On the Syntactic Complexity of Darwinian Membrane Systems*

Jürgen Dassow[1], Erzsébet Csuhaj-Varjú[2]

[1] Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik
PSF 4120, D–39016 Magdeburg, Germany
dassow@iws.cs.uni-magdeburg.de

[2] Computer and Automation Research Institute,
Hungarian Academy of Sciences
Kende u. 13–17, H-1111 Budapest, Hungary
csuhaj@sztaki.hu

Summary. Membrane or P systems form a distributed parallel model of computing which is obtained as an abstraction from the structure and functioning of living cells. In this paper we consider a very basic membrane system and add to it checkers which are special finite automata to check whether or not a configuration of the membrane system is "good" or "bad". The computation is only continued if the configuration is good, otherwise it stops without a result. Such membrane systems are called Darwinian P systems. There are three parameters characterizing the size of a system, the number of membranes, the number of checkers, and the size of the checkers measured by the number of states. We prove that we can generate all sets of Parikh vectors of recursively enumerable languages if we restrict the number and size of checkers to one checker with six states or to five checkers with two states. Moreover, we prove that Darwinian systems with one membrane and checkers with at most two states are also sufficient to generate all Parikh sets of recursively enumerable languages. The latter result is optimal since Darwinian systems with checkers with only one state generate only sets of Parikh vectors of ET0L languages. All results are valid for length sets instead of sets of Parikh vectors, too.

1 Introduction

Membrane systems (also called P systems) form a distributed parallel model of computing which is obtained as an abstraction from the structure and functioning

* This research was supported in part under grant no. D-35/2000 and HUN009/00 by the Intergovernmental S&T Cooperation Programme of the Office of Research and Development Division of the Hungarian Ministry of Education and its German partner, the Federal Ministry of Education and Research (BMBF), and under grant no. F037567 by the Hungarian Scientific Research Fund, "OTKA".

of living cells. They consists of a membrane structure and evolving rules for any region formed by the membranes. The evolving rules substitute a symbol by some other symbols which can also be sent to the inner and outer neighbors of the region. A configuration is a tuple formed by the multisets of symbols occurring in the regions. By maximally parallel applications of the evolving rules of a region to the symbols in the region we produce sequences of configurations. A configuration is called halting, if, for any region, there is no applicable rule. The result of such a "computation" process is the Parikh vector of the multiset (or the number of symbols in the multiset) contained in a region, which is determined in advance, at the end of a successful computation. By this simple type of membrane systems we can only generate semilinear sets or ultimately periodic sets of integers as shown in [3] (see also [6]). In order to increase the generative power several further ideas from cell biology are added, e.g., membrane dissolving, priorities between the evolving rules, cooperation between symbols in a region. For a detailed discussion of membrane systems we refer to the monograph [6].

In [1] the authors introduced checkers for the above simple membrane system. A checker is a finite automaton which reads symbols of a multiset in some order and rejects or accepts if all symbols are read. Such checkers are used to evaluate the configurations of a membrane system. The computation stops without a result if the checker rejects and is continued otherwise. Membrane systems with checkers are called Darwinian because we have a selection process by the checkers.

In [1] it has been shown that Darwinian membrane systems are very powerful. They can generate all recursively enumerable sets of vectors of non-negative integers. Moreover, this computational completeness already holds for Darwinian systems with one membrane and one checker as well as for systems with one membrane and checkers with at most three states.

In this paper we improve the latter result, showing that one membrane and checkers with at most two states are sufficient to generate all sets of Parikh vectors of recursively enumerable languages or all length sets of recursively enumerable languages. Moreover, this bounds are sharp because we show that Darwinian membrane systems with checkers with only one states generate exactly the sets of Parikh vectors or the length sets of ET0L languages which form a proper subset of the recursively enumerable sets. Furthermore, we prove that we can bound the number of checkers and the size of the checkers to small numbers without loosing the computational completeness.

