Membrane computing is a branch of natural computing which 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., of bacteria).

Membrane computing was initiated in 1998 Păun Gh. (2000) and the literature of this area has grown very fast (already in 2003, Thompson Institute for Scientific Information, ISI, has qualified Păun Gh. (2000) as “fast breaking” and the domain as “emergent research front in computer science”). A comprehensive presentation at the level of year 2002 can be found in Păun Gh. (2002) and a recent coverage of the domain can be found in Păun Gh. et.al. (2010) and Frisco P. (2009). The most up-to-date source of information for membrane computing is the P systems webpage (where many papers on the topic can be downloaded).

The initial goal of membrane computing was to learn from cell biology something possibly useful to computer science, and membrane computing 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. Then, a number of applications were reported 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.

The present text is only a quick introduction to membrane computing, only mentioning some basic ideas, types of results and of applications, and indicating some references and links.

A Quick Description of Membrane Computing

The main ingredients of a P system are the membrane structure, delimiting compartments where multisets of objects evolve according to (reaction) rules of a bio-chemical inspiration.

If Figure 1 a P system is depicted, where:
- the membrane structure consists of 5 membranes;
- each membrane defines a compartment (region);
- the external compartment is called skin;
- the area where the P system is places id called environment (yellow in the figure);
- compartments can contain other compartments, objects and rules;
- compartment (or membranes) which do not contain other compartments are called elementary;
- in the figure objects are denoted by capital letters, rules are framed in purple and each compartment has a unique number as identified.

A membrane can contain other membranes, and 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

- **computing efficiency**, the possibility of using parallelism for solving computationally hard problems in a feasible time.

Computationally and mathematically motivated ways of using the rules and of defining the result of a computation are considered in this second case. For instance, the main way of evolving a P system is based on a non-deterministic **maximally parallel use** of rules, with the system being **synchronized** (the clock is universal and in each time unit one uses a maximal multiset of rules in each membrane). Many variants are possible which are not mentioned here (the ones present in Scholarpedia are: catalytic P systems, metabolic P systems, P automata and P systems with symport/antiport. When a P system is constructed as a model of a bio-chemical process, it is examined in terms of dynamical systems, with the (deterministic or probabilistic) evolution in time being the issue of interest, rather than a specific output.

Three main types of P systems have been considered until now:

(i) cell-like P systems,

(ii) tissue-like P systems,

(iii) neural-like P systems.

The first type imitates the (eukaryotic) cell, and its basic ingredient is the **membrane structure**, which is a hierarchical arrangement of membranes (understood as three dimensional vesicles), delimiting compartments where multisets of objects are placed; the objects are in general described by symbols from a given alphabet; 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, i.e., of the type \( \langle u \to v \rangle \) where \( \langle u \rangle \) and \( \langle v \rangle \) are multisets of objects) and transport rules, e.g., **P systems with symport/antiport** (Păun A., Păun, Gh. (2002)), inspired by biological processes (symport rules are of the form \( \langle \{u\},\{v\}\rangle \) or \( \langle \{u\},\{u\}\rangle \) moving objects of multiset \( \langle u \rangle \) through a membrane, and antiport rules are of the form \( \langle \{u\},\{v\}\rangle \) moving the objects of multiset \( \langle u \rangle \) outside a membrane at the same time with moving the objects of multiset \( \langle v \rangle \) inside). Also the objects produced by a transition rule can pass through membranes (we say that they are **communicated among compartments**), going from the current compartment to the external one (\( \text{out} \)) or to one of the internal ones (\( \text{in} \)), if any. The rules can have several other 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, like in Cardelli L. (2005), and so on. Moreover, the objects can be of various types, not only described by letters from a given alphabet, as in the basic class of P systems. For instance, the objects can be described by strings, and then they evolve by string processing rules (for instance, rewriting, splicing, insertion-deletion), they can be pairs name-value, called **conformons** (Frisco P. (2009), or even more complex data structures (arrays).

