Computational Complexity Aspects in Membrane Computing

Giancarlo Mauri, Alberto Leporati
Antonio E. Porreca, Claudio Zandron

Dipartimento di Informatica, Sistemistica e Comunicazione
Università degli Studi di Milano – Bicocca
Viale Sarca 336/14, 20126 Milano, Italy

{mauri,leporati,porreca,zandron}@disco.unimib.it

Abstract. Within the framework of membrane systems, distributed parallel computing models inspired by the functioning of the living cell, various computational complexity classes have been defined, which can be compared against the computational complexity classes defined for Turing machines. Here some issues and results concerning computational complexity of membrane systems are discussed. In particular, we focus our attention on the comparison among complexity classes for membrane systems with active membranes (where new membranes can be created by division of membranes which exist in the system in a given moment) and the classes **PSPACE**, **EXP**, and **EXPSPACE**.
Key words: Membrane systems, Computational complexity.

1 Membrane Systems

Living cells can be seen as very sophisticated devices for information processing, and computer science can benefit trying to understand the basic mechanisms they use and to abstract from them formal computation models. In this view, in [7] Gheorghe Păun introduced membrane systems (also called *P systems*), a class of distributed and parallel computing devices, inspired by the structure and functioning of living cells, and the way the cells are organized in tissues or higher order structures. More specifically, the feature of the living cells that is abstracted by membrane computing is their compartmentalized internal structure, where the cell membrane contains different organelles (compartments or regions), in turn delimited by membranes.

In the original definition, a membrane system consists of a hierarchical (tree-like) structure composed by several membranes, embedded into a main membrane called the *skin*, that separates the system (the cell) from the environment. Membranes divide the Euclidean space into *regions* that contain multisets of *objects* of different kind (representing molecules), denoted by symbols from an alphabet Γ, and region-specific *developmental rules* (bio–chemical reactions taking place in the different compartment of the cell). A *configuration* of the system is a mapping that associates with every membrane the multiset of objects it contains. The application of a rule modifies the content of the membrane were the

rule resides, and hence the configuration of the system, taking a multiset of objects contained in the membrane and producing a new multiset; furthermore, it specifies whether the resulting objects will stay into the current region, or will be transferred one level up (to the external region) or one level down (to one of the inner regions) in the hierarchy. The transfer thus provides communication between regions. More formally, a rule has the form $u \rightarrow v$, where u is a string over Γ, and v is a string over $\Gamma \times \{here, out, in\}$. Thus, each element of v is of the form (a, tar), where a is an object from Γ and tar (target) is either $here$ or out or in. The rule can be applied, i.e. is *enabled*, if and only if the multiset w of objects present in the membrane includes u; its application will subtract u from w, and will produce the objects indicated in v, that will be sent to the corresponding target. The rules are usually applied in a nondeterministic and maximally parallel manner: at each step, all the rules enabled in any membrane will be applied, modifying the system configuration; a nondeterministic choice is done in the case where two different rules have non disjoint enabling multisets. A computing device is obtained in this way: a *computation* starts from an initial configuration of the system and terminates when no further rules can be applied. Usually, the result of a computation is the multiset of objects contained in an *output membrane* or emitted from the skin of the system.

Many variants have been defined up to now, with mathematical, computer science or biological motivation (see, e.g., [9], [10]), including symport/antiport, catalytic, spiking neural, tissue, insertion–deletion, splicing, energy–based P-systems. Applications were reported especially in biology and medicine, but also in other directions, such as economics, approximate optimization, and computer graphics. A number of implementations are currently attempted. An overview of the state-of-the-art and an up-to-date bibliography concerning P systems can be found in [10].

2 Computational Complexity of Membrane Systems

From the very beginning, one of the central research topics in the theory of P systems has been to establish the computational power of the various proposed types of membrane systems, comparing them with conventional models such as finite state automata or Turing machines. Moreover, it has been and remains of interest to describe "universal" membrane systems with minimum number of membranes, or to describe the power of various classes of such systems depending on the number of membranes.

In particular, in [8] membrane systems are considered where membranes play an active role in the computation. In this model, the evolution rules involve both objects and membranes, and the communication of objects through a membrane is performed with the direct participation of the membrane itself; moreover, the membrane structure can be modified also by adding new membranes, which can be created by dividing existing membranes, through a process inspired from cellular *mitosis*. There are two different types of division rules: division rules for

elementary membranes (i.e. membranes not containing other membranes) and division rules for non-elementary membranes.

This variant with active membranes allows to produce an exponential number of membranes in a polynomial time. As a consequence, in [8] it was shown that such systems are able to efficiently solve **NP**-complete problems [6] (by trading time for space), and similar results are given in [3].

Starting from these results, various complexity classes for P systems were defined [11]. Such classes were then compared against the usual complexity classes such as **P** and **NP** (see, e.g., [12] and [2]); in [15], [1] it was shown that complexity classes defined in terms of two variants of P systems with active membranes contain all problems in **PSPACE**. In [13] it was shown that there exists a complexity class for P systems with active membranes which includes the class **PSPACE** and which is included in the class **EXP**; in [16] this result was strengthened by proving equality to **PSPACE**.