2 Matrix Grammars and ET0L Systems

Throughout the paper we assume that the reader is familiar with the basic concepts of formal language theory. We recall here some notions and their notations; for details we refer to [8], [7], [2].

The set of non-empty words over an alphabet V is denoted by V^+; if the empty string λ is included, then we use the notation V^*. A set of strings $L \subseteq V^*$ is said to be a language over V. For a string $w \in V^*$, we denote the length of w by $|w|$, and for a set of symbols $U \subseteq V$ we denote by $|w|_U$ the number of occurrences of letters of U in w. The length set $N(L)$ of a language L is defined as

$$N(L) = \{|w| \mid w \in L\}\,.$$

Let $V = \{a_1, a_2, \ldots, a_n\}$ be an alphabet. With any word $w \in V^*$ we associate its Parikh vector

$$\pi_V(w) = (|w|_{a_1}, |w|_{a_2}, \ldots, |w|_{a_n})\,.$$

For a language $L \subseteq V^*$, we define its Parikh language by

$$\pi_V(L) = \{\pi_V(w) \mid w \in L\}\,.$$

By *PsRE* (and *NRE*) we denote the family of all Parikh sets (and length sets, respectively) of recursively enumerable languages.

For a context-free rule $r = a \to w$, we set $lhs(r) = a$ and $rhs(r) = w$.

A *matrix grammar* with appearance checking is a construct $G = (N, T, S, M, F)$, where N, T are disjoint alphabets, $S \in N$, M is a finite set of sequences of context-free rules over $N \cup T$, and F is a set of occurrences of rules in M. The elements of N are called *nonterminals*, those in T are called *terminals*, S is the *axiom*, the sequences in M are written in the form

$$m = (A_1 \to x_1, \ldots, A_n \to x_n),\ n \geq 1,\ A_i \in N, x_i \in (N \cup T)^* \text{ for } 1 \leq i \leq n,$$

and are called matrices.

For $w, w' \in (N \cup T)^*$ we write $w \Longrightarrow w'$ if there is a matrix $m = (A_1 \to x_1, A_2 \to x_2, \ldots, A_n \to x_n)$ in M and strings $w_1, w_2, \ldots, w_{n+1}$ in $(N \cup T)^*$ such that $w_1 = w, w_{n+1} = w'$, and for each $i = 1, 2, \ldots, n$, one of the following cases holds:

— $w_i = w_i' A_i w_i''$ and $w_{i+1} = w_i' x_i w_i''$, or

— $|w_i|_{A_i} = 0$, $w_i = w_{i+1}$, and $A_i \to x_i$ appears in F.

In words, all the rules in m are used one after the other, possibly skipping those rules which appear in F providing that they cannot be applied to the current sentential form.

The language generated by G is defined by

$$L(G) = \{x \in T^* \mid S \Longrightarrow^* x\},$$

where $\Longrightarrow^*$ is the reflexive and transitive closure of the relation $\Longrightarrow$.

For any recursively enumerable language L, there is matrix grammar G with appearance checking such that $L = L(G)$ (see [2] for a proof). From this fact, it follows that *PsRE* (and *NRE*) are the families of all Parikh sets (and length sets, respectively) of languages generated by matrix grammars with appearance checking.

A matrix grammar $G = (N, T, S, M, F)$ with appearance checking is in *binary normal form* if $N = N_1 \cup N_2 \cup \{S, \#\}$, where these sets are mutually disjoint, and all matrices of M have one of the following forms:

1. $(S \rightarrow XA)$ with $X \in N_1$, $A \in N_2$,
2. $(X \rightarrow Y, A \rightarrow x)$ with $X, Y \in N_1$, $A \in N_2$, $x \in (N_2 \cup T)^*$,
3. $(X \rightarrow Y, A \rightarrow \#)$ with $X, Y \in N_1$, $A \in \{B_1, B_2\} \subset N_2$,
4. $(X \rightarrow \lambda, A \rightarrow x)$ with $X \in N_1$, $A \in N_2$, $x \in T^*$,

