From Super-cells to Robotic Swarms: Two Decades of Evolution in the Simulation of P Systems

Luis Valencia-Cabrera, David Orellana-Martín,
Miguel Ángel Martínez-del-Amor, Mario J. Pérez-Jiménez

Research Group on Natural Computing
Department of Computer Science and Artificial Intelligence
University of Sevilla, Spain
Avda. Reina Mercedes s/n, 41012 Sevilla, Spain
{lvalencia,dorellana,mdelamor}@us.es

Summary. Membrane Computing provides machine-oriented models of computation, with types and variants including different elements inspired from living cells. Proven computationally complete from their inception, they also showed their ability to solve computationally hard problems. Thus, it is crucial accompanying the theoretical studies with the practical materialization of these devices. While their real implementation presents serious limitations, a more affordable goal is the simulation of these machines, both to aid in the understanding of the theoretical models, and to provide tools to solve problems in different areas (Biology, Economy, robot control, etc.) by P systems-based models. Several works have analysed the history and state of the art of the simulation tools for P systems, the last one in 2016. Therefore, instead of repeating this information, we decided to provide a brief summary, along with an interactive tool to visualize on-line the evolution of these simulators, intended to stay updated as new simulation tools keep appearing on stage.

1 Introduction

Natural computing is characterised by the fact that it provides *conceptual and practical tools* inspired in living nature, observing phenomena that can be somehow viewed as calculation procedures. Some branches within this area are inspired by evolutionary processes (as genetic, evolutionary algorithms and others) or certain types of animal behaviours (ant/bee colonies, etc.). These types of approaches have been mostly applied to optimization problems through the provision of approximate solutions. Others, instead, focus in lower levels of organization, as it might be the case of DNA computing or Membrane computing, among others.

Since its appearance in November 1998, as introduced in [12, 13], membrane computing has supplied a number of computational devices (P systems, initially called *super-cell systems*), of different types (cell-like, tissue-like, neuron-like, multienvironment, P colonies, kernel P systems, MP systems, etc.), with many variants, each one providing different elements making them sound to attack certain relevant problems from a theoretical or a practical point of view.

While many of these types and variants of P systems have been used to solve computationally hard problems, and a number of them have provided subtle tools to get thinner and thinner boundaries between **P** and **NP** classes (assuming $\mathbf{P} \neq \mathbf{NP}$), other set of types and variants have been useful in the computational modelling of certain real-life phenomena, both at a micro and a macro level.

The design and verification of solutions are tough tasks, especially when dealing with **NP**-complete problems solved with minimalistic variants of P systems trying to reduce the number of ingredients involved in the design of these machines (types and length of rules, structures, etc.) In addition, the computational modelling of complex systems, usually involving a huge number of elements and interactions among them, represents another challenge in terms of the difficulty of understanding how the system is evolving taking into account all the processes involved in such an evolution.

Given this scenario, it is crucial to have as much help as possible to handle these P systems, and software tools here play an essential role. They can aid in many related tasks, which can differ depending on the specific needs. These assistants can be used to define or specify the syntax and semantics of the solutions for hard problems or the computational models for complex systems. Besides, some tools can be used to help in the formal verification of these solutions. In addition, the simulation of the computational devices supports researchers in the task of exploring the implications of the addition or removal of certain elements on the systems under study.

Bearing in mind the needs just introduced, a variety of software applications and tools have been provided along the years. Not surprisingly, there have been a number of publications explaining those developments. Besides, several works have been presented reviewing the state of the art in the simulation, since the early days of membrane computing up to now. Consequently, it will not be the goal of this survey to provide a new analysis of the same tools. Instead, we will go very briefly over the main facts on the evolution of the simulators, recalling the main sources where any reader can find deep explanations about every simulator, and will present some on-line tool where all the main simulation tools are placed along a time line, starting in 1998 to the present time, and aiming to continue growing in the future.

The rest of this work is consequently devoted to this brief outline of the milestones the discipline of Membrane computing has gone through. Section 2 summarizes the initial steps in the simulation of P systems, from 1998 to 2004. After that, Section 3 describes the main facts related with the period between 2005 and 2010. Then, Section 4 enumerates the most recent developments introduced in this decade, from 2011 to 2017. In Section 5, an interactive tool to show the evolution in the simulation of simulator of P systems is presented. Finally, Section 6 analyses the present and some prospects about the future of the simulation in Membrane computing.

2 Early days in Membrane Computing

Almost two decades ago, the appearance of Membrane computing in 1998 [12] opened a fascinating branch of Natural computing bringing elements from its cellular inspiration, formal languages theory, Turing and the essence of computability models. The *super-cell systems* introduced at that moment (re-named definitely as **P systems** in 2000 [13]) provided characterisations of recursively enumerable languages. The computational power of these systems was not studied in that first work, but while the computational completeness was obtained "without the need of a **parallel** synchronized evolution" of objects and membranes, it was highlighted there the intuition that the theoretical model proposed, taking advantage of the inherent parallelism present in biological cells, could provide tools to explore an exponential search space working non-deterministically on a number of "processors". Later works would prove the actual ability of some variants, as P systems with active membranes, to solve hard problems efficiently, providing a deep analysis in terms of computational complexity and the power of these devices. Besides this intuition, since this very first foundational technical report, the remarks section pointed towards the possibility of a practical implementation of *super-cell systems*, in biological or electronic media. At that respect, the document emphasized the importance of looking for answers to many open problems by **directing the theoretical studies to the most promising and practically relevant path**.

The need of implementing or at least simulating these theoretical devices was therefore always there from the inception of the discipline of Membrane computing. Consequently, the first simulators for P systems were developed soon after the appearance of the foundational work [12], being submitted in 1999, before the first regular paper [13] presenting the discipline (see [15, 17]). As pointed out in a first survey work about simulators [6] (included in [2]), the focus in this first stage was to try to understand better the recently created computing model, with both pedagogical and research purpose, acting as assistants to study different computations of P systems, helping in tasks as design and verification of these cellular solutions, possibly suggesting invariants that could help in proofs.

As also highlighted in that survey of software available up to 2004-2005 [6], some restrictions were present when simulating P systems: on the one hand, the non-determinism of these devices cannot be in essence captured, but somehow simulated by the generation of *pseudo-random numbers*; on the other hand, a crucial aspect in Membrane computing gets definitely constrained or significantly reduced when simulating in digital computers: the massive parallelism inherent at different levels in P systems (objects inside each region evolving simultaneously - not to mention the evolution of different parts of a string in the case of string-based systems -, and every compartment - membrane, cell or neuron - possibly evolving at the same time). Despite those practical limitations, especially present in the early days, the first generation of simulators for cellular systems could provide interesting tools for teaching and mostly research, aiding to a better understanding, valuable support for theoretical research and assistant for verification. For those interested in the details of these initial works, we strongly recommend the reading of [6]. They were extensively analysed in that book chapter, and it would make no major sense to repeat the work conducted there, so we will simply enumerate the main contributions of those years in each direction.

First of all, in those early days appeared the first P system simulators for transition P systems: in [15] they used `LISP` to build a simulator aimed to address real-life problems (as the Brusselator model or initial models in ecological systems, see [16]). and [17], in `Prolog`. They were followed closely by [18] (in `Scheme`, based on the formalization presented in [19]) and [20] (in `Haskell`, based on previous works analysing algorithmic aspects, the framework and the data structures used, see [21, 22, 23, 24, 25]). After these ones, a new simulator for transition P systems using `Java` was presented in [26]. Along with these sequential programs, the first two parallel simulators were presented, with the aim of partially exploiting the capabilities of the theoretical models in terms of efficiency: a first one [27, 28] was implemented in `C++`, making use of `MPI` to communicate threads, while the other one [29], implemented in `Java`, made use of `RMI` for the communication among processes in different computers.

After the first two simulators mentioned [15, 17], simultaneously to the contributions just outlined, some developments were made for additional types of cell devices. That was the case of P systems with active membranes (as defined in [1]), with a first simulator [30] (also see [14]), in `Visual C++`, that also worked with catalytic cell systems. After that two new simulators were presented, one using `CLIPS` in [31] (to solve problems as Bin-packing [32] or CAP [33]), and another contribution using `Prolog`, in [34], to solve other **NP**-complete problems (see [35, 36, 37]), and also including a tool to analyse the descriptive complexity of P systems, based on the so-called *Sevilla carpets* (see [38, 39, 40]).

A relevant contribution was also made within other types of P systems, as it would be the case of the implementation of catalytic P systems presented in [41], and the simulator for maximally parallel multiset-rewriting systems with promoters/inhibitors in [42], used to prove theorems presented in [43, 44]. Other simulator developed during those early years in Membrane computing was SubLP-Studio, for both L and P systems (see [45], available at [138]). It included some graphical components showing the growth of plants [46, 47].