The previous results were given considering either deterministic systems or *confluent* systems, i.e. systems which can operate nondeterministic choices, but in such a way that the result (acceptance or rejection) of every computation is the same (thus, the simulation of a single computation is enough to determine the result of all computations).

In [17] it was shown that, unless $\mathbf{P} = \mathbf{NP}$, a confluent P system working without using membrane division cannot solve an **NP**-complete problem in polynomial time. In fact, the class of problems solved by such system, usually denoted by $\mathbf{PMC}_{\mathcal{NAM}}$, is equal to the standard complexity class **P**. In [14] non-confluent P systems without membrane division and operating in polynomial time are considered. First a solution for the decision problem SAT (satisfiability for Boolean formulas in conjunctive normal form) with a polynomially semi-uniform family of non-confluent P systems without membrane division is proposed, thus proving that $\mathbf{NP} \subseteq \mathbf{NPMC}_{\mathcal{NAM}}$. Then the reverse inclusion is proved, showing that non-confluent P systems without membrane division can be simulated in polynomial time by non-deterministic Turing machines. As a consequence, we have that $\mathbf{NPMC}_{\mathcal{NAM}}$ equals the complexity class **NP**.

3 Conclusion and Future Directions

The classes of P systems with active membranes we have considered are only defined according to which kinds of membrane division rules are available (none, just elementary or both elementary and non-elementary). The same questions may be also worth posing about other restricted classes, such as P systems without object evolution or communication [12, 4], P systems with division but without dissolution, or even purely communicating P systems, with or without polarizations. Finally, we think that the differences between P systems and traditional computing devices deserve to be investigated for their own sake also from the point of view of space-bounded computations.

References

1. Alhazov, A., Martín-Vide, C., Pan, L.: Solving a PSPACE-complete problem by P systems with restricted active membranes. Fundamenta Informaticae, 58 (2), 67–77 (2003)
2. Gutiérrez-Naranjo, M.A., Pérez-Jiménez, M.J.: P systems with active membranes, without polarizations and without dissolution: a characterization of P. In: Calude, C.S., Dinneen, M.J., Păun, Gh., Pérez-Jiménez, M.J., Rozenberg, G. (eds.) Unconventional Computation, 4th International Conference, UC 2005. LNCS, vol. 3699, pp. 105–116. Springer, Heidelberg (2005)
3. Krishna, S.N., Rama, R.: A variant of P systems with active membranes: Solving NP-complete problems. Romanian J. of Information Science and Technology, 2 (4), 357–367 (1999)
4. Leporati, A., Zandron, C., Ferretti, C., Mauri, G.: On the Computational Power of Spiking Neural P Systems. Intern. J. of Unconventional Computing, 5 (5), 459–473 (2009)
5. Leporati, A., Mauri, G., Zandron, C., Păun, Gh., Pérez-Jiménez, M.J.: Uniform Solutions to SAT and Subset Sum by Spiking Neural P Systems. Natural Computing 8 (4), 681–702 (2009)
6. Papadimitriou, C.H.: Computational Complexity. Addison-Wesley, Reading, MA (1993)
7. Păun, Gh.: Computing with membranes. J. of Computer and System Sciences, 61 (1), 108–143 (2000)
8. Păun, Gh.: P systems with active membranes: attacking NP complete problems. In: Antoniou, I., Calude, C.S., Dinneen, M.J. (eds.) Unconventional Models of Computation, pp. 94–115. Springer, London (2000)
9. Păun, Gh.: Membrane Computing. An Introduction. Springer, Berlin (2002)
10. Păun, Gh., Rozenberg, G., Salomaa, A. (eds): The Oxford Handbook of Membrane Computing. Oxford University Press (2009)
11. Pérez-Jiménez, M.J., Romero-Jiménez, A., Sancho-Caparrini, F.: Complexity Classes in Cellular Computing with Membranes. Natural Computing, 2 (3), 265–285 (2003)
12. Pérez-Jiménez, M.J., Romero-Jiménez, A., Sancho-Caparrini, F.: The P versus NP problem through cellular computing with membranes. In: Jonoska, N., Păun, Gh., Rozenberg, G. (eds.) Aspects of Molecular Computing. LNCS, vol. 2950, pp. 338–352. Springer, Heidelberg (2004)
13. Porreca, A. E., Mauri, G., Zandron, C.: Complexity classes for membrane systems. RAIRO Theoretical Informatics and Applications, 40 (2), 141–162 (2006)
14. Porreca, A. E., Mauri, G., Zandron, C.: Non-confluence in divisionless P systems with active membranes. Theoretical Computer Science 411 (6), 878-887 (2010)
15. Sosík, P.: The computational power of cell division in P systems: Beating down parallel computers?. Natural Computing, 2 (3), 287–298 (2003)
16. Sosík, P., Rodríguez-Patón, A.: Membrane computing and complexity theory: a characterization of PSPACE. Journal of Computer and System Sciences, 73 (1), 137–152 (2007)
17. Zandron, C., Ferretti, C., Mauri, G.: Solving NP-complete problems using P systems with active membranes. In: Antoniou, I., Calude, C.S., Dinneen, M.J. (eds.) Unconventional Models of Computation, pp. 289–301. Springer, London (2000)