ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume **13**, Number 2, 2010, 208–216

Notes on Catalytic P Systems

Dragoş SBURLAN

Faculty of Mathematics and Informatics,
Ovidius University of Constanţa, Romania
E-mail: dsburlan@univ-ovidius.ro

Abstract. In this paper we address the possibility of studying the computational capabilities of catalytic P systems with one catalyst by the means of iterated finite state transducers (IFT). In particular, we focus on a variant of catalytic P systems which satisfies in addition to the original definition the following property: if in a region of the P system an object is the subject of a catalytic rule, then in the same region it must exists also a non-cooperative rule that rewrites the same object. Finally, we also give a normal form for catalytic P systems.

1. Introduction

P systems are a computational model introduced by G. Păun in [3]. One of the basic variant considered there was P systems with catalysts and priorities; these systems were shown to be computationally universal. In [2], it is proved a stronger result, that is, priorities among the rules can be discarded from the model without any loss of computational power. Moreover, it was shown that for extended P systems only one membrane and two catalysts are enough for reaching computational universality. However, the computational power for P systems with one catalyst was not established, the common belief being that such systems are not Turing universal. Here we explore a particular case of this problem, namely we introduce a restricted variant of catalytic P systems and we characterize these systems in terms of iterated finite state transducers. Studying this variant by means of IFT's gives us the opportunity to use novel tools in the attempt to tackle the above mentioned conjecture. Additionally, a normal form for catalytic P systems is presented.

2. Preliminaries

We assume the reader is acquainted with the basic notions and notations from the formal language theory (see [6] for more details). Here we only recall the definitions and the results which are useful for the present work.

If FL is a family of languages, then NFL denotes the family of length sets of languages in FL. We denote by REG, CF, REC, and RE the family of regular, context-free, recursive, and recursively enumerable languages, respectively. It is known that $NREG = NCF \subsetneq NREC \subsetneq NRE$.

2.1. Iterated Finite State Transducers

An *iterated (finite state) sequential transducer* (IFT) is a tuple $\gamma = (K, V, q_0, a_0, F, P)$, where K is a finite set of *states*, V is a finite set of symbols (the *alphabet* of γ), $K \cap V = \emptyset$, $q_0 \in K$ is the *initial state*, $a_0 \in V$ is the *starting symbol*, $F \subseteq K$ is the set of *final states*, and P is a finite set of *transition rules* of the form $qa \to xp$, for $q, p \in K$, $a \in V$, and $x \in V^*$.

For $q, p \in K$ and $u, v, x \in V^*$, $a \in V$, a *direct transition* step of γ is defined as $uqav \vdash uxpv$ if and only if $qa \to xp \in P$. The reflexive and transitive closure of the relation $\vdash$ is denoted by $\vdash^*$. In general, for $\alpha, \beta \in V^*$ we say that α *derives* into β and we write $\alpha \Longrightarrow \beta$, if and only if $q_0\alpha \vdash^* \beta p$ for some $p \in K$. By $\Longrightarrow^*$ we denote the reflexive transitive closure of $\Longrightarrow$. If $q_0\alpha \vdash^* \beta p$ such that $p \in F$, then we write $\alpha \Longrightarrow_f \beta$. The language generated by γ is $L(\gamma) = \{\beta \in V^* \mid a_0 \Longrightarrow^* \alpha \Longrightarrow_f \beta, \text{ for some } \alpha \in V^*\}$.

If for each pair $(q, a) \in K \times V$, there is at most one transition rule $qa \vdash xp \in P$, then γ is called *deterministic* (otherwise, it is called *nondeterministic*). The family of languages generated by nondeterministic IFTs with at most $n \geq 1$ states is denoted by IFT_n. It is known from [1] that $CF \subset IFT_2 \subseteq IFT_3 \subseteq IFT_4 = RE$. Moreover, there are non-semilinear languages belonging to IFT_2, and there are non-recursive languages belonging to IFT_3. Consequently, if we denote by $NIFT_n$, $n \geq 1$, the family of length sets of languages from IFT_n, then we have that $NREG = NCF \subsetneq NIFT_2 \subseteq NIFT_3 \subseteq NIFT_4 = NRE$.