Besides complete simulators, other interesting tools appeared during that period, as a library (in `C` language) to ease the representation and simulation of P systems [48].

Finally, in that period some interesting contributions appeared trying to get closer to the implementation of P systems *in silico* through hardware components, as the FPGA based implementation described in [49], were they cited as possible precursor [50].

A much broader explanation of the main milestones of this period and details over the simulators can be found at [6]. Besides, an online tool has been provided to follow the evolution in an interactive way, also providing further details and references, in [139].

3 A second stage: from particular to general

The previous section outlined some facts related with the initial steps in Membrane computing. It implied an exciting period where the emergence of the discipline led to the growth of the research community, the appearance of tools to support the theoretical findings, and some case studies related with **NP**-complete problems and with modelling of biological phenomena started to shed some light about future lines for the practical use of P systems tools in short term. The theories, algorithms and software developed laid crucial foundations for the new stages, clarifying the challenges to face when facing the simulation of the types of machines designed within Membrane computing, sound structures to handle these devices and technical constraints imposed by the state of the technology.

During the following few years, from 2005-2006 until 2010, we witnessed an explosion in the number of types of variants of P systems, from classical cell-like to tissue-like and neuron-like P systems, along with new stochastic and probabilistic approaches, involving environments and other elements enriching the *conceptual tool-kit* of frameworks to solve problems by means of cellular systems. This burst came together with a proportional increase in the number of simulation tools to help managing the different problems addressed during this period. As the number of possible domains to apply new variants of P systems was growing, the choice of developing new software tools for each specific problem or sub-typology started to be at least questionable, given the significant effort for a limited profit. It was therefore the moment of changing the focus of the simulators from specific to general purpose tools, able to deal not with a P system designed for a specific problem, but software to manage as many machines as possible within a certain framework inside Membrane computing, or even providing general functionality applicable to devices as different as cell-like, neuron-like or multi-environment P systems. In most cases, during the previous stage, ad-hoc software tools were developed for checking specific models. This idea was progressively turning into more general views and tools. Besides general purpose tools, a significant feature characterised several simulators in this generation: the application to real-world problems in biology. For a detailed analysis of the software for P systems until 2010 we recommend [3], providing an exhaustive survey within the Oxford Handbook on Membrane Computing [3], most complete collective volume to date. In addition, due to the permanent evolution of the discipline, it would be also advisable to check the P systems web site [137] for updated information about software simulators and tools.

As mentioned in previous sections and detailed in Section 5, we have tried to present in this bulletin a collection of the main references in web format, including the main achievements in each period. We strongly recommend the reader to contact the author if you detect any simulation tool within Membrane computing missing in this site that you consider convenient to include, so that we can maintain it as complete as possible, as a valuable resource for the community and those others interested in Membrane computing and especially those focused in the simulation of its devices, P systems.

In what follows, we will briefly over a non-complete list of significant contributions to this period, but a more exhaustive list can be found in the sources cited and the interactive tool provided in [139].

During the last few years of the first generation, some works had started exploring new paths in the application of Membrane computing. Along with all these simulators presented in that phase for classical P systems, where maximal parallelism was generally applied as the execution strategy, it took place the appearance of the first tool [51] for the modelling of biological processes [53] through a variant of P systems (from [52]) including probabilistic elements applied for the execution of the rules. A different approach (unless with a development inspired in [15, 17, 30]) emerged in those years by the University of Verona, leading to the development of `PSim` (available since 2004 in [138], and described years later in [54, 55]).

Along the year 2006, different initiatives led to the development of several simulators from different places of a growing community. Thus, in a joint effort between Sevilla and Sheffield two implementations of multi-compartmental Gillespie's algorithm to simulate stochastic P systems (in `C` and `Scilab`, respectively) was delivered [143], along with a tool [57] (written in `Java`) for the manipulation of `SBML` files from `CellDesigner`. The framework designed was applied to several relevant papers in the modelling of biological phenomena at a *micro* level [58, 59, 60, 60]. Other significant contributions of this generation were made by the *ad-hoc* simulators (available at [138]) developed by Cazzaniga-Pescini (written in `C`, with parallelism given by `MPI`) to simulate certain stochastic processes in biology (see [62, 64, 63, 65, 66]).

Further developments were made during those years applied to the modelling of biological phenomena, as the simulator of stochastic processes `Cyto-sim` [67, 145], based on the formal model presented in [68] .

Another important field related with P systems appeared between 2004 and 2006, *membrane algorithms* [70, 71, 72, 69], providing new approaches to attack **NP**-complete optimization problems using ingredients from Membrane computing. In [69] some solutions for `TSP` were presented, along with a software tool (in `Java`), to simulate several variants of these devices.

Along with the previous tools for cell-like P systems, many others emerged during this period for other types of devices. Thus, in [73] it was presented a simulator for tissue-like P systems [74], formalism presenting a graph structure (instead of the tree-based one of cell-like P systems), used to solve **NP**-complete problems as in [75]. Similarly, in 2008 a tool for the simulation and verification of Spiking neural P systems [76] was developed [77]. Additionally, a third novel type of P systems, the *conformon P systems* [79, 80], received a special attention, hence leading to the appearance of a first simulator [78] to handle them, being applied to the study of biochemical processes with implications in medicine [81].

Apart from pure simulators, other tools were also developed during thee period. Thus, concerning the graphical representation of P systems in relation with plant modelling, several works [82, 83, 84, 85] deepened into the works conducted by *Geourgiou et al.* in the previous generation. Other works were devoted to the simulation of P systems by other computing models, for instance by means of multi/agent systems [86], aiming to exploit the inherent parallelism of membrane computing solutions through a distributed application.

Another great example of the trend within this period, as a paradigmatic case passing from particular to general, it is worth highlighting the open source project Xholon [146], general-purpose software for modelling, transformation, and simulation not only for P systems but for a broader range of computing models including finite state machines, cellular automata and agent based systems. Based on XML and Java, it supports the Unified Modeling Language (UML) 2.1, systems biology modelling including SBML, and other models and tools.

And we could not finish our overview of this period without mentioning another ambitious open source project within the area of Membrane computing: `P-lingua` [87, 89]. It aimed to become a standard for the specification/definition of P systems, providing a whole framework including from a programming/specification language for different types and variants of P systems, to a number of simulators for those variants, with different tools for parsing/debugging or export of P systems in different formats to increase interoperability with other software.

To end up with this stage, it is worth mentioning a remarkable fact, while not representative from the period and not software related, given its possible significance for the future: a first *in vitro* "implementation" of a kind of membrane computing device, using test tubes as membranes and DNA molecules as objects, evolving under the control of enzymes. A more precise introductory explanation can be found at [7], and the details in [90, 91]. Despite some difficulties and limitations, there is no doubt about the relevance of the achievements reached by that project.

4 A third step: the era of applications

As explained in the previous section, the second period in the simulation of P system implied the development of general purpose simulators and tools, aimed to provide the P systems researchers and developers software that they could adapt to analyse their specific P systems under study, instead of being developed for a specific system. The next step in the natural evolution of these devices is, not surprisingly, crossing the barrier of the assistants for the Membrane computing community, so that they can act as useful tools for people working in other disciplines, from researchers to managers in companies or institutions, that can take advantage of the solutions based on P systems to solve their problems, not focused in the internal details provided by these tools but receiving the software delivered as black boxes that they can use as practical aids in their decision-making processes. It is what we could somehow call the era of applications, putting the results of our research and findings at disposal of a much broader community of researchers and, in general, at disposal of the society.

It is unknown in Membrane computing community if we will see a real implementation of P systems, beyond the attempts commented in previous stages, with their actual power and efficiency in the new few decades. We do not know if we will have those machines able to solve **NP**-complete problems in polynomial time, in many cases even linear time, but does that not necessarily mean we will have to wait until that moment in biochemical technology to find some relevant use of P systems. As Babbage kept working on his ideas, not simply waiting until the precise moment when Turing, Von Neumann and their contemporaries witnessed the first electronic computers based on similar principles, membrane computing must keep moving, finding new ways to provide a step further. Researchers keep looking for frontiers of the practical tractability of very relevant problems through different variants of P systems, and explore ways for their practical implementation. But additionally we can say today that we have provided some conceptual and practical tools that can help solving real problems, as the computational modelling of certain phenomena that have been shown to be approachable with the techniques and software tools developed. Novel modelling approaches and conceptual and software tools have been delivered, some of them showing desirable features more suitable to address certain modelling problems that would be significantly harder to handle with more traditional tools. In the last decade we have seen different examples in this sense both at micro and macro levels.

