Research Topics Arising from the (Planned) P Systems Implementation Experiment in Technion

Renana Gershoni[1], Ehud Keinan[1,2], Gheorghe Păun[3],
Ron Piran[1], Tamar Ratner[1], Sivan Shoshani[1]

[1] Schulich Faculty of Chemistry
Technion – Israel Institute of Technology
Haifa 32000, Israel
[2] The Scripps Research Institute
Departments of Molecular Biology and the Skaggs Institute for Chemical Biology
10550 North Torrey Pines Road, La Jolla, California 92037, USA
E-mails: keinan@technion.ac.il, keinan@scripps.edu
[3] Institute of Mathematics of the Romanian Academy
PO Box 1-764, 014700 Bucharest, Romania
and Department of Computer Science and Artificial Intelligence
University of Sevilla
Avda. Reina Mercedes s/n, 41012 Sevilla, Spain
E-mails: george.paun@imar.ro, gpaun@us.es

Summary. We formulate here a few technical (mathematical) open problems related to the *in vitro* bio-chemical experiment planned in Technion for computing the Fibonacci sequence in terms of P systems. So-called *local-loop-free* P systems are introduced and their universality for various types of P systems as well as other issues are mentioned as research questions.

1 Introduction

Although in the fall of this year membrane computing counts one decade since its beginnings (since the paper [1] was circulated as a technical report of TUCS, Finland), so far no attempt to implement P systems in a laboratory, using a bio-chemical support, was reported. Recently, such an experiment was planned, in the Chemical Faculty of Technion Institute, Haifa. This will be an *in vitro* experiment, using test tubes as membranes and DNA molecules as objects, evolving under the control of enzymes. The computation to implement was chosen to be the generation of a bunch of numbers in the famous Fibonacci sequence.

As expected, such an attempt raised a series of difficulties related to the type of P system which is possible to simulate/implement. After briefly mentioning these difficulties, we introduce the type of P systems which seem to avoid them. The basic issue is to have no loops in the evolution of objects/substances present in a membrane/compartment, because this would lead to cycles which cannot be "read" from outside in a useful way, to equilibrium states which are not "useful" for the computation.

Local-loop-free (in short, LL-free) P systems are introduced and the power of this restriction is a natural issue to investigate from a mathematical and computational point of view. We only show here that LL-free tissue-like P systems with cooperative rules are universal, but the question remains open for other classes of P systems, in particular, for the catalytic ones. A few related questions are mentioned.

2 Difficulties and (Hopefully) Solutions Related to Implementing P Systems

The basic features of membrane computing are (1) *compartmentalization*, by means of cell-like membranes, (2) *multisets* (sets with multiplicity associated with their elements, which means *counting* the objects present in membranes), (3) bio-inspired *evolution rules*, which are reaction-like (for processing multisets), communication rules (e.g., symport and antiport rules), membrane handling rules, etc., (4) *synchronization* of compartment evolution, for instance, using the rules in a maximally parallel manner, (5) *communication* between compartments; we can also mention (6) defining the result of a computation mainly for *halting* computations, but this is not specific to membrane computing (and can also be avoided).

In order to implement a P system in a laboratory, all or most of these features should be implemented. Compartments can be obtained by using standard test tubes or similar labware, multisets are usual in bio-chemistry, but... without a precise counting. Still, by defining carefully some "moles" of substances, one can count in terms of such ad-hoc moles. Anyway, full synchronization and parallelism cannot be guaranteed by bio-chemical reactions, hence a certain degree of non-determinism/approximation should be allowed in the experiment. In particular, a good degree of synchronization can be obtained by "waiting enough", such that all reactions that can take place in a test tube actually take place – and this raises an important issue: these reactions should not cycle, the process should be finite in each compartment of the system. Counting is also needed when reading the result of a computation.

In the experiment planned in Technion, the above mentioned difficulties are solved as follows: (1) test tubes for membranes (compartments), (2) multisets of pre-defined "moles" of DNA molecules, (3) enzyme driven operations with the DNA molecules, with the precaution not to have any cycle in any compartment, (4) waiting enough for reactions to take place and then (5) moving all relevant objects to the next tube in a mechanical way, with (6) the result read by spectrophotometry (certain molecules are marked and their number is estimated).

