Look-Ahead Evolution for P Systems

Sergey Verlan

LACL, Département Informatique, Université Paris Est
61, av. Général de Gaulle, 94010 Créteil, France
verlan@univ-paris12.fr

Summary. This article introduces a new derivation mode for P systems that permits to make a look-ahead on the next configuration and check for some forbidding conditions on it. The interesting point is that the software implementation of this mode needs very small modifications to the standard algorithm of rule assignment for maximally parallelism. As benefits of this mode some non-deterministic proofs become deterministic. As an example we present a generalized communicating P system that accepts 2^n in n steps in a deterministic way. Another example shows that in the deterministic case this mode is more powerful than the maximally parallel derivation mode. Finally, this mode gives a natural way to define P systems that may accept or reject a computation.

1 Introduction

P systems are defined as non-deterministic computational devices. However, for implementation reasons, it is better to limit the inherent non-determinism to a smaller degree and eventually have a deterministic evolution. One of such approaches is based on an examination of the next configuration(s) and cutting off non-deterministic computational branches that have some pre-defined properties. The notion of k-determinism [7, 2] is closely related to such optimizations. For a system having the k-determinism property one can examine all possible future configurations for at most k steps and find a single evolution that is not forbidden. This gives an efficient procedure of the construction of the next configuration. However, it is not an easy task to prove that a P system has this property.

A derivation mode lies in the heart of the semantics of P systems as it permits to specify which multiset among different possible applicable multisets of rules can be applied. When P systems were introduced, only the maximally parallel derivation mode was considered which states that corresponding multisets should be maximal, i.e. non-extensible. With the apparition of the minimal parallel derivation mode [3] the concept of the derivation mode had to be precisely defined and [5] presents a framework that permits to easily define different derivation modes.

This article tries to express the notion of one-step look-ahead in terms of a derivation mode which gives a way to implement P systems in a more efficient way. The look-ahead is a forbidding condition formalized by a set of forbidden rules that should not be applicable after a maximally parallel multiset of normal (non-forbidding) rules was chosen. Such a formalization needs a small overhead and can be easily incorporated and efficiently implemented in already existing software simulators for P systems. In more general way, the look-ahead derivation can be considered as a further evolution of the notion of k-determinism (more precisely of 1-determinism), but without restricting to a deterministic evolution.

The look-ahead mode can give advantages in terms of deterministic evolution of the system and we show an example that demonstrates that the evolution in the look-ahead mode introduces more power into the system. Moreover, it is known that a deterministic evolution usually sequentializes the computation and it needs more steps. With the look-ahead derivation mode we show that deterministic computations can be efficient by giving an example of a P system with minimal interaction that can recognize 2^n in n steps. An interesting side effect of the definition permits to define computations that are accepted or rejected without introducing additional symbols.

2 Definitions

We do not present here standard definitions. We refer to [10] for all details.

We also assume that the reader is familiar with standard notions of P systems, which can be consulted in the book [8] or at the web page [9]. We shall only focus on the semantics of the evolution step. We will follow the approach given in [5], however we will not enter into deep details concerning the notation and the definition of derivation modes given there. Consider a P system Π of any type evolving in any derivation mode. The key point of the semantics of P systems is that according to the type of the system and the derivation mode δ for any configuration of the system C a set of multisets of applicable rules, denoted by $Appl(\Pi, C, \delta)$, is computed. After that, one of the elements from this set is chosen non-deterministically for the further evolution of the system. In order to define the look-ahead derivation mode we suppose that the set of rules of Π, denoted by R, is composed from two parts: normal rules R_N and forbidden rules R_f, i.e. $R = R_N \cup R_f$. Then we define $Appl(\Pi, C, LA\delta)$ as follows:

$$Appl(\Pi, C, LA\delta) = \{R' \mid R' \in Appl(\Pi, C, \delta) \text{ and } R' \cap R_f = \emptyset\}.$$

This means that only those multisets of rules which do not contain any rule from the forbidden set R_f can be considered for further evolution of the system. The set R_f can be replaced by other checking conditions, we shall discuss them in Section 4. By convention, we shall skip δ if it is the maximally parallel derivation mode ($\delta = max$) and call the obtained mode simply look-ahead derivation mode or LA mode.

We remak that the look-ahead derivation can be considered for any derivation mode, however in this article we shall consider only look-ahead derivation for the maximally parallel derivation mode, which is the most commonly used.