Last year, 2016, a new survey was published analysing the state of the art in the simulation of P systems [8], providing a rich list of sources up to 2016, so it would not be worth detailing the same study here, but we will briefly outline some facts of the most recent progress in this topic. Additionally, as the branch of Membrane computing and in particular the simulation of P systems seems to be maintaining a good health, we will add some more recent advances appeared within the last year.

Some of the main results within this third stage were, not surprisingly, consequence of the achievements in the previous stage, to develop the conceptual and software tools deepening into the study of different modelling approaches. A good example of this transition from previous works was `MetaPlab` [142], a computational framework for MP systems, moving several steps forward wit respect to the first simulator [54] based on the metabolic algorithm introduced in [56], developed in the previous stage. Many successful applications of *MP systems* can be found at [141, 142], with special attention at biomedical studies at a micro level.

Another exponent of this trend was the development of a family of software applications (`EcoSim`) to model and simulate the population dynamics of different ecosystems, by using the simulation engines provided by `P-Lingua` framework

[87, 89, 147]. The identification of common needs for the simulation of those different phenomena, along with any other P systems covered by the framework, led to the appearance of `MeCoSim` [92, 148], allowing the customisation, by simple configuration, of apps adapted to each particular problem, having all the infrastructure of `P-Lingua` framework available for parsing, debugging, simulation, verification, and providing a GUI with custom input and output tables, charts, graphs, visualization of structures and connectors with external tools as invariants detectors or model checkers. The disposal of this generic infrastructure boosted the study of different real problems, mostly using probabilistic P systems (known as *PDP systems*) applied to ecosystems ([94, 95, 96], among many others) and other fields ([97, 98, 99]), once the researchers could focus on the specificities of their particular problems and not in the software implementation of the tools needed. Additionally, a protocol was designed [100] to standardise the modelling process by PDP systems, from the problem to the software solution customised for the specific scenario under study. Another work [101] focused on the calibration of the parameters in ecosystems modelling based on P systems. A recent book chapter [11] summarised the main facts and case studies in PDP systems modelling.

And if at a macro level the simulation of ecosystems through P systems following a probabilistic approach experienced a significant growth with major achievements, at a micro level the success of the studies within systems and synthetic biology using P systems following a stochastic approach was remarkable. These significant contributions were supported by solid tools developed with `Infobiotics Workbench` [102, 149]. This software, aimed *"to facilitate the incremental modelling and rapid prototyping of multi-compartment systems"*, accepting two languages to describe the biological models: an extension of `SBML` (that could be generated from `CellDesigner`) and a domain specific language (DSL), implementing lattice population P systems [93]. As the authors mention, *"the reactions described* (using the DSL language) *in these models can be organized in modules (parametrisable sets of reactions), which promote (sub)model reuse and hence facilitate debugging of model entities capturing biological functions"*.

Some other works appeared during this period, focused on the simulation of numerical P systems, as it was the case in [103] or [104]. It was applied to the simulation of robots in [105], and later simulators were developed in sequential [151] and parallel [106] variants (this one based on GPU cards).

A later contribution, by 2014, to the practical application of membrane computing in biology, making use of P-Lingua framework, was `MeCoGUI` [150], incorporating *probabilistic guarded P systems*[107]. This application was used recently for the simulation of the ecosystem of *Pieris napi oleracea* in eastern North America[108, 152]. Additionally, another interesting application related with the study of regenerative processes using these tools was recently published [109].

The following year, 2015, other biological applications at a micro level appeared. Thus, a new simulator of P systems was developed in [110] to study the evolution of the resistance of certain organisms/parasites against antibiotics. Simultaneously, in [111] the authors studied new techniques for solving a Mitogen Activated Protein Kinases (MAPK) cascade by means of P systems, using P-Lingua and MeCoSim. Other applications emerged in 2016 and 2017 [112, 113, 114, 115] and will continue appearing. Additionally, new approaches for the simulation of P systems keep coming up, as it is the case of an interesting agent-based simulator for *kernel P systems* with division rules [116], from the same team that also developed `kPWorkbench` [117, 153], providing another interesting set of tools for the simulation and verification of kernel P systems.

To continue with a fair outline of the achievements on this era, we cannot reduce our mentions to biological applications, because a big amount of efforts and remarkable results were made within a complementary research line: the development of simulation tools taking advantage of the inherent parallelism of P systems by their simulation on parallel architectures. An extensive survey on this topic was presented in 2015 [9], summarising the simulation tools making use of parallel technologies, more specifically those based on GPU devices. Among other works detailed there, we can recall the line followed along PMCGPU project [154]. Please look for the details about parallel simulators in [9], including among others [118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128]. Some more recent results appear after that surveys, including [10, 129].

Apart from those novel technologies applied to the simulation of different types of P systems, the research line started with `P-Lingua` framework kept including new variants along the years, as tissue P systems with cell division [130], spiking neural P systems [131], tissue P systems with cell separation [132], cell-like P systems with symport/antiport rules [133], and more recently cell-like SN P systems [134].

And if we started this survey talking about *super-cells*, the original name of P systems that opening the title of the paper, we think it would be fair closing this overview of simulators in membrane computing paying a special attention to one of the most recent promising research lines within the area: the control of robots and robotic swarms. Over the last few years this path is being intensively explored in different groups from Romania to China. Thus appeared in 2015 `Lulu` [135], an open-source simulator for P colonies, XP colonies and colonies of XP colonies (P swarms), applied to the control of single and multiple robots. See [155] for additional information about the software and its use, that can be downloaded from [156]. In 2016 an application of XP colonies with robotic swarms appeared [136], and this year, 2017, they presented a book [4] collecting the most relevant information from their works about distributed control of robotic swarms using membrane computing.

In the last few years, other works have analysed the different simulation tools appeared in membrane computing over the last two decades. As far as we are concerned, the last survey is [8] in 2016, as commented at the beginning of this section. That work provides an excellent source, listing the main contributions, describing the simulators and classifying them in categories depending degree of generality, applications addressed, etc. As the authors highlight, their paper *is an attempt to provide the details of the tools that are available for membrane* *computing... On one hand we have tools that are being used for specific type of P Systems or the tools which have a specific application. On the other hand there are tools which are comparatively generic in nature. Further this paper lists the tools that have been designed and developed to be used for the biological applications of P Systems.* In fact, the idea of deploying in this work a web tool aiming to stay updated came after reading this extensive analysis in that survey, which also included a useful timeline from 2000 to 2015. We observed the convenience of such a line to visualize the evolution in the simulators, and at the same time we saw it was practically impossible to provide all the tools in a static picture, both due to its inherently dynamic nature (given the number of tools still under construction and to be developed), and because of the space limitation. Thus, we considered an interactive tool aiming to keep updated could fill that gap for members of membrane computing community and for people approaching the discipline for the first time and aiming to see what happened in the past and what is going on up to now.

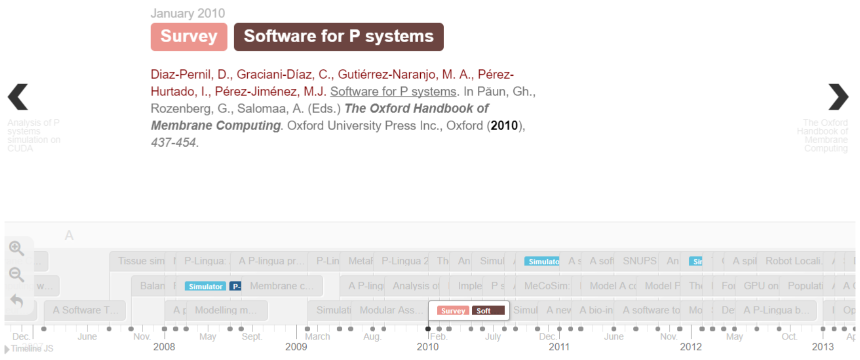
5 An on-line tool to explore the timeline

Along this work we have recalled the main periods in the evolution of Membrane computing, in relation with the development of software tools for the modelling, simulation and verification of P systems, as well as their use for the practical solution of relevant problems. The main references within this field has been outlined, pointing towards collective volumes and survey works have discussed in more depth each of the periods since the inception of the discipline in 1998.

As made significantly evident in previous sections, the last two decades have been very prolific in terms of the number of tools for the modelling and simulation within the area of membrane computing. A good measure is the number of works outlined throughout this paper, and the presence of several survey papers every few years to try to summarise the new contributions taking place and highlight the main achievements within each period.

