simple, of a limited capacity (as the number of objects they contain or of rules they can use), we obtain the notion of *P colony*.

Finally, there are two types of neural-like P systems. One is similar to tissue-like P system in that the cells (neurons) are placed in the nodes of an arbitrary graph and they contain multisets of objects, but they also have a *state* which controls the evolution. Another variant was recently introduced, under the name of *spiking neural P systems*, where one uses only one type of objects, the *spike*, and the main information one works with is the distance between consecutive spikes.

The cell-like P systems were introduced first and their theory is now very well developed; tissue-like P systems have also attracted considerable interest, while the neural-like systems, mainly under the form of spiking neural P systems, were only recently investigated.

The bibliography of membrane computing contains several lists of open problems and research topics, mainly circulated before the yearly Brainstorming Week on Membrane Computing – see [40], [41], [43], [44], [45], [46], [47]. Also the monograph [42] contains a list of open problems.

Following the trace of these problems is not at all an easy task; some problems were solved, many of them were reformulated several times, according to the progress in the study of notions they involved, others were not addressed yet. Probably not all of them are still actual – although this is not always possible to assess. Anyway, the present discussion is only a way to remind to the reader the fact that membrane computing contains many unsettled questions, many research directions which are worth pursuing. We examine systematically only one of the lists mentioned above, the one from the monograph [42]. In general, we describe informally the problems and the progresses in solving them, with several bibliographical indications, but not fully covering the references available in the literature. Further details can be found at the web page mentioned above, [59], while for technical definitions the reader is advised to consult the monograph [42] and the papers cited below. Anyway, the reader is assumed to have some familiarity with membrane computing basic notions and results.

2 The Problems from [42]

There are 39 open problems and research topics formulated in [42], identified with Q1 – Q39.

The first *problem, Q1*, concerns the computing power of catalytic P systems, with or without using dissolution rules. While for cooperative systems it is proved that they are equivalent with Turing machines (Theorem 3.3.3) and the non-cooperative ones can generate only length sets of regular languages (hence semilinear sets, Theorem 3.3.2), the intermediate case remains open in [42] – one only shows that using such systems with membrane dissolving possibilities we can generate non-semilinear sets of numbers.

Somewhat surprising, it was soon proved that also catalytic P systems are computationally universal. First, systems with eight catalysts were used, [57], then with three catalysts, and eventually, in [20] it is proved that two catalysts suffice. It is still *open* whether or not one catalyst suffices, and the conjecture is that this is not the case. Some partial results were obtained in this respect (see, e.g., [19]) and [28] (in this latter paper it is proved, among others, that P systems with only one catalyst and whose rules are all catalytic compute only semilinear sets).

Adding various additional controls on the use of rules usually leads to universality for catalytic P systems with only one catalyst; this is the case of using priorities, possibilities of handling the permeability of membranes, promoters and inhibitors, and so on. However, the problem is left open in the monograph [42] whether non-cooperative systems with non-cooperative rules and, e.g., priorities are universal (*problem Q2*). Also this problem was solved, this time in the negative way: in this case we obtain a characterization of Parikh images of ET0L (extended tabled interactionless Lindenmayer) languages, [56].

According to your knowledge, the case of other regulations (controlling permeability, using promoters or inhibitors) was not settled; it is expected that also in this case, non-cooperative P systems are not universal.

Problem Q3 is of a more general type, and it concerns the study of non-synchronized P systems. Removing the universal clock (hence the synchronization), and replacing the maximally parallel way of using the rules, were two issues many times formulated as research topics and as features to be removed in order to bring P systems closer to the (biological) reality. The literature related to this research topic is large. For instance, in [17] one considers systems with bounded parallelism (e.g., at most k rules, for given k, are used in each region), in [10] one introduces the so-called minimal parallelism (roughly speaking, if a membrane can use any rule, then at least one is used), while [32] and [18] consider systematically this issue. In general, removing the synchronization decreases the power, but additional features (e.g., bi-stable catalysts, i.e., objects switching between two states) can compensate and the Turing completeness is preserved.