Let us consider the particularities of the look-ahead derivation mode. In fact, the rule set R_f gives conditions that shall not be satisfied by the current configuration with the condition that a particular multiset of rules from $Appl(\Pi, C, \delta)$ will be applied. This differentiates the look-ahead derivation mode from permitting or forbidding conditions which are checked *before* the assignment of objects to rules is done (in order to see if the rule is applicable), while the conditions in LA mode are checked *after* all assignments of objects to rules are done. This gives a greater flexibility as such a procedure permits to evaluate next possible configurations and to cut off some of them according to R_f. In such a way the non-determinism of the system may be significantly decreased.

We remark that the overhead introduced by such a procedure is minimal and we discuss in Section 4 possible implementations of the look-ahead derivation mode.

Another interesting point is that it is possible that all multisets from the set $Appl(\Pi, C, \delta)$ contain rules from R_f. In this case, $Appl(\Pi, C, LA\delta)$ will be empty, hence a halting configuration is reached. It is possible to differentiate this halting case from the case when $Appl(\Pi, C, \delta)$ is also empty and naturally introduce rejecting and accepting computations. This is particulary interesting for decision P systems, because it gives a natural way to obtain an answer `yes` or `no` without the need for additional symbols.

3 Examples

In this section we give two examples that show the interest of the look-ahead derivation mode. The first example presents a deterministic recognition of 2^n in n steps using minimal symport/antiport and conditional uniport, while the second example shows how the initial number of symbols can be increased by minimal symport/antiport P systems in a deterministic way.

3.1 Deterministic recognition of 2^n

In this subsection we consider P systems with minimal interaction which are a restricted variant of generalized communicating P systems [12]. We recall that the later systems are a purely communicating model defined on a graph and having rules of form $(A, i)(B, k) \rightarrow (A, j)(B, m)$, where A and B are two multisets of objects and i, j, k, m are labels of membranes (cells). This rule permits to move multisets A and B from cells i and k to cells j and m synchronously. We remark that symport, antiport and conditional uniport [11] rules are a particular case of these general communication rules.

The minimal interaction rules are obtained from the generalized communication rule by restricting multisets A and B to one symbol each. Minimal symport and minimal antiport rules are a particular case of minimal interaction rules.

Consider the following system $\Pi = (O, E, w_1, w_2, R)$, having 2 cells (0 denotes the environment) where $O = \{A, B, Z\}$, $E = \emptyset$, $w_1 = \{A^k\}$, $w_2 = \{B, Z\}$ and $R = R_N \cup R_f$ is defined as follows.

$R_N = \{1 : (A,1)(A,1) \to (A,2)(A,1),\ 2 : (A,1)(B,2) \to (A,2)(B,1)\}$ and

$R_f = \{3 : (A,1)(Z,2) \to (A,0)(Z,2)\}$.

We remark that the first rule is a conditional uniport rule that sends a copy of A from cell 1 to cell 2, providing that another A remains in cell 1. The second rule is an antiport rule exchanging A and B in cell 1 and 2. The third rule is in fact an uniport rule of A to the environment, but because of the definition an interaction of two symbols is required, hence a dummy symbol Z is present in cell 2.

Consider now the evolution of the system. Let C_0 be the initial configuration. If k is even then all three rules are applicable. Hence, $\mathit{Appl}(\Pi, C_0, \mathit{max})$ contains two multisets of applicable rules: $\{1^{k/2}\}$ and $\{1^{k/2-1}, 2, 3\}$. By the definition of the LA mode, the second multiset is eliminated and only the first possibility remains. It is clear that a similar reasoning applies to all configurations C having an even number of symbols A in cell 1 and a symbol B in cell 2. If k is odd, then $\mathit{Appl}(\Pi, C_0, \mathit{max})$ contains following multisets of rules: $\{1^{(k-1)/2}, 2\}$ and $\{1^{(k-1)/2}, 3\}$. By the definition of LA mode the second possibility is eliminated and only the first one remains. The same holds for all configurations having an odd number of A in cell 1 and a copy of symbol B in cell 2.

Now consider the first application of rule 2. It might happen only when the number of symbols A in cell 1 is odd. In all consequent configurations symbol B is present in cell 1. Consider a further configuration having m symbols A in the first cell. If m is even, then again two multisets of rules are applicable in max mode: $\{1^{m/2}\}$ and $\{1^{m/2-1}, 3\}$ and only the first one remains in the LA mode. If m is odd, then there is only one applicable multiset in max mode: $\{1^{(m-1)/2}, 3\}$ and there are no applicable rules in LA mode.