Thus, it was not the goal of this work in the bulletin to repeat those deeper works in the analysis of the simulation tools, but simply putting all of them together, trying to show the whole picture. And as this whole picture is dynamic and will continue growing, we thought a suitable tool to handle this evolution in the simulation tools would be an interactive tool that we could keep updated. Thus, a web resource has been prepared and put at disposal of the community, in order to have an updated source with the main results in the simulation of P systems in an interactive way.

Thus, the tool shared online is as shown in Fig. 1, presented as a separate web. As we can see, it presents a line that we can follow, move, zoom in/out, etc. to explore the evolution of the different contributions over the years. Additionally, the central part shows the details of a specific publication or site, some labels depending on the type of resource, and possible links to external resources. We can click on each separate publication in th timeline, or start navigating to the left or right to go through each single register sorted by publication date.



January 2010

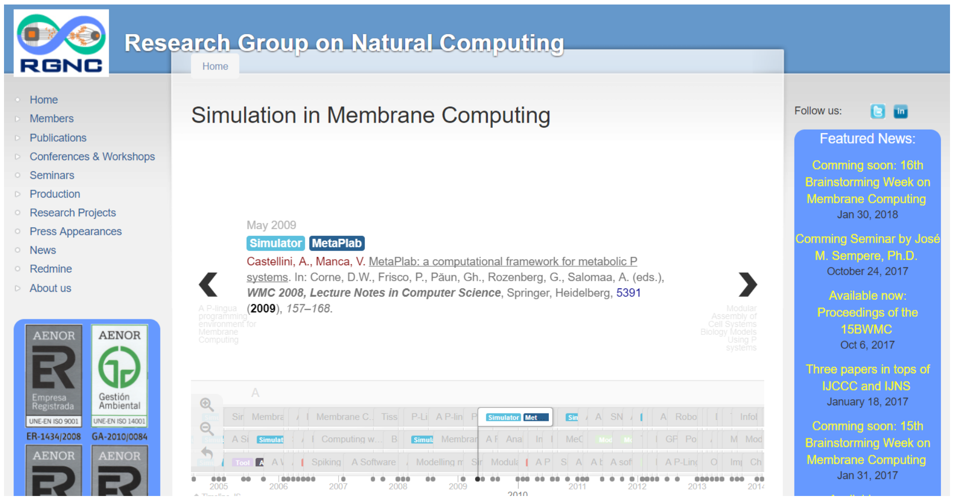
Survey Software for P systems

Diaz-Pernil, D., Graciani-Díaz, C., Gutiérrez-Naranjo, M. A., Pérez-Hurtado, I., Pérez-Jiménez, M.J. Software for P systems. In Păun, Gh., Rozenberg, G., Salomaa, A. (Eds.) *The Oxford Handbook of Membrane Computing*. Oxford University Press Inc., Oxford (**2010**), *437-454*.

Analysis of P systems simulation on CUDA
The Oxford Handbook of Membrane Computing
Dec. June Nov. May Sept. March Aug. Feb. July Dec. June Nov. May Oct. Ap
2008 2009 2010 2011 2012 2013
Timeline JS

Fig. 1. Timeline - isolated

Additionally, the inclusion of this timeline within any other web resource is straightforward, and we not only allow its sharing through other sites but also encourages the community to do it. Please do not hesitate to contact the authors to share the minimal details needed to include this resource in your site if considered convenient (of course, the styles to present the same tools in other sites can be changed through `css`). In Fig. 2 we can see the timeline included in the site of the Research Group on Natural Computing.



Research Group on Natural Computing

Simulation in Membrane Computing

May 2009

Simulator MetaPlab

Castellini, A., Manca, V. MetaPlab: a computational framework for metabolic P systems. In: Corne, D.W., Frisco, P., Păun, Gh., Rozenberg, G., Salomaa, A. (eds.), *WMC 2008, Lecture Notes in Computer Science*, Springer, Heidelberg, 5391 (**2009**), *157–168*.

Featured News:

Comming soon: 16th Brainstorming Week on Membrane Computing
Jan 30, 2018

Comming Seminar by José M. Sempere, Ph.D.
October 24, 2017

Available now: Proceedings of the 15BWMC
Oct 6, 2017

Three papers in tops of IJCCC and IJNS
January 18, 2017

Comming soon: 15th Brainstorming Week on Membrane Computing
Jan 31, 2017

Fig. 2. Timeline - embedded

This timeline tool has been deployed in the website of RGNC at `http://www.gcn.us.es/SimulationMC` [157].

6 Conclusion

The crucial role played by the tools to aid in the tasks of modelling, simulation, verification, etc. of models within the area of membrane computing has been made clear along the last two decades. The evolution of the tools from pedagogical helpers to the current frameworks for the actual solution of practical problems, real applications, reinforces the relevance of these tools to complement the theoretical studies essential for the robustness of the achievements in the discipline.

As highlighted in the previous sections, the aim of this survey is to serve as an updated source for members inside and outside the community of membrane computing interested in the simulation of P systems, so we encourage again the reader to contact the authors if some errors or missing resources are detected and you consider they should be included in this text, considering it is within the scope of this work.

7 References

In what follows, the main contributions to this work are listed under different headings (collective volumes, surveys and extensive bibliography), depending on their category. In these three lists the publications are sorted in order of appearance within the text, which is mostly chronological inside each category of publication.

References

Books and collective volumes

1. Păun, Gh. *Membrane Computing. An Introduction.* Springer, 2002.
2. Ciobanu, G., Păun, Gh., Pérez-Jiménez, M.J. (eds.) *Applications of Membrane Computing.* Springer, 2006.
3. Păun, Gh., Rozenberg, G., Salomaa, A. (eds.) *The Oxford Handbook of Membrane Computing.* Oxford University Press, New York, 2010.
4. Florea, A.G., Buiu, C. *Membrane Computing for Distributed Control of Robotic Swarms: Emerging Research and Opportunities*, Hershey, PA: IGI Global, 2017.
5. Zhang, G., Pérez-Jiménez, M.J., Gheorghe, M. *Real-life Applications with Membrane Computing.* Series: Emergence, Complexity and Computation, **25**, Springer, 2017.

Surveys

6. Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A. Available membrane computing software. In Ciobanu, G., Păun, Gh., Pérez-Jiménez, M.J. (eds.) *Applications of Membrane Computing*, Springer, Heidelberg (2006), 411–436.

7. Díaz-Pernil, D., Graciani-Díaz, C., Gutiérrez-Naranjo, M.A., Pérez-Hurtado, I., Pérez-Jiménez, M.J. Software for P systems. In Păun, Gh., Rozenberg, G., Salomaa, A. (eds.) *The Oxford Handbook of Membrane Computing*, Oxford University Press, 2010, 437–454.
8. Raghavan, S., Chandrasekaran, K. Tools and Simulators for Membrane Computing - A Literature Review. In: Gong M., Pan L., Song T., Zhang G. (eds.) Bio-inspired Computing – Theories and Applications. BIC-TA 2016. *Communications in Computer and Information Science*, **681**, Springer, Singapore, 249–277.
9. Martínez-del-Amor, M.A., García-Quismondo, M., Macías-Ramos, L.F., Valencia-Cabrera, L., Riscos-Núñez, A., Pérez-Jiménez, M.J. Simulating P Systems on GPU Devices: A Survey. *Fundamenta Informaticae*, IOS Press, **136** (2015), 269-284.
10. Martínez-del-Amor, M.A., Macías-Ramos, L.F., Valencia-Cabrera, L., Pérez-Jiménez, M.J. Parallel simulation of Population Dynamics P systems: updates and roadmap. *Natural Computing*, **15**, 4 (2016), 565-573.
11. Zhang, G., Pérez-Jiménez, M.J., Gheorghe, M. Data Modeling with Membrane Systems: Applications to Real Ecosystems. *Real-life Applications with Membrane Computing*, Springer International Publishing, 2017, 259-355.

