On the Efficiency of Promoters and of Cooperative Rules in P Systems

Roberto Barbuti[1], Andrea Maggiolo-Schettini[1], Paolo Milazzo[1], Simone Tini[2]

[1] Dipartimento di Informatica, Università di Pisa
Largo Pontecorvo 3, 56127 Pisa, Italy
{barbuti,maggiolo,milazzo}@di.unipi.it

[2] Dip. di Scienze della Cultura, Politiche e dell'Informazione, Università dell'Insubria
Via Carloni 78, 22100 Como, Italy
simone.tini@uninsubria.it

1 Introduction

Membrane systems (P systems) were introduced by Paun in [5] as distributed parallel computing devices inspired by the structure and the functioning of cells. In the extension of [3] the application of rules may be conditioned by the presence of *promoter objects*. A promoter does not participate in the application of rules, and a single promoter may enable the application of several rules and multiple applications of each one of these rules. P systems with promoters have been shown to be universal even when non-cooperative rules are considered [3]. The same holds for P systems without promoters but with cooperative rules [5]. We aim at comparing the use of promoters with the use of cooperative rules from the point of view of efficiency. Actually, the kind of efficiency we are interested in is not the ability of solving NP complete problems in polynomial time (as in [6]), but the ability of solving in constant time problems solvable in linear time. In this paper we show that there exists a problem that can be solved in constant time with cooperation and that requires at least linear time with promoters. Whether also the opposite holds is left as an open problem.

2 P Systems with Promoters

In P systems with promoters [3] an evolution rule may have some promoters that are objects required to be present in the membrane in order to enable the rule. We can assume that all evolution rules have the following form:

$$u \rightarrow (v_h, here)(v_o, out)(v_1, in_{l_1}) \dots (v_n, in_{l_n})|_p$$

where u is the multiset of objects consumed by the rule, $\{l_1, \dots, l_n\}$ is a set of membrane labels, $v_h, v_o, v_1, \dots, v_n$ are the objects (grouped in multisets by target)

produced by the rule and p is the multiset of promoters of the rule. Application of evolution rules is done with maximal parallelism, as usual. Formally:

Definition 1. *A* P system *Π is given by $\Pi = (V, \mu, w_1, \ldots, w_n, R_1, \ldots, R_n)$, where: (i) V is an* alphabet *whose elements are called* objects*; (ii) $\mu \subset \mathbb{N} \times \mathbb{N}$ is a* membrane structure*, such that $(l_1, l_2) \in \mu$ denotes that the membrane labeled by l_2 is contained in the membrane labeled by l_1; (iii) w_j with $1 \leq j \leq n$ are strings from V^* representing multisets over V associated with the membranes $1, \ldots, n$ of μ; (iv) R_j with $1 \leq j \leq n$ are finite sets of* evolution rules *associated with the membranes $1, \ldots, n$ of μ.*

In this paper we assume P systems to be closed computational devices, namely objects cannot be sent out of the skin membrane (i.e. rules sending objects out are not allowed in the skin membrane) and cannot be received by the skin membrane from outside. We will usually consider the multiset of objects initially contained in the skin membrane as the input of the P system and the multiset of objects contained in the skin membrane of a final configuration as an output. Note that infinite evolutions are not considered as valid computations.

From [1] it follows that any P system with promoters can be translated into another one that computes the same function, by performing equivalent evolution steps, having a flat membrane structure consisting only of the skin. The idea is to enrich the alphabet of the P system with objects labeled with membrane indexes and to use such objects in the skin membrane of the flat system to represent objects placed in some inner membrane of the original system. This technique was previously used in [2] but with P systems without promoters.

Theorem 1. *Every P system with promoters can be translated into another one whose membrane structure consists only of the skin membrane.*

P systems dealing with multiset languages can be either language acceptors or generators. In the first case a multiset is provided as the input and the result of the computation says whether such a multiset belongs to a language or not. In the second case the P systems has a fixed initial configuration and can give as results, in a non-deterministic way, all possible multisets belonging to a given language.