The recognition of 2^n is done as follows. It is known that if a number $k = 2^n$ is divided by 2 in a cycle, then at each step the quotient is always even, except at the end when it becomes 1. For a number $k \neq 2^n$, a similar process yields an odd number $t > 2$. Repeating this procedure for $t-1$ yields another odd number $t' \geq 1$. Rule 1 permits to divide the number of A's in cell 1 by two at each step. Rule 2 permits to decrement once the number of A's in cell 1. Hence, if initially $k = 2^n$, then at each step an even number of A's will be present in cell 2, except the last one where the rule 2 will be applied. Otherwise, when an odd number $t > 2$ of symbols A will be present in cell 1, both rules 1 and 2 will be applied. Further, an odd number of symbols A will appear in cell 1 and the computation will stop. In the first case we obtain an accepting computation, while in the second one the computation is rejecting. We remark that the acceptance of a computation may be done in other ways as it is shown in Section 4.

3.2 Deterministic minimal symport/antiport on a tree structure

In this subsection we consider P systems (having a tree structure) with minimal symport and antiport rules. An antiport rule is denoted as $(u, in; v, out)$ and permits to exchange the multiset of objects v present in the membrane i where this rule is located with the multiset of objects u present in the parent membrane of i. A symport rule, denoted as (u, in) or (v, out), permits to send a multiset v to the parent membrane or the multiset u to one of inner membranes. In the case of minimal antiport, respectively symport, the size of the multisets u and v is equal to one, respectively two.

We start by the following remark.

Remark 1 *For any deterministic P system with minimal symport/antiport rules working in maximally parallel derivation mode, the number of objects initially present inside the system, i.e. not in the environment, cannot be increased.*

The proof of the above assertion may be done in a similar way as it was done for the case of one membrane in [1] and [6]. The main argument used in those articles remains valid: if the number of objects is increasing, then any rule that permits to bring an additional symbol from the environment will be used an arbitrary number of times because of the minimality of rules and determinism.

However, the situation changes if the look-ahead derivation mode is permitted. Then the following construction permits to bring one symbol from the environment, deterministically.

Let $\Pi = (\{p, A\}, \{A\}, [_1[_2]_2]_1, \{p\}, \emptyset, R_1, R_2 \cup R_2^f)$ be a P system with minimal symport rules having two membranes (the first membrane contains initially symbol p while the second one is empty). We define the sets of rules R_1 and R_2 as follows (by the superscript f we denote the forbidding set of rules).

$R_1 = \{1 : (p, out);\ 2 : (pA, in)\}$,

$R_2 = \{3 : (pA, in)\}$ and $R_2^f = \{4 : (A, in)\}$.

The system works as follows. Firstly the symbol p is sent to the environment by rule 1 and after that it brings a copy of symbol A by rule 2. Now, in the maximally parallel derivation mode there are two applicable multisets of rules: $\{3\}$ and $\{1, 4\}$. In LA mode the second multiset is eliminated, hence only rule 3 can be applied. In such a way, the number of symbols (A) is increased, deterministically.

Since the number of objects can be varied, we conjecture that a deterministic register machine can be simulated, i.e. we conjecture that deterministic P systems with minimal symport/antiport working in LA mode can recognize any recursively enumerable set of numbers.

4 Implementation ideas

In this section we discuss some ideas about the practical implementation of the look-ahead maximally parallel derivation mode. We consider the classical implementation of the maximally parallel derivation mode which orders rules and applies the rules maximal number of times according to the order and after that uses backtracking to decrease the number of applications of rules of a higher order and increase the number of applications of rules of a lower order. In this setup it is enough to place rules from R_f after the rules from R_N and to use an additional condition that if a rule from R_f is chosen then the current multiset should be discarded and a new backtracking round should begin. Hence, only the last condition shall be additionally implemented, which is not so difficult.

Another possibility is to replace rules from R_f by an union of finite sets that check the presence of the symbols from the left-hand side of rules from R_f. In this case it is enough to check that these sets are not present in the configuration after all rules are chosen, supposing that the choice of rules marks or blocks in some way used symbols. We recall that the difference between this check and ordinary permitting/forbidding checks is that it should be done *after* an assignment of object to rules is done.