An interesting direction of research was proposed in [8], where time-free and clock-free P systems were considered, those which compute the same set of numbers (or of vectors) with or without synchronization.

Anyway, there are several other results and papers, but this issue still deserves to be investigated – for instance, considering particular classes of P systems (as done in [6]), or relating P systems with other asynchronous computing devices, such as Petri nets.

Problem Q4 deals with P systems with the communication controlled by the concentration of objects (a subset O_c of the alphabet O of objects is considered, and after using evolution rules the objects of O_c are redistributed among membranes in such a way that each region has the same number of objects; objects from $O - O_c$ are not moved). With bi-stable catalysts, such systems were proved to be universal and the question was raised whether this is the case also with usual catalysts, maybe with additional controls on using the rules. This problem is still open. Note the particular way of moving objects across membranes, without using usual target indications – hence the universality of usual catalytic systems does not directly imply the universality of P systems with communication controlled by concentration. A related *question* is *Q7*, which asks for investigating the power of P systems with a bounded number of bi-stable catalysts and communication controlled by concentration (indeed, the universality result mentioned above uses an arbitrary number of bi-stable catalysts).

Problem Q5 asks whether P systems with inhibitors (and one catalyst) are universal, similarly to P systems with promoters (this case is settled in [42]). As expected, the answer was positive, and a proof can be found in [30]. The case of non-cooperative systems with inhibitors was settled in [55], proving the non-universality, and an interesting link between the case of promoters and an open problem in regulated rewriting (concerning the power of so-called random context grammars with limited checking) was established.

The next *problem, Q6*, seems to be still unaddressed, although it is of a real interest. It concerns the possibility of defining the result of a computation in a P system in the external mode, arranging in a string the symbols which are sent to the environment during a computation. Using one catalyst and promoters, in this way one obtains a characterization of recursively enumerable (RE, for short) languages (Theorem 3.7.3 in [42]), but the problem is open for other classes of P systems (known to be universal; characterizations of RE languages are expected also in these cases). The interest for this question lies in the qualitative difference between the internal data structure used in the system (multisets) and the external data structure, that of the result (strings, hence providing positional information).

Problems Q8 and *Q9* concern the power of P systems with symport/antiport rules of a small size (in terms of the number of membranes and the number of symbols involved in each rule). This direction of research was extensively investigated and many results were obtained, some of them proved to be optimal, some of them still not known to be so. A survey can be found in [2], where also the problems not solved yet are pointed out. *Problems Q10* and *Q11* are also about P systems with symport/antiport, and ask about the influence of additional features, such as promoters and inhibitors, or possibilities of controlling membrane permeability. These questions should now be considered in relation with the pretty restrictive results synthesized in [2] – most cases are settled in this way.

Problem Q12 asks for improving a characterization of RE languages in terms of *traces* of a distinguished object in a P system with symport/antiport rules (Theorem 4.4.4 in [42]) and such improvements were soon obtained, e.g., in [25].

Problem Q13 deals with P systems with string-objects, and asks whether or not less than four membranes suffice in order to characterize matrix languages; the question has interesting implications in characterizing RE languages. Improvements were indeed obtained, e.g., in [37], where also results about string-object P systems with replicated rewriting are given (*problem Q14*). The optimality of these results is still open.

P systems with string-objects and parallel rewriting were not too much investigated, and this is the topic of *problem Q15.* Actually, it refers to specific ways of defining the parallel rewriting in such a way to avoid conflicts in target indications of rules. This issue is discussed in some details in [4] (the thesis is available at [59]). A similar case is that of *problems Q16, Q17* which ask for improving universality results form [42] about string-objects P systems with splicing rules; this issue was considered in detail in [58]. Papers by these authors also contain results and further open problems related to these two questions.