Extensive bibliography

12. Păun, Gh. Computing with membranes. *Turku Center for Computer Science*, TUCS Technical report 208 (1998), 1–42.
13. Păun, Gh. Computing with membranes. *Journal of Computer and System Sciences*, **61**, 1 (2000), 108–143.
14. Păun, Gh. P systems with active membranes: Attacking **NP**-complete problems. *Journal of Automata, Languages and Combinatorics*, **6** (2001), 75–90.
15. Suzuki, Y., Tanaka, H. On a LISP Implementation of a Class of P Systems. *Romanian Journal of Information Science and Technology*, **3**, 2 (2000), 173–186.
16. Suzuki, Y., Fujiwara, Y., Tanaka, H., Takabayashi, J. Artificial life applications of a class of P systems: Abstract rewriting systems on multisets. In Calude, C.S., Păun, Gh., Rozenberg, G., Salomaa, A. (eds.) *Multiset Processing. Mathematical, Computer Science, and Molecular Computing Points of View. Lecture Notes in Computer Science*, **2235**, Springer, 2001, 299–346.
17. Maliţa, M. Membrane computing in Prolog. In: *Pre-Proceedings of the Workshop on Multiset Processing*, Curtea de Arges, Romania, TR 140, CDMTCS, University of Auckland, (2000), 159–175.
18. Balbontín-Noval, D., Pérez-Jiménez, M.J., Sancho-Caparrini, F. A MzScheme implementation of transition P systems. In: Păun, G., Rozenberg, G., Salomaa, A., Zandron, C. (eds.) *Lecture Notes in Computer Science*, Springer Heidelberg, **2597** (2003), 58–73.
19. Pérez-Jiménez, M.J., Sancho-Caparrini, F. A formalization of transition P systems. *Fundamenta Informaticae*, **49** (2002), 261–272.
20. Arroyo, F., Luengo, C., Baranda, A.V., Mingo, L. A software simulation of transition P systems in Haskell. In: Păun, G., Rozenberg, G., Salomaa, A., Zandron, C. (eds.), *Lecture Notes in Computer Science*, Springer Heidelberg, **2597** (2003), 19–32.
21. Baranda, A.V., Castellanos, J., Arroyo, F., Gonzalo, R. Data structures for implementing transition P systems in silico. In: *Pre-Proceedings of the Workshop on*

Multiset Processing, Curtea de Arges, Romania, TR 140, CDMTCS, University of Auckland, (2000), 21–34.

22. Arroyo, F., Baranda, A.V., Castellanos, J., Luengo, C., de Mingo, L.F. A Recursive Algorithm for Describing Evolution in Transition P Systems. In *Pre-Proceedings of Workshop on Membrane Computing*, Curtea de Arges, Romania, Technical report 17/01 of Research Group on Mathematical Linguistics, Rovira i Virgili University, Tarragona, Spain (2001), 19–30.
23. Arroyo, F., Baranda, A.V., Castellanos, J., Luengo, C., de Mingo, L.F. Structures and Bio-Language to Simulate Transition P Systems on Digital Computers. In C.S. Calude, Gh. Păun, G. Rozenberg, A. Salomaa, (eds.) Multiset Processing. Mathematical, Computer Science and Molecular Computing Points of View, *Lecture Notes in Computer Science*, **2235**, 1–16.
24. Baranda, A.V., Castellanos, J., Gonzalo, R., Arroyo, F., de Mingo, L.F. Data Structures for Implementing Transition P Systems in Silico. *Romanian Journal of Information Science and Technology*, **4**, 1-2 (2001), 21–32.
25. Baranda, A.V., Castellanos, J., Arroyo, F., Gonzalo, R. Towards an Electronic Implementation of Membrane Computing: A Formal Description of Nondeterministic Evolution in Transition P Systems. In Jonoska, N., Seeman, N.C. (eds.) *Proceedings of DNA-Based Computers, Tampa, Florida, LNCS*, **2340** (2002), 350–359.
26. Nepomuceno-Chamorro, I.A. A Java Simulator for Basic Transition P Systems. In Păun, Gh., Riscos-Núñez, A., Romero-Jiménez, A., Sancho-Caparrini, F. (eds.) *Proceedings of the Second Brainstorming Week on Membrane Computing, Sevilla, Spain*, Report RGNC 01/04, 2004, 309–315.
27. Ciobanu, G., Wenyuan, G. A Parallel Implementation of Transition P Systems. In Alhazov, A., Martín-Vide, C. Păun, Gh. (eds.) *Pre-Proceedings of the Workshop on Membrane Computing, Tarragona, Spain, 2003*, Report RGML 28/03, 169–184.
28. Ciobanu, G., Wenyuan, G. P Systems Running on a Cluster of Computers. In Martín-Vide, C., Păun, Gh., Rozenberg, G., Salomaa, A. (eds.) *Lecture Notes in Computer Science*, **2933** (2004), 123–139.
29. Syropoulos, A., Mamatas, E.G., Allilomes, P.C., Sotiriades, K.T. A Distributed Simulation of Transition P Systems. In Martín-Vide, C., Păun, Gh., Rozenberg, G., Salomaa, A. (eds.) *Lecture Notes in Computer Science*, **2933** (2004), 357–368.
30. Ciobanu, G., Paraschiv, D. P System Software Simulator. *Fundamenta Informaticae*, **49**, 1-3 (2002), 61–66.
31. Pérez-Jiménez, M.J., Romero-Campero, F. A CLIPS Simulator for Recognizer P Systems with Active Membranes. In Păun, Gh., Riscos-Núñez, A., Romero-Jiménez, A., Sancho-Caparrini, F. (eds.) *Proceedings of the Second Brainstorming Week on Membrane Computing, Sevilla, Spain*, Report RGNC 01/04, 2004, 387–413.
32. Pérez-Jiménez, M.J., Romero-Campero, F.J. An efficient family of P systems for packing items into bins. *Journal of Universal Computer Science*, **10**, 5 (2004), 650–670.
33. Pérez-Jiménez, M.J., Romero-Campero, F.J. Attacking the Common Algorithmic Problem by recognizer P systems. *Lecture Notes in Computer Science*, **3354** (2005), 304–315.
34. Cordón-Franco, A., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Sancho-Caparrini, F. A Prolog Simulator for Deterministic P Systems with Active Membranes. *New Generation Computing*, **22**, 4 (2004), 349–364.
35. Cordón-Franco, A., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A., Sancho-Caparrini, F. Implementing in Prolog an Effective Cellular Solution to

the Knapsack Problem. In Martín-Vide, C., Păun, Gh., Rozenberg, G., Salomaa, A. (eds.) *Lecture Notes in Computer Science*, **2933** (2004), 140–152.

36. Cordón-Franco, A., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J. Riscos-Núñez, A., Sancho-Caparrini, F. Cellular Solutions of Some Numerical **NP**-Complete Problems: A Prolog Implementation. In Gheorghe, M. (ed.)*Molecular Computational Models: Unconventional Approaches*, Idea Group, Inc., 2005, 115–149.
37. Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez. A fast P system for finding balanced 2-partition. *Soft Computing*, **9** (2005), 673–678.
38. Ciobanu, G., Păun, Gh., Ştefănescu, Gh. Sevilla Carpets Associated with P Systems. In Cavaliere, M., Martín-Vide, C., Păun, Gh. (eds.) *Proceedings of the Brainstorming Week on Membrane Computing, Tarragona, Spain, 2003*, Report RGML 26/03, 135–140.
39. Riscos-Núñez, A. *Cellular Programming: Efficient Resolution of Numerical* **NP**-*complete Problems*. PhD Thesis, University of Seville, 2004.
40. Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A. On descriptive complexity of P systems. *Lecture Notes in Computer Science*, **3365** (2005), 320–330.
41. Binder, A., Freund, R., Lojka, G., Oswald, M. Implementation of Catalytic P Systems. *Proceedings of CIAA 2004, Ninth International Conference on Implementation and Application of Automata, Kingston, Canada, 2004*, 24–33.
42. Alhazov, A. Maximally Parallel Multiset-Rewriting Systems: Browsing the Configurations. *Proceeding of the Third Brainstorming Week on Membrane Computing, Sevilla, 2005*, RGNC Report 01/2005, 1–10.
43. Margenstern, M., Rogozhin, V., Rogozhin, Yu, Verlan, S. About P Systems with Minimal Symport/Antiport Rules and Four Membranes. In G. Mauri, Gh. Paun, C. Zandron, eds.: *Pre-Proceedings of the Workshop on Membrane Computing WMC5, Universitá di Milano-Bicocca, Italy, 2004*, 283–294.
44. Alhazov, A, Margenstern, M, Rogozhin, V., Rogozhin, Yu., Verlan, S: Communicative P systems with Minimal Cooperation. *Lecture Notes in Computer Science*, **3365** (2005), 161–177.
45. Georgiou, A. Sub-LP Systems A Computational Model for Plant Simulation. MSc Dissertation, University of Sheffield, 2003.
46. Georgiou, A., Gheorghe, M. Generative devices used in graphics. In Alhazov, A. *et al* (eds.) *Pre-proceedings of the Workshop on Membrane Computing Technical Report 28/03, Universitat Rovira i Virgili, Tarragona*, 2003, 266–272.
47. Georgiou, A., Gheorghe, M., Bernardini, F. Membrane-based devices used in computer graphics. In Ciobanu, G., Păun, Gh., Pérez-Jiménez, M.J. *Applications of Membrane Computing*. Springer, 2006, 253–282.
48. Nicolau Jr., D.V., Solana, G., Fulga, F., Nicolau Sr., D.V. A C Library for Simulating P Systems. *Fundamenta Informaticae*, **49**, 1-3 (2002), 241–248.
49. Petreska, B., Teuscher, C. A Reconfigurable Hardware Membrane System. In Martín-Vide, C., Păun, Gh., Rozenberg, G., Salomaa, A. (eds.) *Lecture Notes in Computer Science*, **2933** (2004), 269–285.
50. Madhu, M., Murty, V.S., Krithivasan, K. A Hardware Realization of P Systems with Carriers. Poster presentation at the *Eight International Conference on DNA based Computers*, Hokkaido University, Sapporo Campus, Japan, June 10-13, 2002.
51. Ardelean, I.I., Cavaliere, M. Modelling Biological Processes by Using a Probabilistic P System Software. *Natural Computing*, **2**, 2 (2003), 173–197.