F exactly consists of all rules $A \rightarrow \#$ appearing in matrices of type 3, and there is only one matrix of type 1. # is a trap symbol, once introduced, it is never removed. By [2] and [4], for any matrix grammar G_1 with appearance checking, there is a matrix grammar G_2 with appearance in binary normal form such that $L(G_1) = L(G_2)$. In the sequel we will assume that

– $m_1, m_2, \ldots, m_{q_1}$ are the matrices of type 2 or type 4,
– $m_{q_1+1}, m_{q_1+2}, \ldots, m_{q_2}$ are the matrices of type 3 with $A = B_1$, and
– $m_{q_2+1}, m_{q_2+2}, \ldots, m_q$ are the matrices of type 3 with $A = B_2$.

Moreover, for $1 \leq i \leq q$, let $m_i = (r_i, s_i)$ where r_i and s_i are the context-free rules given in the order as above.

An *ET0L system* is an $n+3$-tuple $G = (V, T, P_1, P_2, \ldots, P_r, w)$, where V is an alphabet, T is a subset of V (the alphabet of terminal symbols), $w \in V^+$ is the axiom, and, for $1 \leq i \leq r$, P_i is a complete set of context-free productions over V, i.e., for any symbol $a \in V$, the production set P_i contains a rule with a being on its left-hand side and an word from V^* on its right-hand side.

For two strings $x = x_1 x_2 \ldots x_n$, $y = y_1 y_2 \ldots y_n$ with $n \geq 1$, $x_i \in V$, $y_i \in V^*$ for $1 \leq i \leq n$, we say that x directly derives y, denoted by $x \Longrightarrow_G y$ if there is a production set P_j, $1 \leq j \leq r$, such that $x_i \rightarrow y_i \in P_j$ for $1 \leq i \leq n$. The language $L(G)$ generated by an ET0L system is defined as the set of all words over T which can be obtained from S by a sequence of direct derivation steps.

A *regularly controlled* ET0L system $G = (V, T, P_1, P_2, \ldots, P_r, S, R)$ is an ET0L system $(V, T, P_1, P_2, \ldots, P_r, S)$ equipped with a regular set R over the set of productions. The language generated by a regularly controlled ET0L system consists of all words over T which can be obtained from S by application of a sequence r of productions where $r \in R$.

In [5] (see also [2]) the following lemma has been shown.

Lemma 1. *For any regularly controlled ET0L system G, there is an ET0L system G' such that $L(G) = L(G')$.*

By *PsET0L* (and *NET0L*) we denote the set of all Parikh sets (and length sets, respectively) of languages generated by ET0L systems.

3 Darwinian Membrane Systems

Throughout the paper we assume that the reader is familiar with basic concepts of membrane computing. We here recall some notions informally; for details we refer to [6].

A *multiset* M is a pair (V,f) where V is a finite set[3] and f is a mapping from V into $\mathbf{N}$. For $a \in V$, $f(a)$ is called the multiplicity of a. The support $supp(M)$ of $M=(V,f)$ is defined as the set of all elements a of V with $f(a)>0$. Given a multiset $M=(V,f)$ with $V=\{a_1,a_2,\ldots,a_n\}$, we define its Parikh vector and its length by setting

$$\pi_V(M)=(f(a_1),f(a_2),\ldots,f(a_n)) \quad \text{and} \quad l(M)=\sum_{i=1}^{n} f(a_i)\,.$$

Let $w\in V^*$. We define the multiset $M(w)$ associated with w by $M(w)=(V,f)$ where $f(a)=|w|_a$. Conversely, any multiset $M=(V,f)$ can be described by a word $w(M)=a_1^{f(a_1)}a_2^{f(a_2)}\ldots a_n^{f(a_n)}$.

In the sequel we often present a multiset by listing all its elements in braces (as in the cases of sets, however, any element a is listed $f(a)$ times).