Problem Q18 deals with P systems with string-objects processed by contextual rules, *problem Q19* proposes the investigation of closure properties of various classes of languages generated by P systems with string-objects, while *problem Q20* introduces a new type of properties of language hierarchies, called "absorbtion results". We recall this property/problem as in [42]: "Given a (finite or infinite) hierarchy of language families, $FL_m, m \geq 1$, and an operation g with languages – say, an n-ary one – such that the families FL_m are not (known to be) closed under the operation g, it is of interest to find results of the following form: *if* $L_1, \dots, L_n \in FL_m$, *then* $g(L_1, \dots, L_n) \in FL_{m+p_g}$, *for some* p_g *depending on* g. When $p_g = 0$ we have a usual closure property."

While contextual P systems were investigated (not with respect to the previous problem) in a series of papers by Indian authors (e.g., [33], [34]), in several cases dealing with array languages, for the other two questions we are not aware of any paper addressing them.

Problem Q21 deals with tissue-like P systems and asks for improving results from the book. Specifically, the question concerns the power of tissue-like P systems with all cells independent (no direct communication channel is provided), communicating only by means of the environment. This is an important issue in so-called population P systems, [3], and it was also addressed in terms of P colonies, [14]. In turn, *problem Q22* asks for improvements of results about neural-like P systems; we know no attempt to solve this problem (in general, the study of this class of P systems needs further, systematic efforts, but this direction of research was somewhat obscured by the much more intensive investigation of spiking neural P systems, introduced in [29].

The next four *problems, Q23-Q26*, concern a research direction which was only preliminarily explored at the time of writing the monograph [42], but which was very much investigated in the last years: computational complexity issues. This is still an important research direction; besides computability power (competence), the computational efficiency (performance) is a key issue in natural computing. Based on a somewhat standard time-space trade-off (in out case, with the space obtained by using biologically inspired operations, such as membrane division, membrane creation, and string replication), one can devise polynomial solutions to computationally hard problems (typically, **NP**-complete, but also **PSPACE**-complete problems). The related literature is very large, with a lot of results (solutions to particular problems, characterizations of known complexity classes, introduction of new complexity classes, etc.), but also the list of open problems in this area is very large: improving existing results; the relation between uniform and semi-uniform solutions (with algorithms constructed in polynomial time, starting from the instance to solve); the relation between deterministic, non-deterministic, weakly confluent, and strongly confluent solution; separation between efficient and non-efficient classes (do the dissolution, or polarizations, or other ingredients make any difference from this point of view?); compare new complexity classes with classic ones; and so on and so forth. We refer to [52], [50], [51] for details and references (of course, a comprehensive source of information about this vivid direction of research is [59]).

Problem Q27 is also related to the computational complexity, but we mention it separately, because it is of a particular form and interest. With good motivation from biology, we can imagine computing devices solving problems in the following setup. An arbitrarily large device is given in advance, "pre-computed", but "free of information" (the idea is not to cheat, introducing the solution of the problem in the sufficiently large initial configuration of the system). Then, a given instance of the problem to solve is introduced, in a polynomial time (hence also using polynomially many bits of information, objects in out case) in the system. In the and of a polynomially long computation, we get the answer to the problem. Such a scenario is not investigated in the classic computational complexity, but it is typical for natural computing, DNA computing included. Which are the restrictions to impose on the pre-computed system, how much information may it contain, how regular it must be, how to define complexity classes in this framework? Such issues are important also because linear or even constant time solutions to `SAT` were devised, in terms of symbol-object P systems ([16]) and spiking neural P systems ([9]), and also because similar situations appear in biology, e.g., with respect to the brain and the liver, which are "arbitrarily large, pre-computed" organs and only part of them is used in each given moment.

A long standing open *problem* in membrane computing was that of the existence of infinite hierarchies on the number of membranes (*Q28*). Because of the equivalence with Turing machines, this hierarchy collapses in most cases, namely at most at the number of membranes sufficient to simulate a universal Turing machine. Therefore, a non-universal class of P systems should be found for answering this question. Attempts to find such a class were made both by S.N. Krishna and by R. Freund, but a fully satisfactory answer (i.e., involving a class of P systems not visibly defined in such a way to obtain an infinite hierarchy on the number of membranes) was provided by O.H. Ibarra, in [27].