These lab solutions request finding a suitable problem or class of problems for which no cycle of substances is possible in any compartment and the solution allows a degree of approximation. We said nothing above about halting, because in the experiment this feature is not taken into consideration: a sequence of numbers (the famous Fibonacci one) are computed, hence several outputs, at precise time moments, are produced.

3 Local-loop-free P Systems

From a theoretical point of view, the central issue is that of finding a non-trivial class of P systems such that the reactions from each compartment are completed in a finite (better: small) number of steps. Otherwise stated, no compartment can contain a cycle of objects which can run forever.

This intuitive goal can be reached in various formal ways. For instance, we can request that no local transition graph contains a cycle (the catalysts are ignored). Specifically, for each region i of a P system with the set of objects O and set R_i of rules in region i, the transition graph $\gamma_i = (O, E)$ associated with region i has the set of edges defined as follows: for each $a, b \in O$,

$$(a, b) \in E \text{ iff there is } u \to v \in R_i \text{ such that } |u|_a \geq 1, |v|_b \geq 1.$$

(For a string x and a symbol a we denote by $|x|_a$ the number of occurrences of a in x.)

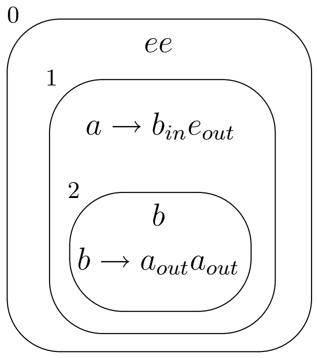

Fig. 1.

A stronger condition is to impose that no object produced in a compartment can evolve in the same compartment. In the case of non-cooperative systems, this means that the local transition graph contains no paths of length longer than or equal to two. (For cooperative systems this assertion is not true: having the rules $a \to b,\ bc \to cc$, the local transition graph contains the path (a, b, c), but it is possible not to actually have two reactions in a row, because without c, the product b of the first reaction cannot evolve.) This latter condition is similar to the way the P systems with immediate communication are defined (see [2]): each product of a reaction is immediately communicated to one of the neighboring membranes.

Thus, formally, we can define several properties which ensure the local-loop-freeness. Defining such properties and investigating the P systems obeying them is one of the *research topics* we want to point out here.

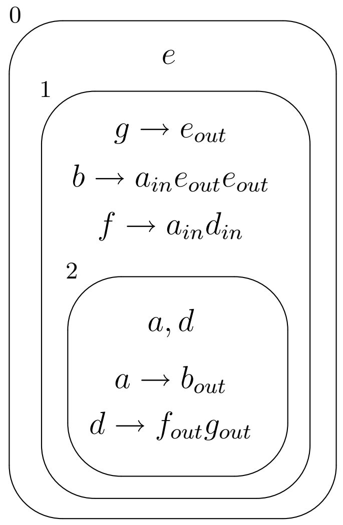

Fig. 2.

In what follows, we briefly discuss the P systems which are local-loop-free (in short, LL-free) in the sense of the previous definition: no cycle exists in any local transition graph.

4 Some Examples

We start by considering three (non-semilinear) sets of numbers which can be computed by P systems of a rather similar form. Figures 1, 2, and 3 present non-cooperative P systems (denoted by Π_1, Π_2, Π_3) generating, respectively, the following sets of numbers:

$$\begin{aligned} N(\Pi_1) &= \{2^n \mid n \geq 1\},\\ N(\Pi_2) &= \{n^2 \mid n \geq 1\},\\ N(\Pi_3) &= \{1, 2, 3, 5, 8, 13, ...\}. \end{aligned}$$

The third sequence is the Fibonacci one (each element is the sum of the previous two; here we start with 1 and 2 as the initial numbers), and this system Π_3 is planned to be implemented.