In tissue-like P systems Martin-Vide C. (2003), 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** Bernardini F., Gheorghe M. (2004). 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 systems: 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 introduced in Ionescu M. (2006), under the name of **spiking neural P systems** (in short, **SN 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.

All these types of P systems can be used in the generative mode (starting from the initial configuration, one proceeds until reaching a halting configuration, when a result is provided - in general, as the number of objects in a given output membrane), or in the accepting mode (for instance, a number is introduced in a membrane, in the form of the multiplicity of a given object, and this number is accepted if the computation halts) - the latter case leads to the notion of a **P automaton**. Extensions to vectors of numbers and to strings and languages were also investigated.

### Power and Efficiency

From a theoretical point of view, P systems are both:

- **powerful** as most classes of P systems 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; also small universal P systems of various types were produced, and

- **efficient**, as 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 a feasible time - typically polynomial). Such a speed-up is obtained by trading space for time, with the space grown exponentially in a linear time by means of bio-inspired operations. The most investigated way to obtain such an exponential workspace is membrane division Păun Gh. (2001), but also membrane creation, membrane separation, string replication and other operations were used. These operations were then extended to cells in tissue-like P systems and even to neurons in SN P systems - and in all cases similar results were obtained: polynomial time solutions to computationally hard problems. Details can be found, e.g., in Perez-Jimenez M.J. (2009) and computational complexity in P systems.

### Applications
As a modeling framework, membrane systems is rather adequate for handling discrete (biological) processes, having many attractive features: easy understandability, scalability and programmability, inherent compartmentalization and ability to handle discrete data, etc. Most applications use cell-like P systems and tissue-like P systems, and the general protocol is the following: a P system is written which models a given process (biological - but not only: also economic processes, evolution of ecosystems, and of other processes were addressed in this framework), capturing the objects, compartments, and evolution rules; then a program is written to simulate this P system, or a program available on internet is used; after that, computer simulations are performed, tuning parameters until a correct description of the real process is obtained; related processes are then investigated within this framework. Details can be found in Ciobanu G. et.al. (2006), including case studies and a description of existing software products (at the level of 2005).

There are also applications of other types, e.g., in computer graphics, cryptography, approximate optimization, and so on. Comprehensive and up-dated information (at the level of year 2009) about all issues mentioned above, from the mathematical theory of P systems, power and efficiency included, for various different classes of P systems, to applications, available software and implementations, can be found in Păun Gh. et.al. (2010) (chapters are dedicated to each significant issue) and, providing the state-of-the-art, in the P systems webpage. In particular, Chapter 1 of Păun Gh. et.al. (2010) is a friendly and comprehensive introduction to membrane computing.

References


Internal references


Sponsored by: Dr. Pierluigi Frisco, Heriot-Watt University, School of Mathematical and Computer Sciences, Edinburgh, UK
Reviewed by: Prof. Giancarlo Mauri, Dipartimento di Informatica, Sistemistica e Comunicazione. Universita' degli studi di Milano Bicocca
Reviewed by: Dr. Marian Gheorghe, Department of Computer Science, The University of Sheffield, UK
Accepted on: 2009-12-21 06:58:08 GMT

Category: Molecular Computing
Membrane computing deals with distributed and parallel computing models, processing multisets of symbol objects in a localized manner. Thus, evolution rules allow for evolving objects to be encapsulated into compartments defined by membranes. The intuition behind the notion of a membrane is a three-dimensional vesicle from biology. However, the concept itself is more general, and a membrane is seen as a separator of two regions. The membrane provides for selective communication between the two regions. The Journal of Membrane Computing (JMC) provides a forum for developing and nurturing an international community of scholars and practitioners who are interested in all aspects of membrane computing: theories, interdisciplinary areas, applications, and implementations. JMC welcomes high-quality submissions that contribute to the full range of membrane computing research, from cell-like P systems, tissue P systems, spiking neural P