Let us formalize the notion of P system used as language acceptor.

Definition 2. *An* acceptor P system *for a multiset language L over an alphabet Σ is a P system $\Pi_L = (\Sigma \cup \mathcal{C} \cup \{T\}, \mu, w_1 \cup \ell, w_2, \ldots, w_n, R_1, \ldots, R_n)$ where: (i) $\mathcal{C}$ is a set of* control objects *such that $\Sigma \cap \mathcal{C} = \varnothing$; (ii) T is a special object not contained in $\Sigma \cup \mathcal{C}$; (iii) w_i, for $1 \leq i \leq n$, are multisets of objects in $\mathcal{C}$; (iv) when the placeholder ℓ is replaced by a multiset of objects the output of the computation of the P system says whether such a multiset belongs to L as follows: the multiset is accepted (belongs to L) if and only if a final configuration can be reached with T appearing in the output.*

We remark that one could define equivalent notions of acceptor P systems without assuming Σ and $\mathcal{C}$ to be disjoint sets or by assuming that a multiset

is accepted if and only if a final configuration can be reached (by ignoring the presence of T). The first of these two alternative notions can be simulated by ours by assuming that there exists in $\mathcal{C}$ a primed copy a' of every object a that should be shared with Σ. Such primed objects are then rewritten into their unprimed version in the first evolution step of the system. The second of the two alternative notions can be simulated simply by adding T to w_1 and by ensuring that there is no rule in R_1 using such a special object.

3 Efficiency of Promoters and of Cooperation

Let us take the language $L = \{a^{2n} \mid n \geq 0\}$. By exploiting cooperative rules, L can be accepted in constant time. In fact, we can take a P system with only one membrane containing the object T and the rules $aa \to \lambda$ and $aT \to \lambda$.

A solution without cooperation and exploiting promoters consists in a membrane with objects T and 1 and the following rules:

$a \to a\|_1$	$T \to F\|_{bb2}$	$b \to \lambda\|_{ok}$	$1 \to 2$	$2 \to 3$
$a \to b\|_1$	$T \to F\|_{cc2}$	$c \to \lambda\|_{ok}$	$3 \to 4$	$4 \to 1\,\|_a$
$a \to c\|_1$	$T \to OK\|_{bc3}$	$OK \to T$		

Such a solution is linear in time w.r.t. n. We can show that without cooperation a solution in constant time cannot be given.

Theorem 2. *The language $\{a^{2n} \mid n \geq 0\}$ cannot be accepted in constant time without using cooperating rules.*

Proof. By contradiction, let us assume that $L = \{a^{2n} \mid n \geq 0\}$ can be accepted in constant time, actually, that there exists and acceptor P system $\mathcal{P}$ able to accept any multiset of the language L in at most k execution steps.

Given a possible accepting execution of $\mathcal{P}$, let $R_1, \ldots, R_k$ be the sets of rules that are applied at least once in each of the k execution steps, respectively. Let $r_{i,j}$, with $1 \leq i \leq k$, denote one of the m_i rules of set R_i, namely $R_i = \{r_{i,1}, \ldots, r_{i,m_i}\}$. Since L is infinite, whereas the number of execution steps and of evolution rules are bounded, there must exist an infinity of different executions (each accepting a different multiset in L) with the same sets of applied rules $R_1, \ldots, R_k$ and that differ only on the number of times such rules are applied in each execution step. Let $x_{i,j}$ be the number of times rule $r_{i,j}$ is applied (in the i-th execution step).

Let $N \subset \mathbb{N}$ be an infinite set s.t. $L' = \{a^{2n} \mid n \in N\}$ is a set of multisets (sub-language of L) accepted by executions in which the same sets of rules $R_1, \ldots, R_k$ are applied. At least one of the sets $R_1, \ldots, R_k$ must contain a rule in which object a is either consumed or used as a promoter, otherwise we would have that only control objects are used so that $\mathcal{P}$ would accept any multiset.