These systems can be represented in a more intuitive way (in what concerns the reactions taking place in compartments and the objects communicated) as tissue-like P systems with immediate communication. For systems Π_1 and Π_3 this is done in Figures 4 and 5, respectively; the case of Π_2 is left to the reader. On the arrows are indicated the objects which are communicated between the respective membranes.

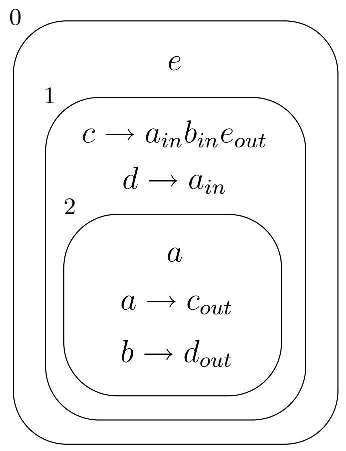

Fig. 3.

In all cases, of both cell-like and tissue-like systems, the result is collected in the form of the number of copies of a special object e in a designated membrane which has no other role in the system. We call it a *output membrane*; it contains no rule, hence no objects can evolve in it. In all cases, the environment can be used instead of this membrane, but it is "more practical" to work with a output membrane.

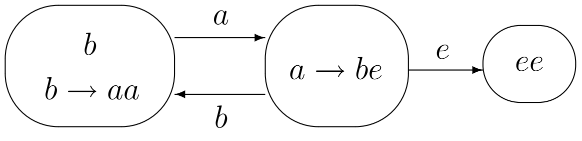

Fig. 4.

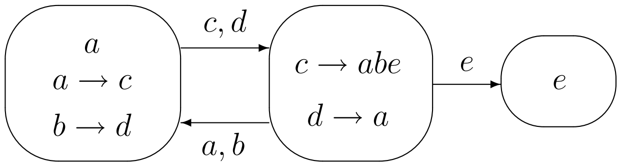

Fig. 5.

During the discussions in Technion in search of a suitable problem to implement, when we have arrived at the construction of the system in Figure 3, a generalized glee was expressed by the young members of the team, who exclaimed: "this is doable!" The idea was summarized at various stages in nice graphical forms – one of them is given in Figure 6 (the output membrane is here the inner one and one additional external membrane is considered as an infinite supplier of "raw materials") – and then a group photo was taken, to celebrate the moment (see Figure 7). Well, whether or not this moment deserves also to be celebrated with champagne it remains to be seen after trying the experiment...

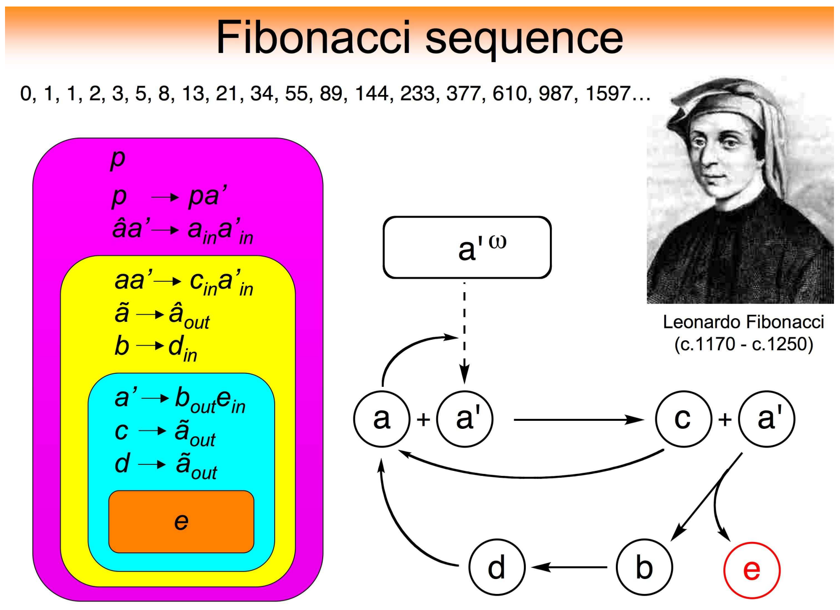

Fig. 6.

