Non-cooperative Polymorphic P Systems and Parallel Communicating ET0L Systems*

Anna Kuczik and György Vaszil

Faculty of Informatics, University of Debrecen
Kassai út 26, 4028 Debrecen, Hungary
kuczik.anna@inf.unideb.hu
vaszil.gyorgy@inf.unideb.hu

Abstract. We demonstrate that languages of non-cooperative polymorphic P systems can be generated by nonreturning parallel communicating ET0L systems, and discuss a further topic for research, namely the identification of the subclass of parallel communicating ET0L systems for which the converse of the statement holds, that is, the precise characterization of languages of non-cooperative polymorphic P systems in terms of the parallel communicating Lindenmayer system model.

1 Introduction

Polymorphic P systems were introduced in[1], they represent a membrane system model where the multiset rewriting rules associated to the regions are not fixed, but dynamically deduced from the contents of certain regions of the systems which dynamically change during the computational process. The study of the non-cooperative variant was started in [2]. A polymorphic P system is non-cooperative, if the left-hand sides of the rules always contain at most one symbol, that is, a multiset with at most one element.

Non-cooperative systems with so called finitely representable regions were investigated in [3]. A region is finitely representable (FIN-representable in short), if the set of possible multisets that can appear as the contents of the region in question during any computation is finite. In [3] we have shown that languages of non-cooperative polymorphic systems where all regions (besides the skin region) are FIN-representable coincide with the Parikh sets of languages of ET0L systems.

Here we consider the general, not necessarily finitely representable case, and establish a relationship of the languages of such P systems and parallel communicating ET0L systems.

* Supported by the University of Debrecen Scientific Research Bridging Fund (DETKA).

2 Preliminaries and Definitions

In this section, we define the basic definitions and notions we will use. For more information about formal language theory, see [7], and [5, 6] for details about membrane computing.

An *alphabet* V is a finite non-empty set of symbols. A string (or word) over V is a finite sequence of elements of V, the set of all strings over V is denoted by V^*, and $V^+ = V^* \setminus \{\lambda\}$ where λ denotes the empty string. For a string $w \in V^*$, we denote by $|w|_S$ the number of occurrences of the letters $x \in S \subseteq V$ in w, but if $S = \{x\}$ is a singleton, we write $|w|_x$ instead of $|w|_{\{x\}}$. If we fix an order $V = \{a_1, a_2, \ldots, a_n\}$ of the letters, then the vector $(|w|_{a_1}, |w|_{a_2}, \ldots, |w|_{a_n})$ is called the Parikh vector of the word $w \in V^*$.

If $\mathbb{N}$ denotes the set of nonnegative integers, then a *multiset* over a set U is a mapping $M : U \to \mathbb{N}$ where $M(a)$, for all $a \in U$, is the multiplicity of a in the multiset M. If U is finite, $U = \{a_1, a_2, \ldots a_n\}$, then M can also be represented by a string $w = a_1^{M(a_1)} a_2^{M(a_2)} \ldots a_n^{M(a_n)}$ (and all permutations of this string) where a^j denotes the string obtained by concatenating $j \in \mathbb{N}$ occurrences of the letter $a \in V$ (with $a^0 = \lambda$).

Lindenmayer systems, of L systems are parallel rewriting mechanisms, see [4, 8] for more information. In the following, we define ET0L systems, which are extended, tabled, and interactionless versions of L systems.

An *ET0L system* is a quadruple $G = (V, T, U, \omega)$ where V is an alphabet, $T \subseteq V$ is a terminal alphabet, $\omega \in V^+$ is the initial word of G, and $U = (P_1, \ldots, P_m)$ where $P_i,\ 1 \leq i \leq m$, are finite sets of context-free productions over V (called *tables*), such that for each $a \in V$, there is at least one rule $a \to w,\ w \in V^*$ in each table.

In each computational step, G rewrites all the symbols of the current sentential form in parallel with the rules of one of the tables in U, this choice is non-deterministic. The language generated by G consists of all terminal strings which can be generated by a series of rewriting steps (computational steps), that is, by a derivation, starting from the initial word.

A *parallel communicating ET0L system* (PC ET0L system, in short) with n components is a (n+3) tuple $\Gamma = (N, K, T, G_1, \ldots, G_n)$, where N is a nonterminal alphabet, T is a terminal alphabet and K is an alphabet of query symbols ($K = \{K_1, \ldots, K_n\}$). The components are ET0L systems, $G_i = (N \cup K \cup T, T, U_i, \omega_i)$ with nonterminal and terminal alphabets as above, a table U_i of rewriting rules $U_i = \{P_{i,1}, \ldots, P_{i,k_i}\}$, and an axiom $\omega_i \in (N \cup T)^*$, $1 \leq i \leq n$. One of the components $G_m,\ m \in \{1, \ldots, n\}$, is called the master grammar of Γ.

An n-tuple $(x_1, \ldots, x_n)$, where $x_i \in (V \cup K)^*,\ \ 1 \leq i \leq n$, is called a configuration of Γ.

Let Γ be a PC ET0L system as above, and let $(x_1, \ldots, x_n)$, $(y_1, \ldots, y_n)$ be two configurations of Γ. A direct derivation step denoted by $(x_1, \ldots, x_n) \Rightarrow (y_1, \ldots, y_n)$, is defined as follows.

1. There is no x_i which contains any query symbol; that is, $x_i \in V^*$ for $1 \leq i \leq n$. Then for each i, $1 \leq i \leq n$, $x_i \Rightarrow_{G_i} y_i$ (y_i is obtained from x_i by a direct derivation step in G_i).

2. There is some $x_i,\ 1 \le i \le n$, which contains at least one occurrence of a query symbol. In this case $(y_1, \dots, y_n)$ is obtained from $(x_1, \dots, x_n)$ as follows:

For each x_i with $|x_i|_K \neq 0$ we write $x_i = z_1 K_{i_1} z_2 K_{i_2} \dots z_t K_{i_t} z_{t+1}$, where $z_j \in V^*,\ 1 \le j \le t+1$, and $K_{i_l} \in K,\ 1 \le l \le t$. If $|x_{i_l}|_K = 0$ for each $i_l,\ 1 \le l \le t$, then $y_i = z_1 x_{i_1} z_2 x_{i_2} \dots z_t x_{i_t} z_{t+1}$ and $y_{i_l} = x_{i_l},\ 1 \le l \le t$. If $|x_{i_l}|_K \neq 0$ for some i_l, $1 \le l \le t$, then $y_i = x_i$. For all $j,\ 1 \le j \le n$, for which y_j is not specified above, $y_j = x_j$.

