

On the Reachability Problem for P Systems with Symport/Antiport

Gheorghe Păun¹ Mario J. Pérez-Jiménez²
Fernando Sancho-Caparrini²

¹Institute of Mathematics of the Romanian Academy
PO Box 1-764, 70700 Bucureşti, Romania, and
Rovira i Virgili University
Pl. Imperial Tàrraco 1, 43005 Tarragona, Spain
gpaun@imar.ro, gp@astor.urv.es

²Department of Computer Science and Artificial Intelligence
University of Seville. {Mario.Perez,Fernando.Sancho}@cs.us.es

Abstract. We address the problem of deciding whether or not a given configuration of a P system can be reached by correct transitions starting from a given initial configuration. Specifically, we consider P systems with symport/antiport rules, an attractive class which was recently introduced. As expected, the problem is undecidable in general, due to the large generative power of P systems, but, somewhat surprisingly, the reachability is decidable for configurations which take into account also the objects which are sent out of the system during the computation (the language describing such configurations is proved to be context-sensitive, hence recursive). These assertions are true both for halting configurations and for arbitrary configurations.

1 Introduction

P systems are distributed parallel computing models which start from the observation that the processes which take place in the complex structure of a living cell can be considered computations. Abstracting from the structure and the functioning of living cells, the basic ingredients of a P system are the *membrane structure*, consisting of several membranes embedded in a main membrane (called the *skin*) and delimiting *regions* where multisets of *objects* are placed; the objects evolve according to given *evolution rules*, which are applied non-deterministically (the rules to be used and the objects to evolve are randomly chosen) in a maximally parallel manner (in each step, all objects which can evolve must do it). The objects can also be communicated from a region to another one. In this way, we get *transitions* from a *configuration* of the system to another configuration. A sequence

of transitions constitutes a *computation*; with each *halting computation* we associate a *result*, the number of objects in a specified *output membrane* (this is an *elementary* membrane, that is, a membrane without any other membrane inside). The objects can also be described by strings over a given alphabet, and in this case the evolution rules are based on string processing operations, and the result of a computation can be a number or a set of strings (a language), but we do not consider this case here.

Since these computing devices were introduced ([11]) several variants have been considered. Many of them were proved to be computationally complete, able to compute all Turing computable sets of natural numbers (all recursively enumerable languages, in the case of string-objects). Moreover, it was shown that when membrane division, membrane creation, or string replication is allowed, NP-complete problems can be solved in linear time. Details (including the current bibliography of the domain and several papers which can be downloaded) can be found at the web address <http://psystems.disco.unimib.it>.

The communication of objects through membranes is one of the most important ingredients of a P system, and, roughly speaking, communication gives power to these computing devices. This led to the question (see [12]) to closely investigate the power of communication, to consider “purely communicative” systems, where the objects are not changed during a computation, but they just change their place with respect to the compartments of a P system. A first attempt to solve this problem was done in [8], where the idea of plasmids, or of vectors from gene cloning is captured by considering certain “vehicle-objects” which carry other objects through membranes. Another biochemical idea, that of membrane transport in pairs of chemicals, was recently followed in [10]. When two chemicals can pass through a membrane only together, in the same direction, the process is called *symport*. When the two chemical pass simultaneously, but in opposite directions, the process is called *antiport*. For biochemical details about symport/antiport processes at the level of alive cell membranes we refer to [1] and to [2].

The idea of coupled transport through membranes was captured in [10] by considering rules of the form (ab, in) , (ab, out) (for symport), and $(a, out; b, in)$ (for antiport). Also, more complex rules can be allowed, for instance, of the form $(ab, out; cd, in)$. (In all cases, a, b, c, d are symbol-objects.)

Both P systems with carriers and with symport/antiport were shown to be computationally universal. In the latter case, when rules of the forms (ab, in) , (ab, out) , $(a, out; b, in)$ are used, five membranes were necessary, while systems with only two membranes are universal in the case of rules $(u, out; v, in)$, with u, v strings of length at most two.

P systems with symport/antiport are attractive from several points of view: (i) they are directly inspired from biochemistry and do not contain any “artificial” ingredient, inspired from theoretical computer science, (2) they observe the conservation law, as no object is created or destroyed during a computation, (3) they essentially take into account the environment, (4) they are mathematically elegant and computationally powerful. Note that usually the first three features are not present in the case of other types of P systems.

Here we address a question which was not considered yet in the membrane computing area, that of reachability of a given configuration with respect to a given system.