We now present the notion of a membrane structure which is one of the basic notions of membrane computing. Intuitively, a membrane structure μ can be given as a Venn diagram of a set and its subsets where a subset is included in another subset or the subsets are disjoint. The borders of the sets correspond to the membranes. A region corresponds to a set from which we have taken away all elements of its subsets. More formally, μ is a tree where the root corresponds to the set, and a node i is a child of a node j, if the set corresponding to i is a subset of the set corresponding to j. If i is a child of j, then we call the region i (more precisely, the region corresponding to i) an inner region of the region j, and the region j is called an outer region of i. Another description of a membrane structure can be given as a string of matching parentheses. If i contains the subsets $i_1,i_2,\ldots,i_n$ (or equivalently, i has the children $i_1,i_2,\ldots,i_n$), then we represent this by $[_i[_{i_1}\ ]_{i_1}[_{i_2}\ ]_{i_2}\cdots[_{i_n}\ ]_{i_n}]_i$. Obviously, the number of regions corresponds to the number of membranes.

Let μ be a membrane structure with m membranes, and let V be an alphabet. A configuration of μ over V is an m-tuple $(K_1,K_2,\ldots,K_m)$, where, for $1\le i\le m$, K_i is a multiset over V, i.e., $K_i=(V,f_i)$ for some f_i. Intuitively, we consider a one-to-one relation between the m regions of μ to the numbers $1,2,\ldots,m$, and assume that the i-th region contains the multiset K_i of symbols of V.

Obviously, if we only consider a configuration itself it is not necessary to know the concrete membrane structure μ, it is sufficient to know the number of regions (or membranes) of μ.

We say that a word

$$C'=[_{i_1}a_{t_1}]_{i_1}[_{i_2}a_{t_2}]_{i_2}\cdots[_{i_q}a_{t_q}]_{i_q},$$

where we assume that the letter a_{t_j} is contained in the i_j-th region, is associated with a configuration $C=(K_1,K_2,\ldots,K_m)$ if

[3] In general, V can be an infinite set, but for our purposes it is sufficient to restrict to finite sets.

$$K_i = \{a_{t_j} \mid i_j = i\}\,.$$

We are interested in evaluations of configurations where we restrict to evaluations which only distinguish "bad" configurations from "good" configurations. This will be done by checkers which are defined as follows. A *checker* is a finite automaton

$$H = (Q, V, m, s_0, F, P),$$

where Q is a finite set of states, V is an alphabet, m is a positive integer (the number of regions of a membrane structure), s_0 is the initial state, F is the set of final states, and P is a set of rules of the form $s[_i a]_i \rightarrow [_i a]_i s'$, where $s, s' \in Q, a \in V, 1 \leq i \leq m$ (which define the transitions of the automaton). If a checker is in state s and reads the object a of region i, then it changes its state to s'.

We extend the relation $\rightarrow$ to words associated with a configuration by setting

$$\begin{array}{l}[_{i_1} a_{t_1}]_{i_1} \dots [_{i_{p-1}} a_{t_{p-1}}]_{i_{p-1}} s [_{i_p} a_{t_p}]_{i_p} \dots [_{i_q} a_{t_q}]_{i_q} \Longrightarrow_H \\ [_{i_1} a_{t_1}]_{i_1} \dots [_{i_p} a_{t_p}]_{i_p} s' [_{i_{p+1}} a_{t_{p+1}}]_{i_{p+1}} \dots [_{i_q} a_{t_q}]_{i_q},\end{array}$$

if $s[_{i_p} a_{t_p}]_{i_p} \rightarrow [_{i_p} a_{t_p}]_{i_p} s' \in P$. Let $\Longrightarrow_H^*$ be the reflexive and transitive closure of $\Longrightarrow_H$.

Given a configuration C of a membrane structure μ over V, the checker H can analyze the words associated with C in the same way as a finite automaton. We say that the configuration C is *accepted* by the checker $H = (Q, V, m, s_0, F, P)$, and we write $H(C) =$ *ok*, if there are a word C' associated with C and a state $s \in F$ such that $s_0 C' \Longrightarrow_H^* C's$.