52. Cavaliere, M. Evolution-Communication P Systems. In: Păun, G., Rozenberg, G., Salomaa, A., Zandron, C. (eds.) *Lecture Notes in Computer Science*, Springer Heidelberg, **2597** (2003), 13–145.
53. Cavaliere, M., Ardelean, I.I. Modeling Respiration in Bacteria and Respiration/Photosynthesis Interaction in Cyanobacteria Using a P System Simulator. In Ciobanu, G., Păun, G., Pérez-Jiménez, M.J. (eds.) *Applications of Membrane Computing*, Springer, Heidelberg (2006), 129–158.
54. Bianco, L., Castellini, A. Psim: A computational platform for metabolic P systems. *Lecture Notes in Computer Science*, **4860** (2007), 1–20.
55. Bianco, L., Manca, V., Marchetti, L., Petterlini, M. Psim: a simulator for biomolecular dynamics based on P systems. In: *2007 IEEE Congress on Evolutionary Computation*, IEEE XPlore, (2008), 883-887.
56. Bianco, L., Fontana, F., Franco, G., Manca, V. P Systems for Biological Dynamics. In Ciobanu, G., Păun, G., Pérez-Jiménez, M.J. (eds.) *Applications of Membrane Computing*, Springer, Heidelberg (2006), 83–128.
57. Nepomuceno, I., Nepomuceno, J.A., Romero-Campero, F.J., Gutiérrez-Naranjo, M. A. A tool for using the SBML format to represent P systems which model biological reaction networks. In Riscos-Nñez, A., Romero-Campero, F.J., Sburlan, D. (eds.) *Third Brainstorming Week on Membrane Computing*, Fénix Editora, 2005, 219–228.
58. Pérez-Jiménez, M.J., Romero-Campero, F.J. A study of the robustness of the EGFR signalling cascade using continuous membrane systems. *Lecture Notes in Computer Science*, **3561** (2005), 268–278.
59. Cheruku, S, Păun, A., Romero-Campero, F.J., Pérez-Jiménez, M.J., Ibarra, O.H. Simulating FAS-induced apoptosis by using P systems. *Progress in Natural Science*, **17** (2007), 424–431.
60. Romero-Campero, F.J., Pérez-Jiménez, M.J. Modelling gene expression control using P systems: The Lac operon, a case study. *Biosystems*, **91** (2008), 438–457.
61. Romero-Campero, F.J., Pérez-Jiménez, M.J. A model of the quorum sensing system in Vibrio fischeri using P systems. *Artificial Life*, **14** (2008), 95–109.
62. Cazzaniga, P., Pescini, D., Besozzi, D., Mauri, G. Tau Leaping Stochastic Simulation Method in P Systems. In Hoogeboom, H., Păun, Gh., Rozenberg, G. (eds.) *Membrane Computing, WMC7, Lecture Notes in Computer Science*, **4361** (2006), 298–313.
63. Cazzaniga, P., Pescini, D., Romero-Campero, F.J., Besozzi, D., Mauri, G. Stochastic approaches in P systems for simulating biological systems. In Gutiérrez-Naranjo, M.A., Păun, Gh., Riscos-Núñez, A., Romero-Campero, F.J. (eds.), *Proceedings of the Fourth Brainstorming Week on Membrane Computing*, RGNC REPORT 02/2006, Fénix Editora, 2006, 145–164.
64. Pescini, D., Besozzi, D., Mauri, G. Investigating local evolutions in dynamical probabilistic P systems. *Proceedings of Seventh International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC05)*, IEEE Computer Press, 2005, 440–447.
65. Pescini, D., Besozzi, D., Mauri, G., Zandron, C. Analysis and simulation of dynamics in probabilistic P systems. In A. Carbone, N. Pierce, Eds., DNA Computing, 11th International Workshop on DNA Computing, DNA11, London, ON, Canada, June 6-9, 2005. LNCS 3892, 236247, SpringerVerlag, 2006.
66. Pescini, D., Besozzi, D., Mauri, G., Zandron, C. Dynamical probabilistic P systems. *International Journal of Foundations of Computer Science*, **17** (2006), 183–204.

67. Sedwards, S., Mazza, T. CytoSym: A formal language model and stochastic simulator of membrane-enclosed biochemical processes. *Bioinformatics*, **23**, 20 (2007), 2800–2802.
68. Cavaliere, M., Sedwards, S. Modelling Cellular Processes Using Membrane Systems with Peripheral and Integral Proteins, *Lecture Notes in Computer Science*, **4210** (2006), 108–126.
69. Nishida, T.Y. Membrane algorithms. *Lecture Notes in Computer Science*, **3850** (2006), 55–66.
70. Nishida, T.Y. An application of P-system: A new algorithm for **NP**-complete optimization problems. In Callaos, N. *et al.* (eds.) *Proceedings of The 8th World Multi-Conference on Systems, Cybernetics and Informatics*, **V** (2004), 109–112.
71. Nishida, N.Y. An approximate algorithm for **NP**-complete optimization problems exploiting P-systems. In *Proceedings of Brainstorming Workshop on Uncertainty in Membrane Computing, Palma de Mallorca*, 2004, 185–192.
72. Nishida, N.Y. Membrane algorithms. Approximate algorithms for **NP**-complete optimization problems. In Ciobanu, G., Păun, Gh., Pérez-Jiménez, M.J. (eds.) *Application of Membrane Computing*, Springer, Berlin, 2005, 301–312.
73. Borrego-Ropero, R., Díaz-Pernil, D., Pérez-Jiménez, M.J. Tissue simulator: A graphical tool for tissue P systems. In Vaszil, Gy (ed.) *Proceedings of the International Workshop of Automata for Cellular and Molecular Computing, MTA SZTAKI, Budapest, Hungary*, 2007, 23–34.
74. Martín-Vide, C., Păun, Gh., Pazos, J., Rodríguez-Patón, A.: Tissue P systems, *Theoretical Computer Science*, 296(2), 2003, 295326.
75. Díaz-Pernil, D., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Riscos-Núñez, A. A uniform family of tissue P system with cell division solving 3-COL in a linear time. *Theoretical Computer Science*, **404** (2008), 76–87.
76. Ionescu, M., Păun, Gh., Yokomori, T. Spiking neural P systems. *Fundamenta Informaticae*, **71** (2006), 279–308.
77. Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J., Ramírez-Martínez, D. A software tool for verification of spiking neural P systems. *Natural Computing*, **7** (2008), 485–497.
78. Frisco, P., Gibson, R.T. A simulator and an evolution program for conformon-P systems. In *TAPS, Workshop on Theory and Applications of P Systems, Timişoara, Romania, IEEE Computer Press*, 2005, 427–430.
79. Frisco, P., Ji, S. Conformons-P systems. *Lecture Notes in Computer Science*, 2568 (2003), 291–301.
80. Frisco, P. The Conformon-P System: A Molecular and Cell Biology-Inspired Computability Model. *Theoretical Computer Science*, **312** (2004), 295–319.
81. Corne, D.W., Frisco, P. Dynamics of HIV infection studied with cellular automata and conformon-P systems. *BioSystems*, **91**, 3 (2008), 531–544.
82. Romero-Jiménez, A., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J. Graphical modelling of higher plants using P systems. *Lecture Notes in Computer Science*, **4361** (2006), 496–506.
83. Romero-Jiménez, A., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J. The growth of branching structures with P systems. In Graciani, C. *et al.* (eds.) *Fourth Brainstorming Week on Membrane Computing, Sevilla, Vol. II*, Fénix Editora, 2006, 253–265.
84. Rivero-Gil, E., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J. Graphics and P systems: Experiments with JPLANT. In Díaz-Pernil, D., Graciani, C., Gutiérrez-