2.2. Membrane Systems

A *catalytic P system* of degree $m \geq 1$ is a construct

$$\Pi = (O, C, \mu, w_1, \ldots, w_m, R_1, \ldots, R_m, i_0)$$

where

- O is an alphabet of *objects*;
- $C \subseteq O$ is the set of *catalysts*;
- μ is a hierarchical tree structure of $m \geq 1$ uniquely labelled *membranes* (which delimit the regions of Π); typically, the set of labels is $\{1, \ldots, m\}$;
- $w_i \in O^*$, for $1 \leq i \leq m$, are the multisets of objects initially present in the m regions of μ;

• R_i, $1 \leq i \leq m$, are finite sets of *evolution rules*; the rules can be *non-cooperative* $a \to v$ or *catalytic* $ca \to cv$, where $a \in O \setminus C$, $v \in ((O \setminus C) \times \{here, out, in\})^*$, and $c \in C$;

• $i_0 \in \{1, \ldots, m\}$ is the label of the *output region* of Π.

The P system is called *mixed* if it satisfies the following property: if there exists a rule $ca \to c\alpha \in R_i$, then there must exists in the same region i at least one non-cooperative rule handling object a, i.e., of type $a \to \beta \in R_i$.

A *configuration* of Π is a vector $C = (\alpha_1, \ldots, \alpha_m)$, where $\alpha_i \in O^*$, $1 \leq i \leq m$, is a multiset of objects present in the region i of Π. The vector $C_0 = (w_1, \ldots, w_m)$ is the *initial configuration* of Π. Starting from the initial configuration and always applying in all membranes a maximal multiset of evolution rules in parallel, one gets a sequence of consecutive configurations. By $\Rightarrow$ is denoted the *transition* between two consecutive configurations. A sequence (finite or infinite) of transitions starting from C_0 represents a *computation* of Π. A computation of Π is a halting one if no rules can be applied in the last configuration (the *halting configuration*). The result of a halting computation is the number of objects from O contained in the output region i_0, in the halting configuration. A non-halting computation yields no result. By collecting the results of all possible halting computations of a given P system Π, one gets $N(\Pi)$ – the set of all natural numbers generated by Π.

The family of all sets of numbers computed by [mixed] catalytic P systems with at most m membranes and k catalysts is denoted by $NOP_m([M]cat_k)$. The above definition can be relaxed such that in a halting configuration one counts only the symbols from a given alphabet $\Sigma \subseteq O$. In particular, one can consider $\Sigma = O \setminus C$; correspondingly, the family of all sets of numbers computed by such particular P systems will be denoted by $NO_{-C}P_m([M]cat_k)$.

It is known (see [7], for instance) that $NO_{-C}P_m([M]cat_k) = NO_{-C}P_1([M]cat_k)$. Moreover, in [2] it is shown that $NO_{-C}P_1(cat_2) = NRE$.

Since mixed catalytic P systems are particular cases of catalytic P systems then we also have that $NO_{-C}P_m(Mcat_k) \subseteq NO_{-C}P_1(cat_k)$.

3. A Normal Form for P Systems with Catalysts

The following result states that any catalytic P system is equivalent with a catalytic P system having a restriction on the form of the rules. The result recalls the Chomsky Normal Form for context-free grammars.

Theorem 1. *For any P system Π with catalysts there exists an equivalent P system $\overline{\Pi}$ with one region and whose rules are of the form $a \to \alpha$, with $|\alpha| \leq 2$, or $ca \to c\beta$, with $|\beta| \leq 1$.*

Proof. As we already stated in Section **??**, for any P system with catalysts and $n > 1$ membranes one can construct an equivalent P system with the same number of catalysts and one membrane. Consequently, without loss of generality, we might assume that Π has only one membrane, that is $\Pi = (O, C, [\,]_1, w_1, R_1, i_0 = 1)$.

Let $O \setminus C = \{a_1, a_2, \ldots, a_p\}$ and let $m = max\{|\alpha| \mid a \to \alpha \in R_1 \text{ or } ca \to c\alpha \in R_1\}$.