Now we introduce the variant of membrane systems we are going to investigate in this paper.

A Darwinian membrane system (for short a Darwinian P system) is a construct

$$\Pi = (O, \mu, M_1, M_2, \dots, M_m, R_1, R_2, \dots, R_m, i_0, H_1, H_2, \dots, H_n, tr),$$

where:

1. O is the alphabet of *objects*,
2. μ is a *membrane structure* (of degree m, with the membranes labeled in a one-to-one manner with non-negative integers, for the remaining part of this paper by $1, 2, \dots, m$),
3. for $1 \leq i \leq m$, M_i is a multiset of objects initially present in the i-th region of μ,
4. for $1 \leq i \leq m$, R_i is the set of *evolution rules* associated with the i-th region of μ, where an evolution rule is of the form $a \rightarrow u$, where $a \in O$, $u \in (O \times \{here, out, in\})^*$,
5. $i_0 \in \{1, 2, \dots, m\}$ indicates the *output membrane* of the system,
6. for $1 \leq i \leq n$, $H_i = (Q_i, O, m, s_{0,i}, F_i, P_i)$ is a checker,
7. *tr* is a regular language over $\{H_1, H_2, \dots, H_n\}$.

We now explain the functioning of a Darwinian P system. Let $C = (K_1, K_2, \ldots, K_m)$ and $D = (K'_1, K'_2, \ldots, K'_m)$ be two configurations of μ. A transition from C to D is performed with respect to the sets $R_1, R_2, \ldots, R_m$ of evolving rules (written as $C \Longrightarrow D$) in the following way. Any R_i is applied in a maximally parallel way to the letters of K_i, i.e. for any letter a in K_i we choose nondeterministically a rule $a \to u$ of R_i and put into K_i all letters from u with an assignment *here*, put nondeterministically any letter of u with assignment *in* into a set K'_j where j is an inner region of i, and all letters of u with assignment *out* into K'_l where l is the outer region of i; if there is no such rule $a \to u \in R_i$, then we put into K'_i as many occurrences of a as we have in K_i.

A sequence of transitions (hence a *computation*) is *successful* if it halts, and all its configurations are accepted by the checkers indicated by a sequence of checkers generated by the expression tr. Formally, a computation

$$C_0 \Longrightarrow C_1 \Longrightarrow \ldots \Longrightarrow C_t$$

is successful, if no further rule can be applied to C_t, and there is a sequence $H_{i_1} H_{i_2} \ldots H_{i_t}$ in tr such that $H_{i_s}(C_s) = ok$ for all $1 \leq s \leq t$.

For a Darwinian membrane system Π as above, we define two sets generated by Π. We set

$$Ps(\Pi) = \{\pi_O(K_{i_0}) \mid (M_1, M_2, \ldots, M_m) \Longrightarrow \ldots \Longrightarrow (K_1, K_2, \ldots, K_m) \text{ is a successful computation}\}$$

and

$$N(\Pi) = \{l(K_{i_0}) \mid (M_1, M_2, \ldots, M_m) \Longrightarrow \ldots \Longrightarrow (K_1, K_2, \ldots, K_m) \text{ is a successful computation}\},$$

i.e., we consider the sets of Parikh vectors or lengths of multisets contained in the output region at the end of a successful computation.

The size of a Darwinian P system can be describe by three parameters: the number of membranes/regions, the number of checkers, and the size of the checkers where the size of a checker is given by the number of its states. With respect to these parameters we define the sets $PsDP(m, n, r)$ and $NDP(m, n, r)$ of all sets $Ps(\Pi)$ and $N(\Pi)$, respectively, which can be generated by Darwinian P systems Π with at most m regions in its membrane structures, at most n checkers, and with checkers of size at most r. If we do not bound some of these parameters, then we use the same notation but we replace the bounds m or n or r by $*$.

The following statements have been proven in [1].