Remark 1. We have only gave the definition of the so-called *non-returning* mode of communication of PC ET0L systems in the previous paragraph, since this is the variant that we will need in this paper.

The *language* generated by a PC ET0L system $\Gamma = (N, K, T, G_1, \dots, G_n)$, where $G_i = (N \cup K \cup T, T, P_i, \omega_i)$, $1 \le i \le n$, is

$$L(\Gamma) = \{\alpha_m \in T^* | (\omega_1, \dots, \omega_n) \Rightarrow^* (\alpha_1, \dots, \alpha_n)\}$$

where $G_m,\ m \in \{1, \dots, n\}$, is the master grammar of Γ, and $\Rightarrow^*$ is the reflexive and transitive closure of $\Rightarrow$.

We are not interested in the character string generated by the ET0L system as a sequence of letters, but only in the multiples of the different letters, i.e. the Parikh vectors of the words. This is necessary because we will connect the ET0L languages to the multiset languages of the P systems. We denote by $Ps(\Gamma)$ the set of Parikh vectors corresponding to the strings of $L(\Gamma)$ (Parikh set of $L(\Gamma)$), and by $PsNPC(ET0L)$ the class of Parikh sets corresponding to the class of languages generated by PC ET0L systems.

Example 1. Consider a PC ET0L grammar system

$$\Gamma = (N, K, T, G_1, G_2)$$

with $N = \emptyset$, $K = \{K_1, K_2\}$, $T = \{a\}$ and $G_i = (N \cup K \cup T, T, U_i, \omega_i),\ 1 \le i \le 2$, where

$$\omega_1 = a,$$
$$U_1 = \{P_{1,1}\} \text{ with } P_{1,1} = \{a \to K_2\},$$

and

$$\omega_2 = aa,$$
$$U_2 = \{P_{2,1}, P_{2,2}\} \text{ with } P_{2,1} = \{a \to a\}, P_{2,2} = \{a \to aa\}.$$

The derivations of Γ start in the configuration $(\omega_1, \omega_2) = (a, aa)$ and the first rewriting step results in (K_2, w_2) where K_2 is a query symbol requesting the sentential form of the second component, and w_2 is either aa or $aaaa$, depending on the table that was used by G_2. In short, we have

$$(a, aa) \Rightarrow (K_2, a^{i_1}) \Rightarrow (a^{i_1}, a^{i_1})$$

where $i_1 \in \{2, 4\}$, and the last step is a communication step. Continuing the derivation we get

$$(a^{i_1}, a^{i_1}) \Rightarrow ((K_2)^{i_1}, a^{i_2}) \Rightarrow ((a^{i_2})^{i_1}, a^{i_2})$$

where $i_2 = i_1$ or $i_2 = 2 \cdot i_1$ depending again on the table used by G_2. If in each rewriting step the second component uses its first table, then the number of symbols in the sentential form of the first component doubles after each communication. If the second table is used by G_2 in each rewriting step, then the number of symbols is four times as high after the second step, as it was after the first step, eight times as high after the third step as after the second, and so on, reaching $2^{\frac{n\cdot(n+1)}{2}}$ after the nth step. If the table choice of G_2 varies between these these possibilities, then the number of symbols is somewhere in between the two extremes. After the nth communication step (which is directly following the nth rewriting step), we have a configuration

$$(a^{(2^m)}, a^{i_n}),\ \text{where}\ n \leq m \leq \frac{n \cdot (n+1)}{2},$$

thus, if G_1 is the master component, then

$$L(\Gamma) = \{a^m \mid 2^n \leq m \leq 2^{\frac{n\cdot(n+1)}{2}} \text{ for some } n \geq 0\}.$$

Polymorphic membrane systems were introduced in [1]. The rules in polymorphic P systems are defined by the contents of specific membrane regions corresponding to the left- and right-hand sides of the rule. As a result, the rules belonging to the regions change(s) during the computation. These rules are called dynamic rules.

A *polymorphic P system* is a tuple

$$\Pi = (O, T, \mu, w_s, \langle w_{1L}, w_{1R}\rangle, \ldots, \langle w_{nL}, w_{nR}\rangle, h_o),$$

where O is the alphabet of objects, $T \subseteq O$ is the set of terminal objects, μ is the membrane structure consisting of $2n+1$ membranes labelled by a symbol from the set $H = \{s, 1L, 1R, \ldots, nL, nR\}$, the elements of the multiset w_s are the initial contents of the skin membrane, the pairs of multisets $\langle w_{iL}, w_{iR}\rangle$ correspond to the initial contents of membranes iL and iR, $1 \leq i \leq n$, and $h_o \in H$ is the label of the output membrane.

The rules of a polymorphic membrane system are not given statically in the initial configuration. In each step, they are dynamically derived based on the contents of the left and right (iL and iR, $1 \leq i \leq n$) membrane pairs. Thus, if the membranes iL and iR belonging to the ith membrane pair contain multisets u and v respectively, then in the next step the contents of their parent membrane is transformed as if the multiset rewriting rule $u \to v$ were present.

If there is at least one rule in a system Π where the number of objects in u (the multiset on the left-hand side) can grow to be greater than one, then we say that Π is a *cooperative* system, otherwise, it is a *non-cooperative* system.

The set of vectors $Ps(\Pi)$ generated by a polymorphic P system Π with the terminal alphabet $T \subseteq O$ is the set of vectors representing the multisets of the terminal objects appearing in the output region h_o in a halting configuration of Π which is reached by a computation starting in the initial configuration of the system. The class of vector languages generated by non-cooperative polymorphic P systems is denoted by $PsOP(polym, ncoo)$.

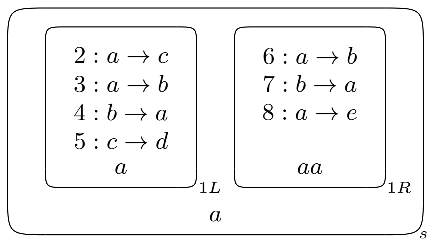


Fig. 1: The initial configuration of the polymorphic P system Π_2 of Example 2.