The question can be raised for any type of P systems, and we expect similar results in all cases: (i) undecidability of the reachability for systems which are computationally universal (intuitively, because there are P systems which generate non-recursive sets of numbers/languages, just consider a configuration which is precisely associated with the result of a computation; whether or not it is reachable is equivalent with the membership of the associated result to the generated non-recursive set/language), (ii) decidability of the reachability of *extended configurations* (this result is obtained both for halting configurations and for configurations which are not necessarily halting). In the case of *conservative* systems, those which observe the conservation law, we call extended configuration a description of both the system (membranes and objects present in their regions) and of objects sent out of the system during the computation. When also erasing rules are allowed, we have to also take into account the objects which are “erased”. Because here we deal only with systems with symport/antiport, and they are conservative, we do not enter into details for the latter case, its study remains as a research topic (the first problem is to give a precise meaning to the idea of “taking into account the objects which are erased”).

An “explanation” for the decidability of the reachability of extended configurations is the fact that they preserve the whole “working space”, hence, intuitively, they can be computed by context-sensitive Chomsky rules. Because the context-sensitive languages are recursive, we find that the reachability of extended configurations is decidable. Actually, we will prove a stronger result: the language of extended configurations (halting or not) which are reachable in a given P system with symport/antiport rules can be generated by a so-called matrix grammar with appearance checking without using λ -rules; such grammars generate a strict subfamily of the family of context-sensitive languages.

References

- [1] B. Alberts et al., *Essential Cell Biology. An Introduction to the Molecular Biology of the Cell*, Garland Publ. Inc., New York, London, 1998.
- [2] I.I. Ardelean, The relevance of biomembranes for P systems, *Fundamenta Informaticae*, 49, 1-3 (2002), 35–43.
- [3] A. V. Baranda, J. Castellanos, F. Arroyo, R. Gonzalo, Towards an electronic implementation of membrane computing: A formal description of nondeterministic evolution in transition P systems, *Proc. 7th Intern. Meeting on DNA Based Computers* (N. Jonoska, N.C. Seeman, eds.), Tampa, Florida, USA, 2001, 273–282.
- [4] J. Dassow, Gh. Păun, *Regulated Rewriting in Formal Language Theory*, Springer-Verlag, Berlin, 1989.
- [5] D. Hauschild, M. Jantzen, Petri nets algorithms in the theory of matrix grammars, *Acta Informatica*, 31 (1994), 719–728.

- [6] O.H. Ibarra, Verification in Queue-Connected Multicounter Machines, *International Journal of Foundations of Computer Science*, 13 (2002), 115–127.
- [7] O. H. Ibarra, T. Bultan, J. Su, Reachability Analysis for Some Models of Infinite-state Transition Systems, *Proceedings of 11th International Conference on Concurrency Theory*, Pennsylvania, 2000, 183–198.
- [8] C. Martin-Vide, Gh. Păun, G. Rozenberg, Membrane systems with carriers, *Theoretical Computer Science*, 270 (2002), 779–796.
- [9] A. Obtulowicz, Membrane computing and one-way functions, *Intern. J. Found. Computer Science*, 12, 4 (2001), 551–558.
- [10] A. Păun, Gh. Păun, The power of communication: P systems with symport/antiport, *New Generation Computers*, 20, 3 (2002), 295–306.
- [11] Gh. Păun, Computing with membranes, *Journal of Computer and System Sciences*, 61, 1 (2000), 108–143, and *Turku Center for Computer Science-TUCS Report No 208*, 1998 (www.tucs.fi).
- [12] Gh. Păun, Computing with membranes (P Systems): Twenty six research topics, CDMTCS TR 119, Univ. of Auckland, 2000 (www.cs.auckland.ac.nz/CDMTCS).
- [13] Gh. Păun, Further research topics about P systems, *Pre-Proceedings of Workshop on Membrane Computing*, Curtea de Argeş, Romania, August 2001, Technical Report 17/01 of Research Group on Mathematical Linguistics, Rovira i Virgili University, Tarragona, Spain, 2001, 243–250.
- [14] Gh. Păun, G. Rozenberg, A guide to membrane computing, *Theoretical Computer Science*, 287, 1 (2002), 73–100.
- [15] Gh. Păun, G. Rozenberg, A. Salomaa, Membrane computing with external output, *Fundamenta Informaticae*, 41, 3 (2000), 259–266.
- [16] M. J. Pérez-Jiménez, A. Romero-Jiménez, Simulating Turing Machines by P systems with external output, *Fundamenta Informaticae*, 49, 1-3 (2002), 273–287.
- [17] M. J. Pérez-Jiménez, F. Sancho-Caparrini, A formalization of transition P systems, *Fundamenta Informaticae*, 49, 1-3 (2002), 261–272.
- [18] G. Rozenberg, A. Salomaa, eds., *Handbook of Formal Languages*, 3 volumes, Springer-Verlag, Berlin, 1997.
- [19] A. Salomaa, *Formal Languages*, Academic Press, New York, 1973.