Theorem 1. *$PsRE = PsDP(1, *, 3) = PsDP(1, 1, *)$ and $NRE = NDP(1, *, 3) = NDP(1, 1, *)$.*

4 Restricting the Number and the Size of Checkers

By Theorem 1, there is no loss of generative power if one simultaneously restricts the number of regions and checkers or the number of regions and the size of the checkers to small numbers. In this section we shall add the third possible result in this direction. We restrict simultaneously the number and the size of the checkers.

Theorem 2. i) $PsRE = PsDP(*,1,6) = PsDP(*,2,4) = PsDP(*,3,3) = PsDP(*,5,2)$,
ii) $NRE = NDP(*,1,6) = NDP(*,2,5) = NDP(*,3,3) = NDP(*,5,2)$.

Proof. We only give a proof for i); the proof of ii) follows completely the same arguments.

We first prove that $PsRE = PsDP(*,3,3)$.

Let L be a recursively enumerable language, and let $G = (N,T,S,M,F)$ be a matrix grammar with appearance checking in binary normal form as given in Section 2 such that $L(G) = L$. We shall also use the notation given there.

We construct the Darwinian P system

$$\Pi = (O, \mu, \{X, A\}, \emptyset, \emptyset, \ldots, \emptyset, R_0, R_1, \ldots, R_{q_1}, 0, H_1, H_2, H_3, tr),$$

where $(S \to XA)$ is the only matrix of type 1 of G and

$$\begin{aligned}
O &= N_1 \cup N_2 \cup T \cup \bigcup_{X\in N_1} \bigcup_{i=1}^{q_1} \{X_i, X_i'\} \cup \bigcup_{A\in N_2} \bigcup_{i=1}^{q_1} \{A_i, A_i'\} \cup \bigcup_{X\in N_1} \bigcup_{i=q_1+1}^{q} \{X_i'\}\,,\\
\mu &= [_0[_1[_2\ldots[_{q_1-1}[_{q_1}\ ]_{q_1}]_{q_1-1}\ldots]_2]_1]_0\,,\\
R_0 &= \{X \to (X_i, in) \mid X = lhs(r_i),\ 1 \le i \le q_1\}\\
&\quad \cup \{A \to (A_i, in) \mid A = lhs(s_i),\ 1 \le i \le q_1\}\\
&\quad \cup \{X \to (X_i', in) \mid X = lhs(r_i),\ q_1+1 \le i \le q\}\\
&\quad \cup \{B \to B \mid B \in N_2 \cup T\}\,,\\
R_1 &= \{X_1 \to X_1' \mid X \in N_1\} \cup \{A_1 \to A_1' \mid A \in N_2\}\\
&\quad \cup \{X_j \to (X_j, in) \mid X \in N_1,\ 2 \le i \le q_1\}\\
&\quad \cup \{A_j \to (A_j, in) \mid A \in N_2,\ 2 \le i \le q_1\}\\
&\quad \cup \{A_i' \to (M(x), out) \mid x = rhs(s_i),\ 1 \le i \le q_1\}\,,\\
&\quad \cup \{X_i' \to (Y, out) \mid Y = rhs(r_i),\ 1 \le i \le q\}\,,\\
R_i &= \{X_i \to X_i' \mid X \in N_1\} \cup \{A_i \to A_i' \mid A \in N_2\}\\
&\quad \cup \{X_j \to (X_j, in) \mid X \in N_1,\ i+1 \le j \le q_1\}\\
&\quad \cup \{A_j \to (A_j, in) \mid A \in N_2,\ i+1 \le j \le q_1\}\\
&\quad \cup \{X_j' \to (X_j', out) \mid X \in N_1,\ i \le j \le q_1\}\\
&\quad \cup \{A_j' \to (A_j', out) \mid A \in N_2,\ i \le j \le q_1\} \text{ for } 2 \le i \le q_1 - 1,\\
R_{q_1} &= \{X_{q_1} \to X_{q_1}' \mid X \in N_1\} \cup \{A_{q_1} \to A_{q_1}' \mid A \in N_2\}
\end{aligned}$$