Remark 2. If the regions labeled by hL and hR for some $h \in \{1, \ldots, n\}$ are elementary regions, then the rule represented by their contents $\langle w_{hL}, w_{hR} \rangle$ stays the same during the whole computation (it is a non-dynamic rule or static rule). In what follows we sometimes use a simplified notation $r_h : w_{hL} \to w_{hR}$ in this case.

Example 2. Consider the polymorphic P system

$$\Pi_2 = (O, T, \mu, w_s, \langle w_{1L}, w_{1R} \rangle, \ldots, \langle w_{8L}, w_{8R} \rangle, s)$$

where $O = T = \{a, b, c, d, e\}$, and the membrane structure is $\mu = [\,[\ldots]_{1L}\,[\ldots]_{1R}\,]_s$, where the child membranes of 1L are $[\,]_{2L}\,[\,]_{2R} \ldots [\,]_{5L}\,[\,]_{5R}$ and the children of 1R are $[\,]_{6L}\,[\,]_{6R} \ldots [\,]_{8L}\,[\,]_{8R}$. Let

$$w_s = a, \; w_{1L} = a, \; w_{1R} = aa,$$

and using the simplified notation for static rules, let the rules applicable in 1L be

$$r_2 : a \to c, \; r_3 : a \to b, \; r_4 : b \to a, \; r_5 : c \to d,$$

and the rules applicable in 1R be

$$r_6 : a \to b, \; r_7 : b \to a, \; r_8 : a \to e,$$

as can also be seen in Figure 1.

3 FIN-representable Subsystems of Polymorphic Systems

The idea of FIN-representability was introduced in [3]. For a polymorphic P system $\Pi = (O, T, \mu, w_s, \langle w_{1L}, w_{1R}\rangle, \ldots, \langle w_{nL}, w_{nR}\rangle, h_o)$, let $w_{j,h}$ for some $h \in \{s, 1L, 1R, \ldots, nL, nR\}$ denote the multiset contained by region h after the jth step of the computation of Π for some $j \geq 0$. We call $w'_{j,h}$ an element of the *successor set* of $w_{j,h}$, denoted as $w'_{j,h} \in \sigma_{j,h}(w_{j,h})$, if $w'_{j,h}$ can be obtained from $w_{j,h}$ by the rules associated to the region h, as determined by the configuration of Π which is reached in the jth step of the computation.

If for the same $w_{j,h}$ as above, we fix $\sigma^0_{j,h}(w_{j,h}) = \{w_{j,h}\}$ for any $j \geq 0$, and for $k \geq 0$ we have $\sigma^{k+1}_{j,h} = \sigma_{j+k,h}(\sigma^k_{j,h}(w_{j,h}))$ (where we extend the range of the function from multisets to sets of multisets in the natural way), then we can define

$$\sigma^*_{j,h} = \bigcup_{k \geq 0} \sigma^k_{j,h}(w_{j,h}).$$

A region h of Π is *finitely representable* or FIN-representable in short, if the set of successor multisets of the initial contents of h, w_h, is finite, that is, $\sigma^*_{0,h}(w_h)$ is finite.

We say that a finite transition system $M = (Q, q_0, \delta)$ *represents the rule configurations* of a FIN-representable polymorphic P system Π if the contents of the pairs of membranes labelled by $1L, 1R, \ldots, nL, nR$ of the configuration sequences of $\Pi = (O, T, \mu, w_s, \langle w_{1L}, w_{1R}\rangle, \ldots, \langle w_{nL}, w_{nR}\rangle, s)$

$$(w_s, w_{1L}, w_{1R}, \ldots, w_{nL}, w_{nR}) = (w^0_s, w^0_{1L}, w^0_{1R}, \ldots, w^0_{nL}, w^0_{nR}) \Rightarrow$$
$$\Rightarrow (w^1_s, w^1_{1L}, w^1_{1R}, \ldots, w^1_{nL}, w^1_{nR}) \Rightarrow \ldots \Rightarrow (w^i_s, w^i_{1L}, w^i_{1R}, \ldots, w^i_{nL}, w^i_{nR}) \Rightarrow \ldots$$

are in a one-to-one correspondence with the state sequences

$$q_0 = q^0 \Rightarrow q^1 \Rightarrow \ldots \Rightarrow q^i \Rightarrow \ldots$$

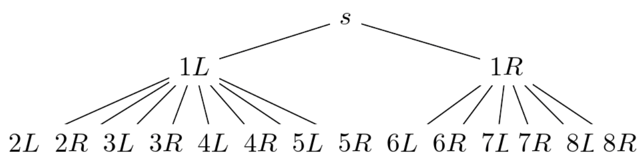
of the transition system M. Moreover, $q^h,\ h \geq 0$, is a halting state of M, if and only if no rules are applicable in any of the regions $1L, 1R, \ldots, nL, nR$ in the corresponding configuration $(w^h_s, w^h_{1L}, w^h_{1R}, \ldots, w^h_{nL}, w^h_{nR})$ of Π.

Based on this correspondence, the configuration sequences of Π can be abbreviated as the sequence

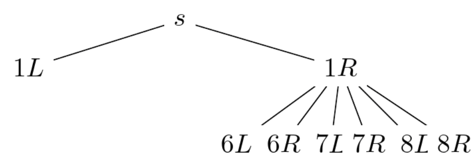
$$(w_s, q_0) = (w^0_s, q^0) \Rightarrow (w^1_s, q^1) \Rightarrow \ldots \Rightarrow (w^i_s, q^i) \Rightarrow \ldots$$

combining the contents of the (not necessarily finitely representable) skin region and the current state of the finite representation of the rest of the system.

In [3] we have shown how to construct such a transition system for any FIN-representable membrane system. Polymorphic P systems are not FIN-representable in general, but we can still use finite transition systems to represent FIN-representable regions. Since left-hand membranes of non-cooperative polymorphic systems (the membranes labeled with $iL,\ 1 \leq i \leq n$) may contain at



(a) The membrane structure μ of the P system Π_2.