Problem Q29 is related to the previous one (to one of the pre-solutions to Q28), namely it asks the investigation of unary P systems, those systems using only one type of objects in their membranes. This kind of problem was addressed in [48], and the question was then systematically investigated in [1]. Several trade-off results were proved (universality with n objects and m membranes, with given values for n and m), but several cases are still open. In some sense, an answer to this problem is provided also by spiking neural P systems, because they use only one object, the spike.

An interesting *problem*, not fully explored, is *Q30*: in "standard" P systems, the membrane structure is somewhat auxiliary with respect to the multisets, as the processed data structure are the multisets and the result of a computation is defined in terms of multisets; what about reversing the perspective, and consider a cell-like P system as a tree processor. This makes sense in the case of systems with a dynamical membrane structure, making use of operations with membranes (and there are many operations of this kind: creating and dissolving membranes, merge, separate, exocytosis, endocytosis, etc.). The problem was addressed in a series of papers (see, e.g., [21], [31], [53]), but it still deserves further efforts.

Problem Q31 is a very general one: compare P systems with other distributed computing systems. Ambient calculus, stream X-machines, Petri nets were already considered, but much more things remain to be done. References can be found in [42] and at the web address [59].

In turn, *problem Q32* (consider P systems working in the accepting mode) has led to a powerful direction of research, initiated in [15] and much investigated after that ; a survey of results, as well as problems which remain to be solved in this area, can be found in [11] and in [38].

Problem Q33 suggests to investigate classes of P systems with a small number of membranes, possibly non-universal. The problem can be extended to considering systematically classes of small P systems, hence their descriptional complexity in terms of various parameters which can be naturally defined (number of rules, of membranes, size of rules, etc.). Such investigations are somewhat usual in membrane computing, this general research topic is continuously present, without being possible to consider it as solved. For instance, there are a series of classic problems in descriptional complexity (see, e.g., [26]) which were not addressed for P systems.

More specific is *problem Q34*: consider reversible P systems (starting with an adequate definition of reversibility) – we know no answer to this question.

Problem Q35 is again very general and important by the possible applications of the systems answering it: define and examine P systems with "approximate" components, in terms of probabilistic, fuzzy, or rough set theory. Several attempts were made to address this issue, but, in spite of the fact that a dedicated workshop was organized with this topic (see [54]), the results are far from being satisfactory. Taking into account that biologists work with approximations and sort of fuzzy logics ("sufficient nutrients", "excess of enzymes", "not enough reactants", etc.), this direction of research waits for a breakthrough which then is expected to have an important development and significant applications.

Next *problem, Q36*, asks to combine ideas from membrane computing and eco-grammar systems, more generally, from grammar systems area (see [12] for details). At this time, there are several papers of this type, classes of grammar systems with ingredients borrowed from membrane computing, comparisons between families of languages generated in the two frameworks, types of limited parallelism inspired from the cooperation protocols from grammar systems. Also P colonies mentioned above can be considered an answer to this question. On the other hand, the problem is so general that further investigations are possible.

Problem Q37 is a multiple one, all sub-problems being considered "exotic": what about introducing quantum ingredients in P systems (quantum objects, entanglement, superposition, teleportation)? what about reproduction ? (do we need for that to consider a "genome" of a P system, a description of it placed in a distinguished membrane playing the role of the nucleus?); what about computing beyond Turing? The first question was addressed in a series of papers (a survey can be found in [35]), the third one in [7], but we know no paper addressing the second issue (except, indirectly, the paper [13], which opens a large number of related research directions). All these questions need considerable further efforts to clarify them.

Problem Q38 asks for a more systematic study of conformon-P systems, as introduced in [24]; the same author has a series of subsequent papers in this area (see references at [59]).

The last *question, Q39*, refers to gemmation, an operation inspired from biology, introduced in [5] and then investigated in several papers – for which we again refer to [59].