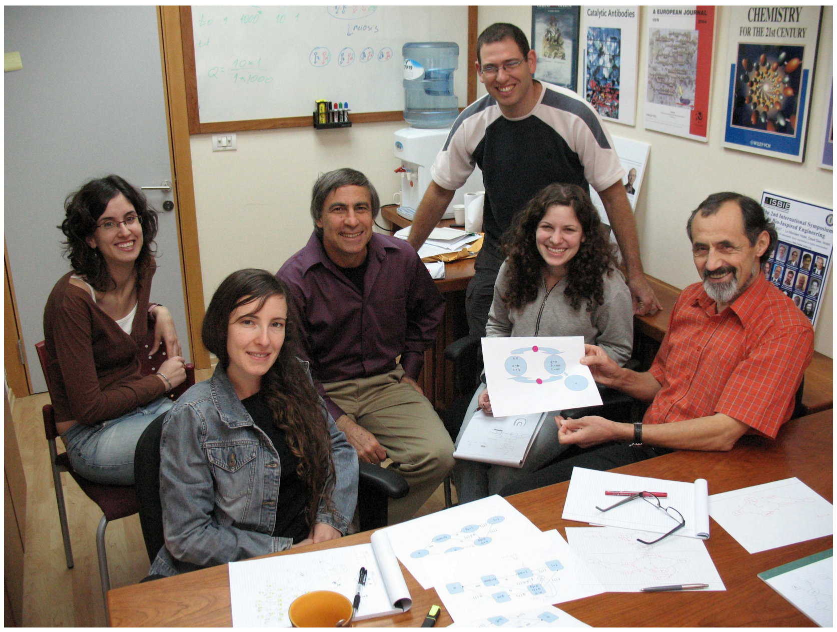

Fig. 7.

5 And Now Start the Problems

We list here only some of the most natural ones:

- Prove universality for LL-free P systems. Of course, in this case we need to consider as successful only halting computations. For tissue-like cooperative systems we give a proof of universality in the next section, but for other classes of P systems, in particular, for catalytic P systems (with two or more catalysts, or with one catalyst and various controls on using the rules) the problem remains open.
- What about considering systems with a membrane structure like those in Figures 1–3, i.e., with only two membranes for computing and one additional membrane for collecting the result of a computation? Are also such LL-free systems universal? Note that even the system in the next section, using cooperative rules, has three "computing" membranes.
- Find other examples of systems (of interest) with the LL-freeness property or with the membrane structure of the form in Figures 1–3.
- Note that the systems considered above are of a generative form, they start from an initial configuration and generate infinite sequences of numbers. Devise input-output systems, computing a function (of some interest).

- Is any chance to solve NP-complete problems in this framework?
- What about sorting, ranking, or other computer science applications of P systems (as reported in the literature), based on LL-free P systems?

6 Universality of Cooperative LL-free Systems

We denote by $tNOP_m^{llf}(coo)$ the family of sets of numbers $N(\Pi)$ generated by LL-free tissue-like P systems with cooperative rules having at most $m \geq 1$ membranes, with immediate communication and with the result collected in a special output membrane which has only this role (no object evolve in this membrane, it has no rule inside). If the result is collected in the environment, then this output membrane can be saved, but here we choose to consider it.

In this framework, we can immediately prove the following result (as usual, NRE denotes the family of recursively enumerable sets of natural numbers):

Theorem 1. *$tNOP_m^{llf}(coo) = NRE$ for all $m \geq 4$.*

The proof is based on simulating a register machine $M = (m, H, l_0, l_h, I)$ (number of registers, set of labels, initial label, halting label, set of instructions) by a P system Π constructed as suggested in Figure 8. Without any loss of the generality, we may assume that when halting, M has all registers different from register 1 empty.

All labels in H, primed versions of them (for each $l \in H$ we consider $l', l'', l''', l^{iv}, \bar{l}, \hat{l}$, too), as well as objects $a_r, 1 \leq r \leq m$, associated with the registers of M are objects in Π. We start with only one object in the system, namely l_0, present in membrane 1.