(b) The membranes with labels iL, iR for $i \in trim(\mu) = \{1, 6, 7, 8\}$ in the membrane structure μ of the P system Π_2. Also, $lchild(s) = \{1\}$, $lchild(1) = \{6, 7, 8\}$.

Fig. 2: The tree representation of the membrane structure of the P system Π_2 of Example 2 and its trimmed version.

most one symbol (and this symbol can only be rewritten by chain rules-rules having one symbol on both the left- and right-hand sides), the left-hand membranes of any non-cooperative P system must always be FIN-representable. Moreover, the finite representation of any left-hand membrane sufficiently represents the regions that are inside it, so once we have the finite representation of a left-hand membrane, we do not need to pay further attention to any of its child membranes.

To formalize this idea, let $h \in trim(\mu)$ if the pair of membranes labeled by hL, hR are not contained in any (FIN-representable) left-hand membrane of μ. To define $trim(\mu)$, consider the tree representation of the membrane structure μ where the root is labeled by s, and each node has an equal number of children that are labeled by left-labels iL and right-labels iR, $i \in \{1, \ldots, n\}$. Let us consider the paths in this tree leading from the root s towards the leaves of the tree (which correspond the elementary membranes in the deepest level of μ), and let $\{i_1L, \ldots, i_kL\} \subseteq \{1L, \ldots, nL\}$ be those left-labels which label the nodes that are encountered as the first left-labels on these paths. Then $trim(\mu) = \{i_1, \ldots, i_k\}$ is the corresponding set of indices. Let also $i \in lchild(h) \subseteq \{1, \ldots, n\}$ for some $h \in trim(\mu)$ if the membrane labeled by iL is directly contained in a membrane labeled by s or hR. An example demonstrating these notions can be seen in Figure 2.

A transition system representing the FIN-representable region hL for some $h \in \{1, \ldots, n\}$ is defined as $M_{hL} = (Q_{hL}, q_{hL,0}, \delta_{hL})$, where Q_{hL} is the finite set of states, $q_{hL,0}$ is the initial state, and δ_{hL} is the state transition relation.

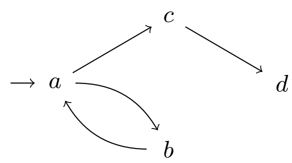


Fig. 3: Graphical demonstration of the possible membrane contents of the region $1L$ of the P system Π_2 of Example 2 with arrows indicating the initial contents and the successor relation between the multisets.

A state $q_{hL} \in Q_{hL}$ of such a transition system represents two important properties concerning the region hL in question:

1. What is the current symbol $w_{hL} = x_{hL} \in O$ contained by the region. This symbol is denoted by $\mathit{symbol}(q_{hL})$.
2. Whether all regions contained by hL reached a halting state. In this case, that is, if no rule is applicable in hL and in any of the regions contained by hL, we have $\delta_{hL}(q_{hL}) = \emptyset$.

Now, using the states of the transition systems $M_{jL} = (Q_{jL}, q_{jL,0}, \delta_{jL})$ for $j \in \mathit{trim}(\mu) = \{i_1, \dots, i_k\}$, we can abbreviate a configuration

$$(w_s, w_{1L}, w_{1R}, \dots, w_{nL}, w_{nR})$$

of the polymorphic P system Π above as

$$(w_s, q_{i_1L}, w_{i_1R}, \dots, q_{ikL}, w_{i_kR})$$

where q_{jL} are the states from the finite state sets of the transition system M_{jL}.

The way how these state representations change is described by the state transition function δ_{jL} of the transition system. If w'_s and w'_{i_jR} can be obtained from w_s and w_{i_jR}, respectively, for all $1 \leq j \leq k$, using the appropriate rules $\mathit{symbol}(q_{i_jL}) \to w_{i_jR}$ implied by the current configuration of the system, then

$$(w_s, q_{i_1L}, w_{i_1R}, \dots, q_{ikL}, w_{i_kR}) \Rightarrow (w'_s, q'_{i_1L}, w'_{i_1R}, \dots, q'_{ikL}, w'_{i_kR})$$

is the abbreviated notation of a possible computational step in Π, where $q'_{i_jL} \in \delta_{i_jL}(q_{i_jL})$, or $q'_{i_jL} = q_{jL}$ if $\delta_{i_jL}(q_{i_jL}) = \emptyset$.

Example 3. Let us construct the transition system $M_{1L} = (Q_{1L}, q_{1L,0}, \delta_{1L})$ representing the FIN-representable region $1L$ of the P system Π_2 of Example 2.

The possible contents of the region can be seen on Figure 3, so we need the states $Q_{1L} = \{q_{1L,0}, q_{1L,1}, q_{1L,2}, q_{1L,3}\}$ where the initial state is $q_{1L,0}$, and $\mathit{symbol}(q_{1L,0}) = a$, $\mathit{symbol}(q_{1L,1}) = b$, $\mathit{symbol}(q_{1L,2}) = c$, $\mathit{symbol}(q_{1L,3}) = d$. The transition relation is defined as

$$\begin{array}{ll} \delta_{1L}(q_{1L,0}) = \{q_{1L,1}, q_{1L,2}\}, & \delta_{1L}(q_{1L,1}) = \{q_{1L,0}\}, \\ \delta_{1L}(q_{1L,2}) = \{q_{1L,3}\}, & \delta_{1L}(q_{1L,3}) = \emptyset. \end{array}$$

Note that $q_{1L,3}$ represents a halting state of region $1L$.

Now the computational step

$$(a, \langle a, aa\rangle, \langle w_{2L}, w_{2R}\rangle, \langle w_{3L}, w_{3R}\rangle, \langle w_{4L}, w_{4R}\rangle, \langle w_{5L}, w_{5R}\rangle,$$
$$\langle w_{6L}, w_{6R}\rangle, \langle w_{7L}, w_{7R}\rangle, \langle w_{8L}, w_{8R}\rangle) =$$
$$(a, \langle a, aa\rangle, \langle a, c\rangle, \langle a, b\rangle, \langle b, a\rangle, \langle c, d\rangle, \langle a, b\rangle, \langle b, a\rangle, \langle a, e\rangle) \Rightarrow$$
$$(aa, \langle c, bb\rangle, \langle a, c\rangle, \langle a, b\rangle, \langle b, a\rangle, \langle c, d\rangle, \langle a, b\rangle, \langle b, a\rangle, \langle a, e\rangle)$$

of Π_2 can be abbreviated as

$$(a, \langle q_{1L,0}, aa\rangle, \langle w_{6L}, w_{6R}\rangle, \langle w_{7L}, w_{7R}\rangle, \langle w_{8L}, w_{8R}\rangle) =$$
$$(a, \langle q_{1L,0}, aa\rangle, \langle a, b\rangle, \langle b, a\rangle, \langle a, e\rangle) \Rightarrow$$
$$(aa, \langle q_{1L,2}, bb\rangle, \langle a, b\rangle, \langle b, a\rangle, \langle a, e\rangle).$$

4 PC ET0L Systems and Non-cooperative Polymorphic P Systems

In this section we show that languages of non-cooperative P systems can be generated by parallel communicating ET0L systems.