In addition, assume for our convenience that the rules of Π are labeled in an unique manner with numbers from the set $\{1, \ldots, card(R_1)\}$.

Then one can construct an equivalent P system $\overline{\Pi} = (\overline{O}, C, [\ ]_1, w_1, \overline{R_1}, i_0 = 1)$ where

$$\begin{aligned} \overline{O} &= O \cup \{a_{(i,j)} \mid 1 \leq i \leq p, 1 \leq j \leq m\} \\ &\cup \{X_{(i,j)} \mid i : a \to \alpha_i \in R_1, 1 \leq j \leq m-2\}. \end{aligned}$$

The set $\overline{R_1}$ is defined as follows (for the simplicity of the explanations, we will only consider the rules in $\overline{R_1}$ that are useful for simulating a non-cooperative rule from R_1; the rules corresponding to a catalytic rule are defined similarly, therefore we will not present them here). Let $i : a \to a_{j_1} a_{j_2} \ldots a_{j_k} \in R_1$ and let $t = m - k$. Then we add to $\overline{R_1}$ the rules:

$$\begin{aligned} a &\to X_{(i,1)} \\ X_{(i,1)} &\to X_{(i,2)} \\ &\ldots \\ X_{(i,t-1)} &\to X_{(i,t)} \end{aligned} \tag{1}$$

$$\begin{aligned} X_{(i,t)} &\to a_{(j_1,k-1)} X_{(i,t+1)} \\ X_{(i,t+1)} &\to a_{(j_2,k-2)} X_{(i,t+2)} \\ &\ldots \\ X_{(i,t+k-3)} &\to a_{(j_{k-2},2)} X_{(i,t+k-2)} \\ X_{(i,t+k-2)} &\to a_{(j_{k-1},1)} a_{(j_k,1)} \end{aligned} \tag{2}$$

$$\begin{aligned} a_{(i,m)} &\to a_{(i,m-1)} \\ a_{(i,m-1)} &\to a_{(i,m-2)} \\ &\ldots \\ a_{(i,1)} &\to a_i \end{aligned} \tag{3}$$

The proof is based on the existence of the universal global clock that governs the functioning of the P system (the clock marks equal time units for the whole system, hence synchronization is possible). While trying to simulate the application of an arbitrary non-cooperative rule with several rules of type $a \to \alpha$, with $|\alpha| \leq 2$, one has to accomplish two conditions. Firstly, one has to guarantee that all the objects from α will eventually be produced. Secondly, these objects must be produced at the "proper" time: all of them in the same moment (a local synchronization) and according with the simulation of other rules that were started at the same time with $a \to \alpha$ (a global synchronization).

Consequently, the rules presented above are grouped according with their function in the simulation. The first group represents a set of "delaying" rules (they are used while simulating the rules with a shorter right hand side in order to synchronize

their executions with those that have the longest right hand side). These rules are "chained", hence, staring from an object a, an object $X_{(i,t)}$ is produced in exactly t computational steps. The second group is responsible for producing in consecutive computational steps the objects $a_{(j_1,k-1)}, a_{(j_2,k-2)}, \ldots, a_{(j_{k-1},1)}, a_{(j_k,1)}$ (in order of their production, the last two being produced in the same time). For an object $a_{(i,l)}$ in this sequence, the index l represents the number of computational steps that $\overline{\Pi}$ will perform, starting from its production and until the object a_i is produced (see the third group of rules). Finally, one can remark that the objects $a_{j_1}, a_{j_2}, \ldots, a_{j_k}$ are produced in the same computational step by $\overline{\Pi}$ (while simulating the rule $i : a \to a_{j_1}a_{j_2}\ldots a_{j_k} \in R_1$). Moreover, all the other rules from Π that started at the same moment as $i : a \to a_{j_1}a_{j_2}\ldots a_{j_k}$, are simulated in the same manner by $\overline{\Pi}$ and their output is produced in the same computational step as mentioned above. Consequently $\overline{\Pi}$ correctly simulates any computation of Π, hence the statement holds true. □

4. Mixed Catalytic P Systems with One Catalyst and IFTs

In what follows we prove that the family of sets of numbers computed by mixed catalytic P systems with only one catalyst is included in the family of the length sets of the languages generated by iterated finite state transducers with at most 3 states.