3 Final Remarks

As also said before, these notes mainly intend to call the attention to the many topics which are still unsettled in membrane computing, in spite of the rapid development of this area. The given references are by no means complete (but we have tried to indicate the main sources of information, useful to the reader who wants to investigate any of these issues). Of course, many other open problems and research topics are formulated in various papers, in particular, the ones mentioned in the end of the Introduction. For instance, applications of membrane computing raise a series of problems of different types, related to the time evolution of P systems modeling biological or bio-medical processes (dynamical systems issues, with the important detail that in out case we deal with discrete dynamical systems, hence the differential equations are not the most adequate tool to use). A wealth of research topics and open problems are available in the recently introduced area of spiking neural P systems ([29]), computing devices inspired from the way neurons communicate by means of electrical impulses of a unique form (*spikes*). Paper [47] contains 26 such questions, most of them still waiting to be considered. There are mentioned there problems related to the computing power (accepting or generating numbers or strings, transducing finite strings or infinite sequences), complexity, variants (asynchronous, parallel), applications (pattern recognition), neural computing issues (learning/training), neuro-biology (adding further biological ingredients, such as astrocytes), links with other models (e.g., Petri nets), etc.

References

[1] A. Alhazov, R. Freund, M. Oswald: Cell/symbol complexity of tissue P systems with symport/antiport rules. *Intern. J. Found. Computer Sci.*, 17, 1 (2006), 3–26.

[2] A. Alhazov, R. Freund, Y. Rogozhin: Computational power of symport/antiport: history, advances and open problems. In [23], 1–30.

[3] F. Bernardini, M. Gheorghe: Population P systems. *J. Universal Computer Sci.*, 10, 5 (2004), 509–539.

[4] D. Besozzi: *Computational and Modelling Power of P Systems.* PhD Thesis, Univ. degli Studi di Milano, Italy, 2004.

[5] D. Besozzi, C. Zandron, G. Mauri, N. Sabadini: P systems with gemmation of mobile membranes. *Proc. ICTCS 2001*, Torino, 2001 (A. Restivo, S.R. Della Rocca, L. Roversi, eds.), LNCS 2202, Springer, Berlin, 2001, 136–153.

[6] N. Busi: On the computational power of the mate/bud/drip brane calculus: interleaving vs, maximal parallelism. In [23], 144–158.

[7] C. Calude, Gh. Păun: Bio-steps beyond Turing. *BioSystems*, 77 (2004), 175–194.

[8] M. Cavaliere, S. Sburlan: Time and synchronization in membrane systems. *Fundamenta Informaticae*, 64, 1-4 (2005), 65–77.

[9] H. Chen, M. Ionescu, T.-O. Ishdorj: On the efficiency of spiking neural P systems. *Proc. 8th Intern. Conf. on Electronics, Information, and Communication*, Ulanbator, Mongolia, June 2006, 49–52.

[10] G. Ciobanu, L. Pan, Gh. Păun, M.J. Pérez-Jiménez: P systems with minimal parallelism. *Theoretical Computer Sci.*, 378, 1 (2007), 117–130.

[11] E. Csuhaj-Varju: P automata. In [36], 19–35.

[12] E. Csuhaj-Varjú, J. Dassow, J. Kelemen, Gh. Păun: *Grammar Systems. A Grammatical Approach to Distribution and Cooperation.* Gordon and Breach, London, 1994.

[13] E. Csuhaj-Varju, A. Di Nola, Gh. Păun, M.J. Pérez-Jiménez, G. Vaszil: Editing configurations of P systems, *Proc. Third Brainstorming Week on Membrane Computing*, Sevilla, 2005, RGNC Report 01/2005, 131–155.

[14] E. Csuhaj-Varju, J. Kelemen, A. Kelemenova, Gh. Păun, G. Vaszil: Cells in environment: P colonies. *Journal of Multiple-valued Logic and Soft Computing*, 12, 3-4 (2006), 201–215.

[15] E. Csuhaj-Varju, G. Vaszil: P automata or purely communicating accepting P systems. In [49], 219–233.

[16] E. Czeizler: Self-activating P systems. In [49], 234–246.