Theorem 1. $PsOP(polym, ncoo) \subseteq PsNPC(ET0L)$.

Proof. Let $\Pi = (O, T, \mu, w_s, \langle w_{1L}, w_{1R}\rangle, \ldots, \langle w_{nL}, w_{nR}\rangle, s)$ be a non-cooperative polymorphic P system. Since Π is non-cooperative, its left-hand regions are FIN-representable, so they are represented by the finite transition systems $M_{iL} = (Q_{iL}, q_{iL,0}, \delta_{iL}),\ 1 \leq i \leq n$. In the following construction, we are interested in the ones that are the outmost, that is, the ones that are directly contained in a right-hand region, that is, in the regions jL for $j \in trim(\mu)$.

We construct a PC ET0L system Γ, such that $Ps(\Gamma) = Ps(\Pi)$. We need components for all regions that are not necessarily FIN-representable, that is, for the skin, and for all right-hand regions which are not contained in left-hand regions, and we also have $n + 2$ assistant components in addition.

Before continuing the presentation of the PC ET0L system, we introduce a number of notations that will ease the description of Γ.

If $lchild(h) = \{i_1, \ldots, i_{k_h}\}$ for a region $h \in trim(\mu)$, let $\bar{Q}_{lchild(h)} = Q_{i_1L} \times \ldots \times Q_{i_{k_h}L}$ denote the direct product of the sets Q_{jL}, the state sets of the transition systems $M_{jL} = (Q_{jL}, q_{jL,0}, \delta_{jL})$ for all labels $j \in lchild(h)$ (representing the FIN-representable children of membrane h).

Let also, for some $\bar{q} \in \bar{Q}_{lchild(h)}$, $symbol(\bar{q}) = \{x \mid x = symbol(q_{i_jL}),\ 1 \leq j \leq k_h\}$ be the set of symbols corresponding the contents of the regions in $lchild(h)$ in the configuration represented by the component states of $\bar{q}$. Moreover, let $\bar{q}' \in succ(\bar{q})$ for some $\bar{q}' = (q'_{i_1L}, \ldots, q'_{i_{k_h}L})$, $\bar{q} = (q_{i_1L}, \ldots, q_{i_{k_h}L}) \in \bar{Q}_{lchild(h)}$, if and only if, $q'_{i_jL} \in \delta_{i_jL}(q_{i_jL})$. If $\delta_{i_jL}(q_{i_jL}) = \emptyset$, then $q'_{i_jL} = q_{i_jL},\ 1 \leq j \leq k_h$.

We say that $\bar{q} = (q_{i_1L}, \ldots, q_{i_{k_h}L}) \in \bar{Q}_{lchild(h)}$ is *all-final*, if all components of the direct product are final states, that is, if $\delta_{i_jL}(q_{i_jL}) = \emptyset$, for all $1 \leq j \leq k_h$.

Let us denote by $w^{h,i}$ for some $w \in V^*$ the string $g^{h,i}(w)$, where we extend $g^{h,i} : O \to V$ to strings, and for $a \in O$, $g^{h,i}(a) = a^{h,i}$, the symbol labeled with the indices $h \in \{s, jR \mid j \in trim(\mu)\}$, $i \in \{0, 1, \ldots, d+1\}$. Here d denotes the maximal depth of the trimmed membrane structure, that is $d = max(D)$ where $D = \{\text{the depth of } (jL) \mid j \in trim(\mu)\}$ (we assume that the skin membrane is not elementary).

If $trim(\mu) = \{i_1, \ldots, i_k\}$, then let

$$\Gamma = (V, K, T, G_s, G'_s, G_{i_1R}, G'_{i_1R} \ldots, G_{i_kR}, G'_{i_kR}, G_c, G_m)$$

where G_m is the master component,

$$\begin{aligned} V = O \cup \{&a^{h,i} \mid a \in O, h \in \{s, jR \mid j \in trim(\mu)\}, 0 \le i \le d+1\} \cup \\ &\{[\bar{q}]^i_h \mid \bar{q} \in \bar{Q}_{lchild(h)}, h \in \{s, jR \mid j \in trim(\mu)\}, 0 \le i \le d+1\} \cup \\ &\{S_c, S'_c, S''_c, S_m, S'_h \mid h \in \{s, jR \mid j \in trim(\mu)\}\} \end{aligned}$$

is the alphabet of nonterminals, and the set $K = \{K_{jR}, K'_{jR} \mid j \in trim(\mu)\} \cup \{K_s, K'_s, K_c, K_m\}$ is the alphabet of query symbols.

The components $G_I = (V \cup K, T, \omega_I, U_I)$, $I \in \{jR, jR' \mid j \in trim(\mu)\} \cup \{s, s', c, m\}$ of Γ are defined as follows.

(1) If $h \in \{jR \mid j \in trim(\mu)\}$ is an elementary region, then let

$$\begin{aligned} &\omega_h = w_h^{h,0}, \text{ and} \\ &U_h = \{P_h\} \text{ with} \\ &P_h = \{a^{h,i} \to a^{h,i+1}, a^{h,d+1} \to a^{h,0} \mid a \in O,\ 0 \le i \le d\}. \end{aligned}$$

These components maintain the non-dynamical right-hand sides of the rules corresponding to the membrane label h. The symbols in these regions will be replacing the left-hand sides of the corresponding dynamical rules by communication. The communication will be initiated with queries introduced by the components generating the string corresponding to their parent regions (the regions where the rules they represent can be applied). Since the simulation of one P system step will be done in $d+2$ rewriting steps, the symbols are indexed, and the indices are changed in a circular manner.