Naranjo, M.A., Păun, Gh., Pérez-Hurtado, I., Riscos-Núñez, A. (eds.) *Sixth Brainstorming Week on Membrane Computing, Fénix Editora, Sevilla*, 2008, 241–254.
85. Rivero-Gil, E., Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J. Romero-Jiménez, A., Riscos-Núñez, A. A software tool for generating graphics by means of P systems. *Natural Computing*, Springer Netherlands, **10**, 2 (2011), 879–890.
86. Acampora, G., Loia, V. A proposal of multi-agent simulation system for membrane computing devices. In: *2007 IEEE Congress on Evolutionary Computation*, IEEE XPlore, (2008), 4100-4107.
87. Díaz-Pernil, D., Pérez-Hurtado, I., Pérez-Jiménez, M.J., Riscos-Núñez, A. A P-Lingua programming environment for membrane computing. *Lecture Notes in Computer Science*, 5391 (2009), 187-203.
88. García-Quismondo, M., Gutiérrez-Escudero, R., Pérez-Hurtado, I., Pérez-Jiménez, M.J., Riscos-Núñez, A. An overview of P-Lingua 2.0. *Lecture Notes in Computer Science*, 5957 (2010), 264-288.
89. García-Quismondo, M., Gutiérrez-Escudero, R., Pérez-Hurtado, I., Pérez-Jiménez, M.J., Riscos-Núñez, A. An overview of P-Lingua 2.0. *Lecture Notes in Computer Science*, 5957 (2010), 264-288.
90. Gershoni, R., Keinan, E., Păun, Gh., Piran, R., Ratner, T., Shoshani, S. Research topics arising from the (planned) P systems. *6th Brainstorming Week on Membrane Computing, Fénix Editora*, 2008, 183-192.
91. Keinan, E. Membrane computing. *Google Patents*, 2009.
92. Pérez-Hurtado, I., Valencia-Cabrera, L., Pérez-Jiménez, M.J., Colomer, M.A., Riscos-Núñez, A. Mecosim: A general purpose software tool for simulating biological phenomena by means of p systems. *IEEE Fifth International Conference on Bio-inpired Computing: Theories and Applications (BIC-TA 2010)*, I (2010), 637–643.
93. Romero-Campero, F.J., Twycross, J., Cámara, M., Bennett, M., Gheorghe, M., Krasnogor, N. Modular Assembly of Cell Systems Biology Models Using P systems. *International Journal of Foundations of Computer Science*, **20**, 3 (2009), 427-442.
94. Colomer, M.A., Margalida, A., Sanuy, D., Pérez-Jiménez, M.J. A bio-inspired computing model as a new tool for modeling ecosystems: The avian scavengers as a case study. *Ecological Modelling*, Elsevier, **222**, 1 (2011), 33–47.
95. Colomer, M.A., Margalida, A., Valencia, L., Palau, A. Application of a computational model for complex fluvial ecosystems: The population dynamics of zebra mussel Dreissena polymorpha as a case study. *Ecological Complexity*, **20** (2014), 116–126.
96. Fondevilla, C., Colomer, M.A., Fillat, F., Tappeiner, U. Using a new PDP modelling approach for land-use and land-cover change predictions: A case study in the Stubai Valley (Central Alps). *Ecological Modelling*, **322** (2016), 101–114.
97. Valencia-Cabrera, L., García-Quismondo, M., Pérez-Jiménez, M.J., Su, Y., Yu, H., Pan, L. Modeling Logic Gene Networks by Means of Probabilistic Dynamic P Systems. *International Journal of Unconventional Computing*, Old City Publishing Inc., **9** (2013), 445–464.
98. Lefticaru, R., Ipate, F., Valencia-Cabrera, L., Ţurcanu, A., Tudose, C., Gheorghe, M., Pérez-Jiménez, M.J., Niculescu, I.M., Dragomir, C. Towards an integrated approach for model simulation, property extraction and verification of P systems. *Proceedings of 10th Brainstorming Week on Membrane Computing*, Fénix Editora, **I** (2012), 291–318.

99. Gheorghe, M., Ipate, F., Lefticaru, R., Pérez-Jiménez, M.J., Ţurcanu, A., Valencia-Cabrera, L., García-Quismondo, M., Mierla, L. 3-COL problem modelling using simple Kernel P systems. *International Journal of Computer Mathematics*, Taylor & Francis, **90** (2013), 816–830.
100. Colomer, M. A., Margalida, A., Pérez-Jiménez, M.J. Population Dynamics P System (PDP) Models: A Standardized Protocol for Describing and Applying Novel Bio-Inspired Computing Tools. *PLoS ONE*, **4** (2013).
101. Lérida, J.L., Agraz, A., Solsona, F., Colomer, M.A. PSysCal: a parallel tool for calibration of ecosystem models. *Cluster Computing*, **17**, 2 (2014), 271–279.
102. Blakes, J., Twycross, J., Romero-Campero, F. J., Krasnogor, N. The Infobiotics Workbench: an integrated in silico modelling platform for Systems and Synthetic Biology. *Bioinformatics, Oxford*, **27**, 23 (2011), 3323-3324.
103. Buiu, C., Arsene, O., Cipu, C., Patrascu, M. A software tool for modeling and simulation of numerical P systems. *BioSystems*, **103**, (2011), 442–447.
104. Arsene, O., Buiu, C., Popescu, N. SNUPS - a simulator for numerical membrane computing. *International Journal of Innovative Computing, Information and Control*, **7**, (2011), 3509–3522.
105. Pavel, A. B., Vasile, C. I., Dumitrache, I. Robot Localization Implemented with Enzymatic Numerical P Systems. In Prescott, T.J., Lepora, N.F., Mura, A., Verschure, P.F.M.J. (Ed.) *Biomimetic and Biohybrid Systems: Proceedings of the First International Conference, Living Machines, Barcelona*, (2012), 204–215.
106. García-Quismondo, M., Macías-Ramos, L. F., Pérez-Jiménez, M. J. Implementing Enzymatic Numerical P Systems for AI Applications by means of Graphic Processing Units. *Beyond Artificial Intelligence: Contemplations, Expectations, Applications*, Springer Verlag, **4**, (2013), 137–157.
107. García-Quismondo, M., Martínez-del-Amor, M.A., Pérez-Jiménez, M.J. Probabilistic Guarded P Systems, A New Formal Modelling Framework. In: Gheorghe, M., Rozenberg, G., Salomaa, A., Sosík, P., Zandron, C. (Eds.) Membrane Computing. CMC 2014. *Lecture Notes in Computer Science*, 8961 (2014), 194–214.
108. García-Quismondo, M., Reed, J.M., Chew, F.S., Martínez-del-Amor, M.A. Pérez-Jiménez, M.J. Evolutionary response of a native butterfly to concurrent plant invasions: Simulation of population dynamics. *Ecological Modelling*, **360** (2017), 410–424.
109. García-Quismondo, M., Levin, M., Lobo, D. Modeling regenerative processes with membrane computing. *Information Sciences*, **381** (2017), 229–249.
110. Campos, M., Llorens, C., Sempere, J.M., Futami, R., Rodriguez, I., Carrasco, P., Capilla, R., Latorre, A., Coque, T.M., Moya, A., Baquero, F. A membrane computing simulator of trans-hierarchical antibiotic resistance evolution dynamics in nested ecological compartments (ARES). *Biology Direct*, **10**, 1 (2015), 41.
111. Shaalan, B., Muniyandi, R.C. Implementing Mitogen Activated Protein Kinases Cascade on Membrane Computing Using P-Lingua. *Journal of Computer Science*, **11**, 1 (2015), 178–187.
112. Li, J., Huang, Y., Xu, J. Decoder Design Based on Spiking Neural P Systems. *IEEE Transactions on NanoBioscience*, **15**, 7 (2016), 639–644.
113. Huang, Y., Li, J., Xu, J. Microfluidic Half Adder Chip Based on Spiking Neural P Systems Technical Journal of the Faculty of Engineering, *Revista Tecnica de la Facultad de Ingenieria Universidad del Zulia*, **39**, 9 (2016), 317–323.
114. Liu, X., Xue, J. A Cluster Splitting Technique by Hopfield Networks and P Systems on Simplices. *Neural Processing Letters*, **46**, 1 (2017), 171–194.