Theorem 2. $NIFT_3 \supseteq NOP_1(Mcat_1)$.

Proof. Given an arbitrary mixed catalytic P system $\Pi = (O, C, w_1, R_1, i_1)$ such that $C = \{c\}$, then one can construct an iterated finite transducer $\gamma = (K, V, q_0, a_0, F, P)$ which simulates Π as follows.

Without loss of generality we assume that the initial configuration of Π is $w_1 = ca_0$.

In addition, let us consider the following sets of objects from O:

$X = \{A \in O \mid (\exists)\ A \to \alpha \in R_1 \text{ and } (\nexists)\ cA \to c\beta \in R_1\}$;

$Y = \{A \in O \mid (\exists)\ A \to \alpha \in R_1 \text{ and } cA \to c\beta \in R_1\}$;

$T = \{A \in O \mid (\nexists)\ A \to \alpha \in R_1 \text{ and } (\nexists)\ cA \to c\beta \in R_1\}$.

One can remark that $O = X \cup Y \cup T \cup \{c\}$.

Based on the above settings the IFT γ is defined as follows:

$$\begin{aligned} K &= \{q_0, q_1, q_2\}, \\ V &= O \setminus \{c\}, \\ F &= \{q_0\}, \end{aligned}$$

and the set of rules P is constructed in the following manner:

- for any $a \in T$ we add to P the rule $q_0a \to aq_0$;
- for any $a \in X \cup Y$ and $a \to \alpha \in R_1$ we add to P the rules $q_0a \to \alpha q_1$;
- for any $a \in T$ we add to P the rule $q_1a \to aq_1$;
- for any $a \in X \cup Y$ and $a \to \alpha \in R_1$ we add to P the rules $q_1a \to \alpha q_1$;
- for any $a \in Y$ and $ca \to c\alpha \in R_1$ we add to P the rules $q_1a \to \alpha q_2$;

- for any $a \in T$ we add to P the rule $q_2 a \to a q_2$;
- for any $a \in X \cup Y$ and $a \to \alpha \in R_1$ we add to P the rules $q_2 a \to \alpha q_2$;
- for any $a \in Y$ and $ca \to c\alpha \in R_1$ we add to P the rules $q_0 a \to \alpha q_2$.

For a simpler understanding of the proof we gave in Figure 1 the representation of γ as a finite state automaton with output.

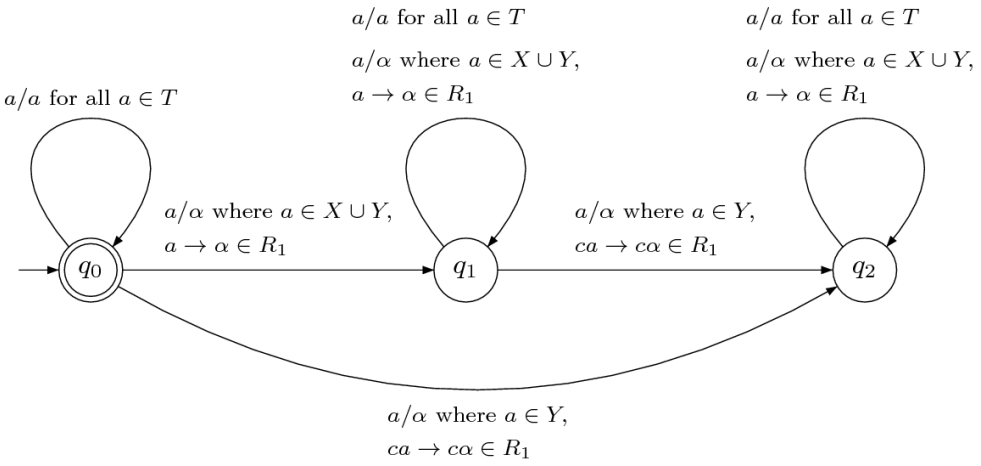

Fig. 1. The representation of γ as a finite state automaton with output, simulating a mixed catalytic P system with one catalyst.

The construction was designed such that each string processed by γ during its computation will correspond to a configuration of Π. Moreover, one iteration of γ simulates the maximal parallel applications of the rules of Π.