The rejecting condition may be replaced by an emptiness check of a particular cell, or, in a more general setup, by checking for some finite state conditions like it is done for P automata (see [4] for an overview).

5 Final remarks

In this paper we introduced a new derivation mode for P systems: the look-ahead mode. In some sense, this mode is an extension of the maximally parallel derivation mode and all results formulated for the latter one are true for the look-ahead mode. We also think that in the non-deterministic case both modes have same computational properties. In a lot of cases forbidden rules can be replaced by trapping rules that will move corresponding symbols to a trap membrane or will transform them to trapping symbols and the computation will never stop. However, in the deterministic case the behavior of two modes is very different, as it is shown in Subsection 3.2.

We would like to mention some differences between the look-ahead mode and the concept of k-determinism introduced in [7]. The notion of k-determinism is a property of a P system that permits to examine all possible future configurations for at most k steps and find a single evolution that is not forbidden. This property cannot be easily checked for a P system. The look-ahead derivation mode is not a property but a procedure that permits to possibly limit the non-determinism of the system.

As further research topics we would mention the extension of the look-ahead for k steps ahead. However, it is not clear if the gain in power is justified as the computational overhead needed to compute further k configurations is quite big. Another interesting problem would be the study of the efficiency of the new mode. We think that a lot of existing proofs can be simplified using the look-ahead and, moreover, efficient deterministic or almost deterministic solutions for different computational problems may be constructed. In particular, it would be interesting to give a deterministic simulation of a register machine by deterministic minimal symport/antiport P systems.

Instead of forbidden rules one may consider sets or even multisets of rules that cannot be applied together. This is a generalization of the concept of forbidden rules, because set R_f corresponds to set of pairs $\{(r, r') \mid r \in R_N, r' \in R_f\}$. This permits a finer control of rules, in some sense similar to programmed grammars, and it can be implemented quite easily.

Acknowledgements

The author would like to thank A. Alhazov, E. Csuhaj-Varjú and R. Freund for their precious comments related to the topic of the paper. The author also acknowledges the support by the Science and Technology Center in Ukraine, project 4032.

References

1. A. Alhazov, Y. Rogozhin, S. Verlan: Symport/antiport tissue P systems with minimal cooperation, In *Proceedings of the ESF Exploratory Workshop on Cellular Computing (Complexity Aspects), Sevilla (Spain)*, 37–52.
2. A. Binder, R. Freund, G. Lojka, M. Oswald: Implementation of catalytic P systems. In *Proc. of CIAA 2004* (M. Domaratzki et al. eds.), LNCS 3317, 45–56.
3. G. Ciobanu, L. Pan, G. Păun, M. J. Pérez-Jiménez: P systems with minimal parallelism. *Theor. Comput. Sci.* 378(1), (2007), 117–130.
4. E. Csuhaj-Varjú: P automata. In *Proc. of WMC 2004* (G. Mauri et al., eds.), Milano, Italy, 2005, LNCS 3365, 19-35.
5. R. Freund, S. Verlan: A formal framework for static (tissue) P systems. *Proc. of WMC 2008* (G. Eleftherakis et al., eds.), Thessaloniki, Greece, Springer, 2007, LNCS 4860, 271–284.
6. P. Frisco and H. Hoogeboom. P systems with symport/antiport simulating counter automata. *Acta Informatica*, 41(2-3):145–170, 2004.
7. M. Oswald: *P Automata.* PhD thesis. Vienna University of Technology (2003).
8. Gh. Păun, *Membrane Computing. An Introduction.* SpringerVerlag, 2002, 163, 226–230.
9. The Membrane Computing Web Page: `http://ppage.psystems.eu`
10. G. Rozenberg, A. Salomaa, eds., *Handbook of Formal Languages.* Springer-Verlag, Berlin, 1997.
11. S. Verlan, F. Bernardini, M. Gheorghe, M. Margenstern: On communica- tion in tissue P systems: conditional uniport. In *Proc. of WMC 2006* (H. J. Hoogeboom et al., eds.), Leiden, 2006, LNCS 4361, 521–535.
12. S. Verlan, F. Bernardini, M. Gheorghe, M. Margenstern: Generalized communicating P systems. *Theor. Comput. Sci.* 404 (1-2) (2008), 170–184.