Let us assume first that a is not consumed by any of the rules in $R_1, \ldots, R_k$, but used as a promoter of some of these rules. Let $r_{i,j}$ be any of the rules having a as a promoter, namely $r_{i,j} = o \to u|_{va^p}$ with $o \in V$, $u, v \in V^*$ and $p \in \mathbb{N}^+$. Since $r_{i,j}$ has been applied ($x_{i,j}$ times), we have that at the i-th step there must be at least

p occurrences of a. This must hold for any multiset in L' to be accepted, namely for any $n_1 \in N$. Now, given any $n_1 \in N$, let us take $n_2 = n_1 + 1$. It holds that $n_2 \notin N$, since N contains only even numbers whereas n_2 is odd by construction. We have that $a^{n_2} \notin L$ but it is accepted by an execution of the acceptor P system in which rules $R_1, \ldots, R_k$ are applied exactly as many times as in the execution that accepts a^{n_1}. This holds because the additional a is not consumed by any rule and has not an influence on the applicability of rules having such an object as a promoter. Hence, we have a contradiction.

Let us assume now that there are some rules in $R_1, \ldots, R_k$ consuming a. Let $r_{i,j}$ be any of these rules, namely $r_{i,j} = a \to u|_{va^p}$ with $u, v \in V^*$ and $p \in \mathbb{N}$. Since $r_{i,j}$ has been applied ($x_{i,j}$ times), we have that at the $(i+1)$-th step $x_{i,j}$ copies of u are present. This holds for any multiset in L' to be accepted, namely for any $n_1 \in N$, and, differently from the previous case, we have that the value of $x_{i,j}$ might be proportional to n_1. Given any $n_1, n_2 \in N$ with $n_1 < n_2$, let us take $n_3 = n_1 + 1$. It holds that $n_3 \notin N$, since N contains only even numbers whereas n_3 is odd by construction. We have that $a^{n_3} \notin L$ but it is accepted by an execution of $\mathcal{P}$ in which applied rules are $R_1, \ldots, R_k$, and they are applied at least as many times as in the execution that accepts a^{n_1} and at most as many times as in the execution that accepts a^{n_2}. The fact that T is produced is guaranteed by the fact that, in order to accept a^{n_1}, T was either present from the beginning as a control object of the initial configuration or produced by one of the rules in $R_1, \ldots, R_k$, and this still holds for a^{n_3} where the initial control objects and the applied rules are the same. Moreover, the fact that the accepting execution of a^{n_3} terminates is guaranteed by the fact that the objects produced during the execution were produced (possibly in a greater quantity) also during the accepting execution of a^{n_2}. This means that the accepting execution of a^{n_3} does not enable the application of rules that were not applicable in the accepting execution of a^{n_2}. Summing up, also this case leads to a contradiction.

Open Problem. Does there exist any language that can be accepted in constant time only if promoters are exploited?

References

1. R. Barbuti, A. Maggiolo-Schettini, P. Milazzo and S. Tini: A P systems flat form preserving step-by-step behaviour. Fundam. Inform. 87, 1–34, 2008.
2. L. Bianco and V. Manca: Encoding–decoding transitional systems for classes of P systems. Proc. Work. on Membrane Computing, LNCS 3850, 134–143, 2006.
3. P. Bottoni, C. Martin-Víde, G. Păun and G. Rozemberg. Membrane systems with promoters/inhibitors. Acta Informatica 38, 695–720, 2002.
4. M. Ionescu and D. Sburlan: On P systems with promoters/inhibitors. J. Univ. Comp. Sci. 10, 581-599, 2004.
5. G. Păun, Membrane computing. An introduction, Springer, 2002.
6. C. Zandron, C. Ferretti and G. Mauri: Solving NP-Complete problems using P systems with active membranes. Proc. UMC'2K, 289-301, 2000.