If the current string (say w) processed by the IFT is composed only by the symbols from T, then γ remains in $q_0 \in F$ and stops, accepting the string. This situation corresponds to the halting configuration of Π (that is, Π contains in its region the multiset cw and no rules can be further applied).

In case w contains symbols form $X \cup Y$, then γ starts the simulation of the maximal parallel applications of the rules of Π. Since γ processes strings at each iteration, then the simulation of Π has to accomplish the following task: all the symbols which are the subject of a rule of Π have to be processed also by γ. Recall that γ processes strings and in these strings there might be symbols from T (which are not the subject of any rule) in any position. Consequently, one has to be sure that any symbol in a configuration of Π that is a subject of a rule (non-cooperative or catalytic) has to have the opportunity to be rewritten in the corresponding string processed by γ (by the corresponding rule). This is why, γ uses the rules $q_i a \to a q_i$ for $q_i \in Q$, $1 \leq i \leq 3$, and $a \in T$ (that is, while processing the string, γ "skips" all the symbols that are not the subject of any rule).

In one iteration of γ one can apply at most once a rule corresponding to a catalytic rule of Π (recall that the P system functioning semantics define such behaviour). More precisely, assuming that w is the current processed string, we have

- either γ is in state q_0 and executes a rule of type $q_0a \to \alpha q_2$ for $q_0, q_2 \in K$, $a \in Y$, and $ca \to c\alpha \in R_1$. This situation occurs when γ processes $w = w_1aw_2$, $w_1 \in T^*$, and $w_2 \in (X \cup Y \cup T)^*$ (w has the prefix w_1 composed only by symbols from T, followed by the symbol $a \in Y$ that triggers the simulation of the catalytic rule; the symbols from w_2 that belong to $X \cup Y$ will trigger only the simulation of the non-cooperative rules).

- or γ is in state q_1 and executes a rule of type $q_1a \to \alpha q_2$ for $q_1, q_2 \in K$, $a \in Y$, and $ca \to c\alpha \in R_1$). This situation occurs when γ processes $w = w_1aw_2$, where w_1 is described by the regular expression $T^*(X|Y)(X|Y|T)^*$, $w_2 \in (X \cup Y \cup T)^*$ (the symbols from w_1 and w_2 that belong to $X \cup Y$ will trigger only the simulation of the non-cooperative rules).

The IFT γ simulates Π by processing strings (hence the order of symbols is precisely defined). The design of γ guarantees that, if applicable, a rule corresponding to a catalytic rule of Π is executed at most once. In addition, γ simulates the maximal parallel application of the rules of Π. Consequently, one can put in a bijective correspondence the configurations of Π obtained during a computation with the strings processed by γ.

□

Based on the above theorem, the following result holds true.

Corollary 3. *If $NIFT_3 \subset NRE$ then $NOP_1(Mcat_1) \subset NRE$.*

Remark 4.1. The construction presented in Theorem 2 uses an IFT γ with 3 states to simulate an arbitrary mixed catalytic P system. We conjecture that the number of states of γ can be decreased to 2. If one replaces the rules of type $q_ia \to \alpha q_j$ by the set of rules $q_ia \to Perm(\alpha)q_j$, where by $Perm(\alpha)$ one understands the set of all permutations of string α, then one can perform a "convenient" local "shuffle" of the string processed by γ such that any symbol that is the subject of a catalytic rule actually has the chance to be rewritten by such a rule. By doing this one can avoid the usage of state q_1 in the definition of γ.