(2) If $h \in \{s, jR \mid j \in trim(\mu)\}$ is a non-elementary region, then let

$$\omega_h = w_h^{h,0}[\bar{q}_0]^0_h, \text{ where } lchild(h) = \{i_1, \ldots, i_{k_h}\} \text{ and } \bar{q}_0 = (q_{i_1L,0}, \ldots, q_{i_{k_h}L,0})$$
$$\text{with } q_{i_jL,0} \text{ being the initial state of } M_{i_jL},\ 1 \le j \le k_h.$$

This initial string (and the sentential form of such a component in general) is an indexed representation of the contents of membrane h (indexed with h and a value from $i \in \{0, 1, \ldots, d+1\}$), augmented with a symbol corresponding to the states of the transition systems representing the contents of the left-hand children of region h. The configuration of the left-child regions of h are represented as the direct product of the states of the transition systems M_{jL}, $j \in lchild(h)$, corresponding to the left-children of h.

Now, the tables of the component associated to region $h \in \{s, jR \mid j \in trim(\mu)\}$, as above, are defined in the following way.

$$
\begin{aligned}
U_h &= \{P_{h,\bar{q}} \mid \bar{q} \in \bar{Q}_{lchild(h)}\}, \text{ where} \\
P_{h,\bar{q}} &= \{[\bar{q}]_h^0 \to [\bar{q}]_h^1,\ [\bar{r}]_h^0 \to F \mid \bar{r} \neq \bar{q},\ \bar{r} \in \bar{Q}_{lchild(h)}\} \cup \\
&\quad \{[\bar{q}]_h^i \to [\bar{q}]_h^{i+1} \mid 1 \leq i \leq k-1,\ k = level(h)\} \cup \\
&\quad \{a^{h,k} \to K_{i_jR},\ b^{h,k} \to b^{h,k+1} \mid k = level(h), a \in symb(\bar{q}), \\
&\quad\ \ b \notin symb(\bar{q})\} \cup \\
&\quad \{a^{h,i} \to a^{h,i+1},\ a^{h,d+1} \to a^{h,0} \ \mid 0 \leq i \leq d,\ i \neq k, a \in O\} \cup \\
&\quad \{a^{g,i} \to a^{g,i+1},\ a^{g,d+1} \to a^{h,0} \ \mid k = level(h), k+1 \leq i \leq d, \\
&\quad\ \ g \neq h,\ a \in O\} \cup
\end{aligned}
$$

$$
\begin{aligned}
&\{[\bar{q}]_h^k \to [\bar{q}']_h^{k+1},\ [\bar{q}']_h^i \to [\bar{q}']_h^{i+1},\ [\bar{q}']_h^{d+1} \to [\bar{q}']_h^0 \mid \bar{q}' \in succ(\bar{q}), \\
&\ \ k = level(h),\ k+1 \leq i \leq d\} \cup \\
&\{[\bar{t}]_g^i \to \lambda \mid \bar{t} \in \bar{Q}_{lchild(g)}, g \in \{s, jR \mid j \in trim(\mu)\} \\
&\ \ 0 \leq i \leq d+1\}.
\end{aligned}
$$

The tables above simulate the application of the current instances of the dynamical rules to the contents of the right-hand regions h. Each possible combination of rules can be applied.

The left-hand sides of the rules available in region h are represented by the states of the transition systems corresponding to the left-children of h, that is, by the symbol $\bar{q} \in \bar{Q}_{lchild(h)}$. The table $P_{h,\bar{q}}$ corresponding to this symbol (indexed by this symbol) can be applied if an indexed version of the nonterminal $[\bar{q}]_h$ is present in the sentential form (otherwise the trap symbol F is introduced). After k steps, where k is the depth of membrane h, the symbols on the left-hand sides of the dynamical rules introduce the query symbol querying the components which correspond to the right-hand sides of the rules in question, simulating this way the execution of the current instances of the corresponding dynamical rules. The delay depending on the depth of membrane h is necessary in order to start the rule application from the skin membrane, and then to continue step-by-step towards the deeper regions. This is important, since the changing of the dynamical rules should only be simulated after they were applied in their current form.

In the step when the queries are introduced, the configuration change of the FIN-representable left-hand regions are also simulated by the changing of the symbol $\bar{q}$ representing the states of the transition systems to $\bar{q}' \in succ(\bar{q})$, the representation of a possible successor configuration. If $\bar{q}' \neq \bar{q}$, then the table $P_{h,q'}$ corresponding to the new configuration will be applied in the next simulating cycle.

In addition to the components described so far, we also need two additional ones in order to be able to finish the simulation. The task of these components is to check whether the simulated configuration of the P system is a halting configuration (only in this case, the finishing of the simulation is allowed), and then to delete the nonterminal symbols from the generated string.

(3) Components $G'_h = (V \cup K, T, S'_h, U'_h),\ h \in \{s, jR \mid j \in \mathit{trim}(\mu)\}$ are defined as

$$\begin{aligned}
\omega'_h &= S'_h,\\
U'_h &= \{P'_h\} \text{ where}\\
P'_h &= \{S'_h \to K_c,\ S_c \to K_c,\ S'_c \to K_h\} \cup\\
&\quad \{[\bar{q}]^1_h \to \lambda,\ [\bar{r}]^1_h \to F \mid \bar{q}, \bar{r} \in \bar{Q}_{lchild(h)} \text{ where } \bar{q} \text{ is all-final},\\
&\quad\ \bar{r} \text{ is not all-final}\} \cup\\
&\quad \{a^{h,1} \to F,\ b^{h,1} \to \lambda \mid a, b \in O, a \in \mathit{symb}(\bar{q}), b \notin \mathit{symb}(\bar{q})\}.
\end{aligned}$$

These components check whether the simulated configuration is a halting configuration of the P system Π. When the sentential form of G_c turns to S'_c, Γ attempts to finish the generative process. First it initiates the checking phase of the derivation by making these primed components query their unprimed counterparts. Once the sentential forms are transferred from G_h to G'_h, these erase all symbols from all strings if all the left-hand configurations are halting, and moreover, if no further rewriting is possible with the current instance of the dynamical rules. If these conditions hold, the sentential forms disappear, otherwise the trap symbol F is introduced.