115. Giannakis, K., Andronikos, T. Membrane automata for modeling biomolecular processes. *Natural Computing*, **16**, 1 (2017), 151–163.
116. Lefticaru R., Macías-Ramos L.F., Niculescu I.M., Mierlă, L. Agent-Based Simulation of Kernel P Systems with Division Rules Using FLAME. In: Leporati A., Rozenberg G., Salomaa A., Zandron C. (eds) *Membrane Computing. CMC 2016. Lecture Notes in Computer Science*, Springer, **10105** (2017), 286–306.
117. Gheorghe, M., Ipate, F., Mierlă, L., Konur, S. Stochastic approaches in P systems for simulating biological systems. *Proceedings of the Thirteenth Brainstorming Week on Membrane Computing*, Fénix Editora, 2015, 179–194.
118. Cecilia, J.M., García, J.M., Guerrero, G.D., Martínez-del Amor, M.A., Pérez-Hurtado, I., Pérez-Jiménez, M.J. Simulating a P system based efficient solution to SAT by using GPUs. *Journal of Logic and Algebraic Programming.* **79**, 6 (2010), 317–325.
119. Cecilia, J.M., García, J.M., Guerrero, G.D., Martínez-del Amor, M.A., Pérez-Hurtado, I., Pérez-Jiménez, M.J. Simulation of P systems with active membranes on CUDA. *Briefings in Bioinformatics* **11**, 3 (2010), 313–322.
120. Cabarle, F.G.C., Adorna, H., Martínez-del-Amor, M.A. An improved GPU simulator for spiking neural P systems. In: *Sixth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA)*, 2011, 262–267.
121. Cabarle, F.G.C., Adorna, H., Martínez-del-Amor, M.A. A spiking neural P system simulator based on CUDA. In: Gheorghe, M., Păun, Gh., Rozenberg, G., Salomaa, A., Verlan, S. (eds.), *CMC 2011, Lecture Notes in Computer Science*, **7184** (2012), 87–103.
122. Martínez-del-Amor, M.A., Karlin, I., Jensen, R.E., Pérez-Jiménez, M.J., Elster, A.C. Parallel simulation of probabilistic P systems on multicore platforms. *Proceedings of the Tenth Brainstorming Week on Membrane Computing*, **II** (2012), 17–26.
123. Bangalan, Z.F., Soriano, K.A.N., Juayong, R.A.B., Cabarle, F.G.C., Adorna, H.N., Martínez-del-Amor, M.A. A GPU Simulation for Evolution-Communication P Systems with Energy Having no Antiport Rules, *Proceedings of the Eleventh Brainstorming Week on Membrane Computing* (2013), 25–50.
124. Maroosi, A., Muniyandi, R.C. Accelerated simulation of membrane computing to solve the N-queens problem on multi-core. In: Panigrahi B.K., Suganthan P.N., Das S., Dash S.S. (eds) *Swarm, Evolutionary, and Memetic Computing. SEMCCO 2013. Lecture Notes in Computer Science*, Springer, **8298** (2013).
125. Maroosi, A., Muniyandi, R.C., Sundararajan, E.A., Zin, A.M. Improved implementation of simulation for membrane computing on the graphic processing unit. *Procedia Technology*, **11** (2013), 184–190.
126. Martínez-del-Amor, M.A., Carrasco, J.P., Pérez-Jiménez, M.J. Simulating a Family of Tissue P Systems Solving SAT on the GPU, *Eleventh Brainstorming Week on Membrane Computing (11BWMC)*, Fnix Editora, 2013, 201–220.
127. Martínez-del-Amor, M.A., Macías-Ramos, L.F., Valencia-Cabrera, L., Riscos-Núñez, A., Pérez-Jiménez, M.J. Accelerated simulation of P systems on the GPU: a survey. In: Pan, L., Păun, Gh., Pérez-Jiménez, M.J., Song, T. (eds.) *Bio-Inspired Computing - Theories and Applications. Communications in Computer and Information Science*, Springer, **472** (2014), 308–312.
128. Cabarle, F., Adorna, H., Martinez-del-Amor, M.A. Simulating Spiking Neural P systems without delays using GPUs. In Nunes de Castro, L. (Ed.)*Natural Computing for Simulation and Knowledge Discovery*, IGI Global, (2014), 109–121.

129. Carandang, J.P.A., Villaflores, J.M.B., Cabarle, F.G.C., Adorna, H.N., Martínez-del-Amor, M.A. CuSNP: Spiking Neural P Systems Simulators in CUDA. *Romanian Journal of Information Science and Technology*, **20**, 1 (2017), 57–70.
130. Martínez-del-Amor, M.A., Pérez-Hurtado, I., Pérez-Jiménez, M.J., Riscos-Núñez, A. A P-lingua based simulator for tissue P systems. *The Journal of Logic and Algebraic Programming*, **79**, 6 (2010), 374–382.
131. Macías-Ramos, L.F., Pérez-Hurtado, I., García-Quismondo, M., Valencia-Cabrera, L., Pérez-Jiménez, M.J., Riscos-Núñez, A. A P-lingua based simulator for spiking neural P systems. In: Gheorghe, M., Păun, Gh., Rozenberg, G., Salomaa, A., Verlan, S. (eds.) *CMC 2011, Lecture Notes on Computer Science*, Springer, **7184** (2012), 257–281.
132. Pérez-Hurtado, I., Valencia-Cabrera, L., Chacon, J.M., Riscos-Núñez, A., Pérez-Jiménez, M.J.: A P-lingua based simulator for tissue P systems with cell separation. *Romanian Journal of Information Science and Technology*, **17**, 1 (2014), 89–102.
133. Macías-Ramos, L.F., Valencia-Cabrera, L., Song, B., Song, T., Pan, L., Pérez-Jiménez, M.J. A P-lingua based simulator for P systems with symport/antiport rules. *Fundamenta Informaticae*, **139**, 2 (2015), 211–227.
134. Valencia-Cabrera, L., Wu, T., Zhang, Z., Pan, L., Pérez-Jiménez, M.J. A simulation software tool for cell-like spiking neural P systems. *Romanian Journal of Information Science and Technology*, **20**, 1 (2017), 71–84.
135. Florea, A.G., Buiu, C. Development of a software simulator for P colonies - Applications in robotics. *International Journal on Unconventional Computing*, **12**, 2–3 (2016), 189-205.
136. Florea, A.G., Buiu, C. Synchronized dispersion of robotic swarms using XP colonies. *Proceedings of the 8th International Conference on Electronics, Computers and Artificial Intelligence*, ECAI 2016, 2017, 1–6.

Web links

137. P systems web page (Ppage). `http://ppage.psystems.eu`
138. P systems Software (in Ppage). `http://ppage.psystems.eu/Software`
139. P systems simulation timeline. `https://www.gcn.us.es/SimulationMC`
140. Research Group on Natural Computing - University of Seville: `http://www.gcn.us.es`
141. Center for BioMedical Computing - Verona: `http://www.cbmc.it/`
142. Meta PLab site: `http://mplab.scienze.univr.it/index.html`
143. P System Modelling Framework at the University of Sheffield: `http://staffwww.dcs.shef.ac.uk/people/M.Gheorghe/PSimulatorWeb/P_Systems_applications.htm`
144. Natural Computing Group - Polytechnic University of Madrid: `http://www.gcn.upm.es/`
145. Cyto-sim site: `https://sites.google.com/site/cytosim/home`
146. The Xholon Project: `http://www.primordion.com/Xholon`
147. P-Lingua website. `http://www.p-lingua.org/`
148. MeCoSim website. `http://www.p-lingua.org/mecosim/`
149. Infobiotics website. `http://infobiotics.org/`
150. MeCoGUI website. `http://www.p-lingua.org/wiki/index.php/MeCoGUI`

151. García-Quismondo, M. A Java-Based P-Lingua Simulator for Enzymatic Numerical P Systems `http://www.cs.us.es/blogs/mgarcia/research/software_tools/java_simulator_enps/`
152. Pieris oleracea model website. `http://www.p-lingua.org/wiki/index.php/PGSP_systems:_Pieris_oleracea`
153. kPWorkbench website. `http://kpworkbench.org/`
154. PMCGPU project. `http://www.p-lingua.org/wiki/index.php/PMCGPU`
155. Florea, A.G., Buiu, C. Lulu - a software simulator for P colonies. Use case scenarios and demonstration videos. Zenodo. `http://doi.org/10.5281/zenodo.34350` (2015).
156. Florea, A.G., Buiu, C. Lulu - an open-source software simulator of P colonies and P swarms. `https://github.com/andrei91ro/lulu_pcol_sim` (2016).
157. Timeline in RGNC. `http://www.gcn.us.es/SimulationMC`