We also conjecture that IFT's with 3 states can be used to simulate arbitrary catalytic P systems. In this respect one might use an IFT defined similarly as the one presented in Theorem 2 (for a given P system Π, if $Z = \{A \in O \mid (\exists)\ cA \to c\alpha \in R_1$ and $(\nexists)\ A \to \beta \in R_1\}$ then one can construct an IFT γ as described in Figure 2). However, in this case, assuming that the current string that has to be processed by γ is of type $w = w_1A_1w_2A_2 \dots w_nA_nw_{n+1}$ with $A_i \in Z$, $1 \leq i \leq n$, and $w_j \in (X \cup Y \cup T)^*$, $1 \leq j \leq n+1$ (i.e., w contains several symbols from Z), then it follows that symbols $A_2, \dots, A_n$ have no chance to be rewritten in the current step by a rule of γ which corresponds to a catalytic rule of Π (simulation which contradicts the functioning of catalytic P systems). For example, if no object from w_1 forces γ to change the state from q_1 to q_2 (or from q_0 to q_2), then γ is obliged to enter q_2 while processing the symbol $A_1 \in Z$; consequently, the rest of the objects from w will be processed in state q_2, hence one can notice that $A_2, \dots, A_n$ do not have the

same chance to be rewritten as A_1. It remains an open problem whether or not this restriction is important for the overall simulation.

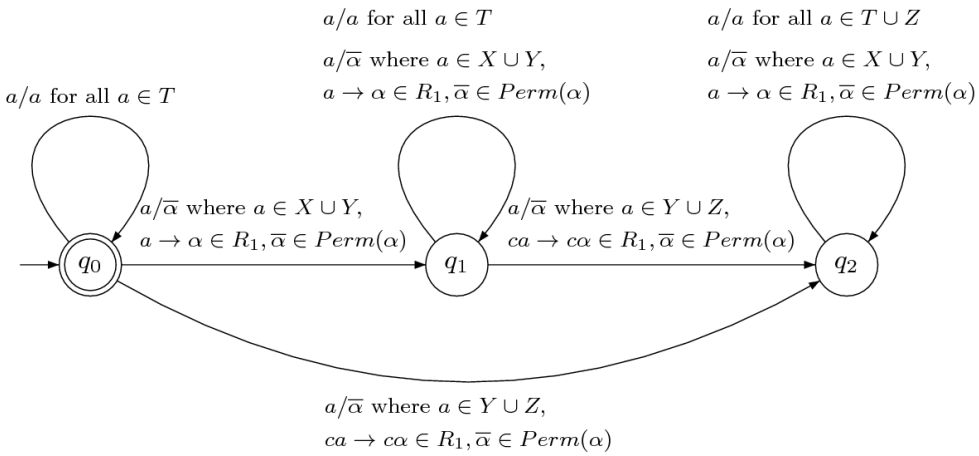

Fig. 2. The representation of γ as a finite state automaton with output, in the attempt of simulating a catalytic P system with one catalyst.

5. Conclusions

In this paper we gave a normal-form theorem for catalytic P systems. We also investigated the relation between mixed P systems with one catalyst and iterated finite transducers. This last topic is of a particular interest because it converts an open problem from the P system framework to an open problem from the string rewriting theory. In addition, the simplicity of the construction (as presented in Figure 1) gives hopes for establishing the computational power of catalytic P systems.

Acknowledgements. Thanks to anonymous reviewers for several suggestions on the previous version of the paper. The work of the author was supported by the CNCSIS IDEI-PCE grant, no. 551/2009, Romanian Ministry of Education, Research and Innovation.

References

[1] Bordihn H., Fernau H., Holzer M., Manca V., Martin-Vide C., *Iterated Sequential Transducers as Language Generating Devices*, Theoretical Computer Science, **369**, 1 (2006), pp. 67–81.

[2] Freund R., Kari L., Oswald M., Sosik P., *Computationally Universal P Systems Without Priorities: Two Catalysts Are Sufficient*, Theoretical Computer Science, **330**, 2 (2005), pp. 251–266.

[3] Păun G., *Computing with Membranes*, J. Computer and System Sci., **61** (2000), pp. 108–143.

[4] Păun G., *Membrane Computing. An Introduction*, Springer-Verlag, Berlin, 2002.

[5] Păun G., Rozenberg G., Salomaa A., eds., *The Oxford Handbook of Membrane Computing*, Oxford University Press Inc., New York, 2010.

[6] Rozenberg G., Salomaa A., eds., *Handbook of Formal Languages*, 3 volumes, Springer-Verlag, Berlin, 1997.

[7] Sburlan D., *Further Results on P Systems with Promoters/Inhibitors*, International Journal of Foundations of Computer Science, **17** (2006), pp. 205–221.