(4) Components G_c and G_m are defined as follows.

$$\begin{aligned}
\omega_c &= S_c,\\
T_c &= \{P_c\} \text{ with}\\
P_c &= \{S_c \to S_c, S_c \to S'_c, S'_c \to S'_c\}.
\end{aligned}$$

When this component changes its sentential form to S'_c, the final checking phase of the simulation process is initiated, as explained above.

The result of the derivation is produced by component G_m, the master component which is defined below.

$$\begin{aligned}
\omega_m &= S_m,\\
T_m &= \{P_m\} \text{ with}\\
P_m &= \{S_m \to K_c,\ S_c \to K_c,\ S'_c \to K_s S''_c,\\
&\qquad S''_c \to K'_s K'_{i_1 R} \dots K'_{i_k R} \mid \text{ where } \mathit{trim}(\mu) = \{i_1, \dots, i_k\}\} \cup\\
&\quad \{a^{s,1} \to a \mid a \in T\} \cup \{a^{s,1} \to \lambda \mid a \in V - T\}.
\end{aligned}$$

When G_c turns its sentential form to S'_c, this component receives the sentential form of component G_s (which corresponds to the output region of Π), then erases all nonterminals, and also queries the primed components G'_h to make sure that no trap symbol was introduced, that is, the computation of Π

was correctly simulated by Γ. If all went well, the word corresponding to the terminal contents of the skin region is generated.

To show that the PC ET0L system Γ simulates the P system Π, consider the abbreviated version of the initial configuration of Π, as introduced in the previous section,

$$(w_s, q_{i_1L}, w_{i_1R}, \ldots, q_{ikL}, w_{i_kR})$$

where $trim(\mu) = \{1_1, \ldots, i_k\}$. This corresponds to the initial configuration of Γ

$$(w_s^{s,0}[\bar{q}_{s,0}]_s^0, S'_s, w_{1R}^{1R,0}[\bar{q}_{1R,0}]_{1R}^0, S'_{1R}, \ldots, w_{nR}^{nR,0}[\bar{q}_{nR,0}]_{nR}^0, S'_{nR}, S_c, S_m)$$

in the following way. In general, symbols of type $[\bar{q}]_h^j,\ h \in \{s, 1R, \ldots, nR\}$ represent an element of the direct product of the state sets of M_{iL} where $iL \in lchild(h)$. In the initial confguration above, a symbol $[\bar{q}_0]_h^0$ represents the left-hand sides of the rules that can be used in region h in the initial configuration, since $\bar{q}_0$ is the direct product of the initial configurations of the transition systems $M_{iL},\ iL \in lchild(h)$. If we let x_{iL} be the symbol which is contained by a region $iL \in lchild(h)$ in the initial configuration, then occurrences of this symbol in w_h should be rewritten by the current state of the dynamical rule $x_{iL} \to w_{iR}$. This will be achieved by communication, by replacing occurrences of x_{iL}^k (for some k) in the sentential form of G_h with the sentential form of component G_{iR}. The replacement takes place in the order of the depth of the rule in question, starting with the rules in the skin region (the rules with left-hand regions of depth 1).

The initial step of Γ is

$$(w_s^{s,0}[\bar{q}_{s,0}]_s^0, S'_s, w_{1R}^{1R,0}[\bar{q}_{1R,0}]_{1R}^0, S'_{1R}, \ldots, w_{nR}^{nR,0}[\bar{q}_{nR,0}]_{nR}^0, S'_{nR}, S_c, S_m) \Rightarrow$$
$$(w_s^{s,1}[\bar{q}_{s,0}]_s^1, K_c, w_{1R}^{1R,1}[\bar{q}_{1R,0}]_{1R}^1, K_c, \ldots, w_{nR}^{nR,1}[\bar{q}_{nR,0}]_{nR}^1, K_c, S_c, S_m),$$

and then we get

$$(w_s^{s,1}[\bar{q}_{s,0}]_s^1, S_c, w_{1R}^{1R,1}[\bar{q}_{1R,0}]_{1R}^1, S_c, \ldots, w_{nR}^{nR,1}[\bar{q}_{nR,0}]_{nR}^1, S_c, S_c, S_m),$$

after the following communication. The tables $P_{h,\bar{q}_0}$ had to be used in each component $G_h,\ h \in \{s, jR \mid j \in trim(\mu)\}$ (or the trap symbol F would have been introduced). By continuing to use the same tables, the rule applications in the skin region (the region with depth $k = 1$) are simulated.

$$(w_s^{s,1}[\bar{q}_{s,0}]_s^1, S_c, w_{1R}^{1R,1}[\bar{q}_{1R,0}]_{1R}^1, S_c, \ldots, w_{nR}^{nR,1}[\bar{q}_{nR,0}]_{nR}^1, S_c, S_c, S_m) \Rightarrow$$
$$(u^{s,2}[\bar{q}_{s,0}]_s^2, K_c, w_{1R}^{1R,2}[\bar{q}_{1R,0}]_{1R}^2, K_c, \ldots, w_{nR}^{nR,2}[\bar{q}_{nR,0}]_{nR}^2, K_c, S_c, S_m)$$

where $u^{s,2}$, the sentential form of G_s (the component corresponding to the skin region) is the result of rewriting each symbol in $symb(\bar{q}_{s,0})$ to the appropriate query symbols (where $\bar{q}_{s,0}$ is the representation of the initial states of all the regions jL with $j \in lchild(s)$). "Appropriate" means here that if region iL is the left-hand region and iR is the right-hand region of rule i, and x_{iL} is the contents of region iL, then all occurrences of $x_{iL} \in symbol(\bar{q}_{s,0})$ in $w_s^{s,0}$ are rewritten to K_{iR} in $u^{s,1}$, which are replaced by the contents of region $1R$ in the following communication step.