For each ADD instruction $l_i : (\texttt{ADD}(r), l_j, l_k)$ in I we introduce the rules

$$\begin{aligned}
&l_i \to a_r \bar{l}_i \text{ in membrane 1,}\\
&\bar{l}_i \to \hat{l}_i \text{ in membrane 2,}\\
&\hat{l}_i \to l_j \text{ and}\\
&\hat{l}_i \to l_k \text{ in membrane 3.}
\end{aligned}$$

The simulation of the ADD instruction is obvious: the increment of register r is done in membrane 1 and the non-deterministic choice of the next instruction to apply is done in membrane 3.

For each SUB instruction $l_i : (\texttt{SUB}(r), l_j, l_k)$ in I we introduce the rules

$$\begin{aligned}
&l_i \to l'_i l''_i \text{ in membrane 1,}\\
&l'_i a_r \to l'''_i \text{ and}\\
&l''_i \to l^{iv}_i \text{ in membrane 2,}\\
&l^{iv}_i l'''_i \to l_j \text{ and}\\
&l^{iv} l'_i \to l_k \text{ in membrane 3.}
\end{aligned}$$

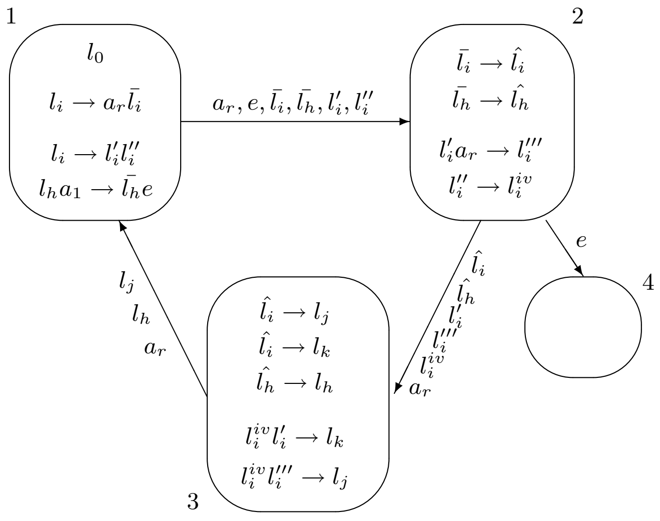

Fig. 8.

The simulation of the SUB instruction is done in three steps (a complete cycle through membranes 1, 2 and 3). In membrane 1 one introduces the objects l'_i, l''_i which are moved, together with all objects a_r to membrane 2. Here, l'_i tries to subtract one from the value of register r. If this is possible, then the object l'''_i is introduced, otherwise l'_i remains unchanged. Simultaneously, l''_i introduces the object l^{iv}_i. This object behaves as a checker in membrane 4: if it mets here l'''_i (hence the subtraction was possible), then one introduces the object l_j, otherwise one introduces the object l_k. In both cases the continuation is as necessary in the register machine M.

We also consider the rules

$$\begin{aligned}&l_h a_1 \to \bar{l}_h e \text{ in membrane 1,}\\&\bar{l}_h \to \hat{l}_h \text{ in membrane 2,}\\&\hat{l}_h \to l_h \text{ in membrane 3.}\end{aligned}$$

In the end of the computation with respect to M, the object l_h transforms all objects a_1 (hence the contents of the first register of M) into objects e, which are moved to membrane 4. Thus, the computation ends with a number of copies of e in membrane 4 equal to the number generated in the first register of M. Thus, $N(M) = N(\Pi)$ and we have the inclusion $NRE \subseteq tNOP_4^{llf}(coo)$.

The converse inclusion can be proved in a straightforward way or we can invoke for it the Turing-Church thesis. □

References

1. Gh. Păun: Computing with membranes. *Journal of Computer and System Sciences*, 61, 1 (2000), 108–143 (first circulated as TUCS Report 208 - Turku Center for Computer Science, November 1998, `www.tucs.fi`).
2. Gh. Păun: *Computing with Membranes: An Introduction*. Springer, Berlin, 2002.