[17] Z. Dang, O.H. Ibarra: On P systems operating in sequential and limited parallel modes. *Pre-proc. of the Workshop on Descriptional Complexity of Formal Systems, DCFS 2004*, London, Ontario, 164–177.

[18] R. Freund: Asynchronous P systems and P systems working in the sequential mode. In [36], 36–62.

[19] R. Freund: Particular results for variants of P systems with one catalyst in one membrane. *Proc. Fourth Brainstorming Week on Membrane Computing*, Sevilla, 2006, Fenix Editora, Sevilla, 2006, vol. II, 51–50.

[20] R. Freund, L. Kari, M. Oswald, P. Sosik: Computationally universal P systems without priorities: two catalysts are sufficient. *Theoretical Computer Sci.*, 330, 2 (2005), 251–266.

[21] R. Freund, M. Oswald, A. Păun: P systems generating trees. In [36], 309–219.

[22] R. Freund, M. Oswald, P. Sosik: Reducing the number of catalysts needed in computationally universal P systems without priorities. *Proc. DCFS Workshop*, Budapest, Hungary, 2003, 102–113.

[23] R. Freund, Gh. Păun, G. Rozenberg, A. Salomaa, eds.: *Membrane Computing, International Workshop, WMC6, Vienna, Austria, 2005, Selected and Invited Papers.* LNCS 3850, Springer, Berlin, 2006,

[24] P. Frisco: The conformon-P system: A molecular and cell biology-inspired computability model. *Theoretical Computer Sci.*, 312, 2-3 (2004), 295–320.

[25] P. Frisco, H.J. Hoogeboom: Simulating counter automata by P systems with symport/antiport. In [49], 288–301.

[26] J. Gruska: Descriptional Complexity of Context-Free Languages. *Proc. Symp. on Mathematical Foundations of Computer Science, MFCS*, High Tatras, 1973, 71–83.

[27] O.H. Ibarra: On membrane hierarchy in P systems. *Theoretical Computer Sci.*, 334, 1-3 (2005), 115–129.

[28] O.H. Ibarra, Z. Dang, O. Egecioglu, G. Saxena: Characterizations of catalytic membrane computing systems. *Proc. MFCS 2003*, Bratistalva, Slovakia, 480–489.

[29] M. Ionescu, Gh. Păun, T. Yokomori: Spiking neural P systems. *Fundamenta Informaticae*, 71, 2-3 (2006), 279–308.

[30] M. Ionescu, D. Sburlan: On P systems with promoters/inhibitors. *J. Universal Computer Sci.*, 10, 5 (2004), 581–599.

[31] N. Jonoska, M. Margenstern: Tree operations in P systems and λ-calculus. *Fundamenta Informaticae*, 59, 1 (2004), 67–90.

[32] J. Kleijn, M. Koutny: Synchrony and asynchrony in membrane systems. *Membrane Computing, WMC2006, Leiden, Revised, Selected and Invited Papers*, LNCS 4361, Springer, 2006, 66–85.

[33] S.N. Krishna, K. Lakshmanan, R. Rama: On the power of P systems with contextual rules. *Fundamenta Informaticae*, 49, 1-3 (2002), 167–178.

[34] L. Lakshmanan: *On the Crossroads of P Systems and Contextual Grammars: Variants, Computability, Complexity and Efficiency.* PhD Thesis, Dept. of Mathematics, Indian Institute of Technology, Madras, India, 2003.

[35] A. Leporati: Quantum (UREM) P systems: Background, definition and computational power. *Proc. WMC8*, Thessaloniki, June 2007, 33–56.

[36] G. Mauri, Gh. Păun, M.J. Pérez-Jiménez, G. Rozenberg, A. Salomaa, eds.: *Membrane Computing. 5th Intern. Workshop, WMC2004, Milan, Italy, June 2004. Revised Selected and Invited Papers*, LNCS 3365, Springer, Berlin, 2005.

[37] M. Mutyam: Rewriting P systems. Improved hierarchies. *Theoretical Computer Sci.*, 334, 1-3 (2005), 161–176.