Continuing this way, step-by-step, eventually reaching the deepest regions of Π with the simulation, we arrive to a configuration

$$(v_s^{s,d+1}[\bar{q}_s]_s^{d+1}, \alpha, v_{1R}^{1R,d+1}[\bar{q}_{1R}]_{1R}^{d+1}, \alpha, \dots, v_{nR}^{nR,d+1}[\bar{q}_{nR}]_{nR}^{d+1}, \alpha, S_c, S_m),$$

and then

$$(v_s^{s,0}[\bar{q}_s]_s^0, \alpha, v_{1R}^{1R,0}[\bar{q}_{1R}]_{1R}^0, \alpha, \dots, v_{nR}^{nR,0}[\bar{q}_{nR}]_{nR}^0, \alpha, S_c, S_m),$$

where $v_h^{h,0}$ is such, that v_h is the string corresponding to the contents of region $h \in \{s, jR \mid j \in trim(\mu)\}$ after the first rewriting step, while $\bar{q}_h$ is the representation of the the contents of each region $jL \in lchild(h)$ (these are the left-hand regions of the rules that are applicable in region h).

The derivation simulating the functioning of Π might continue this way in a similar fashion, until G_c introduces the nonterminal S_c'. If this happens in the appropriate step, we have

$$\begin{aligned}(v_s^{s,0}[\bar{q}_s]_s^0, K_c, v_{1R}^{1R,0}[\bar{q}_{1R}]_{1R}^0, K_c, \dots, v_{nR}^{nR,0}[\bar{q}_{nR}]_{nR}^0, K_c, S_c', K_c) \Rightarrow \\ (v_s^{s,0}[\bar{q}_s]_s^0, S_c', v_{1R}^{1R,0}[\bar{q}_{1R}]_{1R}^0, S_c', \dots, v_{nR}^{nR,0}[\bar{q}_{nR}]_{nR}^0, S_c', S_c', S_c') \Rightarrow \\ (v_s^{s,1}[\bar{q}_s]_s^1, K_s, v_{1R}^{1R,1}[\bar{q}_{1R}]_{1R}^1, K_{1R}, \dots, v_{nR}^{nR,1}[\bar{q}_{nR}]_{nR}^1, K_{nR}, S_c', K_s S_c''),\end{aligned}$$

and then

$$\begin{aligned}(v_s^{s,1}[\bar{q}_s]_s^1, v_s^{s,1}[\bar{q}_s]_s^1, v_{1R}^{1R,1}[\bar{q}_{1R}]_{1R}^1, v_{1R}^{1R,1}[\bar{q}_{1R}]_{1R}^1, \dots \\ \dots, v_{nR}^{nR,1}[\bar{q}_{nR}]_{nR}^1, v_{nR}^{nR,1}[\bar{q}_{nR}]_{nR}^1, S_c', v_s^{s,1}[\bar{q}_s]_s^1 S_c'').\end{aligned}$$

Now, if for any $h \in \{s, jR \mid j \in trim(\mu)\}$, none of the strings $v_h^{h,1}$ can be rewritten with the current instances of the dynamical rules, and moreover, all $\bar{q}_h$ represent state combinations where all the elements are halting states, then G_h' all produce the empty string, and G_m generates a terminal word v' containing the terminal objects from v_s, and then the derivation is finished as

$$\begin{aligned}(v_s^{s,2}[\bar{q}_s]_s^2, \lambda, v_{1R}^{1R,2}[\bar{q}_{1R}]_{1R}^2, \lambda, \dots, v_{nR}^{nR,2}[\bar{q}_{nR}]_{nR}^2, \lambda, S_c', v' K_s' K_{1R}' \dots K_{nR}') \Rightarrow \\ (v_s^{s,2}[\bar{q}_s]_s^2, \lambda, v_{1R}^{1R,2}[\bar{q}_{1R}]_{1R}^2, \lambda, \dots, v_{nR}^{nR,2}[\bar{q}_{nR}]_{nR}^2, \lambda, S_c', v'),\end{aligned}$$

or otherwise the trap symbol F is introduced, and the derivation cannot be successful any more.

5 Conclusions and Topics for Investigation

We have continued the investigations concerning the generative power of non-cooperative polymorphic P systems, and we have shown that they can be simulated by nonreturning PC ET0L systems, thus, that the class of Parikh sets of nonreturning PC ET0L languages include the class of languages generated by non-cooperative polymorphic P systems.

It seems to be clear that the simulation does not work in the other direction, namely that polymorphic P systems cannot simulate PC ET0L systems in a straightforward manner, because the communication structure of parallel communicating systems can be more complicated then those which could be imitated by the simple tree-like membrane structure of P systems. On the other hand, it seems to be possible to simulate computations of centralized PC ET0L systems (systems where only the master component can introduce queries). The exact characterization of languages of non-cooperative polymorphic systems with PC ET0L systems having some kind of a simplified communication structure seems to be an interesting topic for further research which we also plan to investigate in the future.

References

1. Alhazov, A., Ivanov, S., Rogozhin, Y.: Polymorphic P systems. In: Gheorghe, M., Hinze, T., Păun, G., Rozenberg, G., Salomaa, A. (eds.) Membrane Computing. Lecture Notes in Computer Science, vol. 6501, pp. 81–94. Springer-Verlag, Berlin, Heidelberg (2011)
2. Ivanov, S.: Polymorphic P systems with non-cooperative rules and no ingredients. In: Gheorghe, M., Rozenberg, G., Salomaa, A., Sosík, P., Zandron, C. (eds.) Membrane Computing. Lecture Notes in Computer Science, vol. 8961, pp. 258–273. Springer International Publishing, Cham (2014)
3. Kuczik, A., Vaszil, G.: Simple variants of non-cooperative polymorphic P systems. Journal of Membrane Computing (2024). https://doi.org/10.1007/s41965-024-00145-0
4. Lindenmayer, A.: Mathematical models for cellular interactions in development I. Filaments with one-sided inputs. Journal of Theoretical Biology **18**(3), 280–299 (1968)
5. Păun, G.: Membrane Computing: An Introduction. Springer-Verlag, Berlin, Heidelberg (2002)
6. Păun, G., Rozenberg, G., Salomaa, A. (eds.): The Oxford Handbook of Membrane Computing. Oxford University Press, Oxford (2010)
7. Rozenberg, G., Salomaa, A. (eds.): Handbook of Formal Languages. Springer-Verlag, Berlin Heidelberg (1997)
8. Rozenberg, G., Salomaa, A.: Lindenmayer systems: impacts on theoretical computer science, computer graphics, and developmental biology. Springer Science and Business Media, Berlin, Heidelberg (1992)