[38] M. Oswald: *P Automata.* PhD Thesis, TU Viena, Austria, 2003.

[39] Gh. Păun: Computing with membranes. *Journal of Computer and System Sciences*, 61, 1 (2000), 108–143 (and Turku Center for Computer Science-TUCS Report 208, November 1998, `www.tucs.fi`).

[40] Gh. Păun: Computing with membranes (P systems): Twenty six research topics. *CDMTCS Technical Report* 119, University of Auckland, New Zealand, 2000.

[41] Gh. Păun: Further research topics about P systems. *Pre-proc. Workshop on Membrane Computing*, Curtea de Argeş, 2001, *Technical Report* 17/01 of Research Group on Mathematical Linguistics, Rovira i Virgili Univ., Tarragona, 2001, 243–250.

[42] Gh. Păun: *Computing with Membranes: An Introduction.* Springer, Berlin, 2002.

[43] Gh. Păun: Membrane computing: Some non-standard ideas. *Aspects of Molecular Computing* (N. Jonoska, Gh. Păun, G. Rozenberg, eds.), LNCS 2950, Springer, Berlin, 2004, 322–337.

[44] Gh. Păun: Further open problems in membrane computing. *Brainstorming Week on Membrane Computing*, Sevilla, February 2004, TR 01/04 of Research Group on Natural Computing, Sevilla University, 2004, 354–365.

[45] Gh. Păun: Further twenty six open problems in membrane computing. *Proc. Third Brainstorming Week on Membrane Computing*, Sevilla, 2005, RGNC Report 01/2005, 249–262.

[46] Gh. Păun: 2006 Research topics in membrane computing. *Proc. Fourth Brainstorming Week on Membrane Computing*, Sevilla, 2006, Fenix Editora, Sevilla, 2006, vol. II, 235–252.

[47] Gh. Păun: Twenty six research topics about spiking neural P systems. *Proc. Fifth Brainstorming Week on Membrane Computing*, Sevilla, 2007, Fenix Editora, Sevilla, 2007, 263–280.

[48] Gh. Păun, J. Pazos, M.J. Pérez-Jiménez, A. Rodriguez-Patón: Symport/antiport P systems with three objects are universal. *Fundamenta Informaticae*, 64, 1–4 (2005), 353–367

[49] Gh. Păun, G. Rozenberg, A. Salomaa, C. Zandron, eds.: *Membrane Computing. Intern. Workshop WMC-CdeA2002, Curtea de Argeş, Romania, August 2002, Revised Papers.* LNCS 2597, Springer, Berlin, 2003.

[50] M.J. Pérez-Jiménez: An approach to computational complexity in membrane computing. In [36], 85–109.

[51] M.J. Pérez-Jiménez, A. Romero-Jiménez, F. Sancho-Caparrini: *Teoria de la complejidad en modelos de computación celular con membranas.* Kronos Editorial, Sevilla, 2002.

[52] A.E. Porreca, G. Mauri, C. Zandron: Complexity classes for membrane systems. *RAIRO - Theoretical Informatics and Applications*, 40, 2 (2006), 141–162.

[53] R. Rama, H. Ramesh: On generating trees by P systems. *Proc. SYNASC 05, Timişoara, Romania*, IEEE Press, 2005, 462–466.

[54] F. Rossello, ed.: *Proc. Brainstorming Workshop on Uncertainty in Membrane Computing.* Palma de Mallorca, Nov. 2004.

[55] D. Sburlan: Further results on P systems with promoters/inhibitors. *Intern. J. Found. Computer Sci.*, 17, 1 (2006), 206–222.

[56] D. Sburlan: Non-cooperative P systems with priorities characterize PsET0L. In [23], 363–370.

[57] P. Sosik: The power of catalysts and priorities in membrane systems. *Grammars*, 6, 1 (2003), 13–24.

[58] S. Verlan: *Head Systems and Applications to Bio-informatics.* PhD Thesis, LITA, Univ. Metz, France, 2004.

[59] P Systems Web Page: `http://psystems.disco.unimib.it`.