$$\begin{aligned}
&\cup \{X'_{q_1} \to (X'_{q_1}, out) \mid X \in N_1\} \cup \{A'_{q_1} \to (A'_{q_1}, out) \mid A \in N_2\},\\
H_1 &= (\{s_0, s_1, s_2\}, O, q, s_0, \{s_1, s_2\},\\
&\quad \bigcup_{A \in N_2} \bigcup_{i=1}^{q_1} \bigcup_{j=1}^{q_1} \{s_0[_jA_i]_j \to [_jA_i]_j s_1,\ s_2[_jA'_i]_j \to [_jA'_i]_j s_2\}\\
&\quad \cup \bigcup_{X \in N_1} \bigcup_{i=1}^{q_1} \bigcup_{j=1}^{q_1} \{s_1[_jX_i]_j \to [_jX_i]_j s_1,\ s_0[_jX'_i]_j \to [_jX'_i]_j s_2\}\\
&\quad \cup \{s_i[_0C]_0 \to [_0C]_0 s_i \mid C \in N_2 \cup T,\ 1 \le i \le 2\}),\\
H_2 &= (\{s_0, s_3, s_4\}, O, q_1, s_0, \{s_3, s_4\},\\
&\quad \{s_0[_1X'_i]_1 \to [_1X'_i]_1 s_3 \mid X \in N_1,\ q_1 + 1 \le i \le q_2\}\\
&\quad \cup \{s_0[_jX'_i]_j \to [_jX'_i]_j s_4 \mid X \in N_1,\ q_2 + 1 \le i \le q\}\\
&\quad \cup \{s_{i+2}[_0C]_0 \to [_0C]_0 s_{i+2} \mid C \in (N_2 \setminus \{B_i\}) \cup T\},\ 1 \le i \le 2\}),\\
H_3 &= (\{s_0, s_5\}, O, q_1, s, \{s_5\}, \bigcup_{A \in N_1 \cup N_2 \cup T} \{s_0[_0A]_0 \to [_0A]_0 s_5, s_5[_0A]_0 \to [_0A]_0 s_5\}),\\
tr &= ((H_2 \cup \bigcup_{i=1}^{q_1} H_1^{2i}) H_3)^*.
\end{aligned}$$

First we note that if a computation halts with the configuration $C = (K_0, K_1, \dots, K_{q_1})$, then $H_3(C) = ok$ holds. Since H_3 only successfully checks configurations where all letters are in region 0, we have $C = (K_0, \emptyset, \emptyset, \dots, \emptyset)$. If K_0 contains a letter from $N_1 \cup N_2$, then there is a rule in R_0 which can be applied. Thus, in a halting configuration all elements of K_0 are in T. Hence the halting condition of Π is only obtained if the terminating condition of G is satisfied.

Now let us assume that we have a configuration $C = (K_0, K_1, K_2, \dots, K_{q_1})$ such that $H_3(C) = ok$. As above we have $C = (K_0, \emptyset, \emptyset, \dots, \emptyset)$. This situation also holds for the initial configuration $C_0 = (\{A, X\}, \emptyset, \emptyset, \dots, \emptyset)$. Furthermore, in both cases the next checker is H_1 or H_2.

Let $C \Longrightarrow C'$. Let us first assume that H_2 is used. We have $H_2(C') = ok$ if and only if the configuration $C' = (K_0 \setminus \{X\}, \{X'_i\}, \emptyset, \emptyset, \dots, \emptyset)$ satisfies $A \notin K_0$, $q_1 + 1 \le i \le q$, and $m_i = (X \to Y, A \to \#)$. If the computation is not blocked, then $H_3(C'') = ok$ has to hold for C'' with $C' \Longrightarrow C''$. This implies $C'' = ((K_0 \setminus \{X\}) \cup \{Y\}, \emptyset, \emptyset, \dots, \emptyset)$. This means that $C \Longrightarrow C' \Longrightarrow C''$ holds if and only if $w \Longrightarrow_{m_i} w''$ where $K_0 = M(w)$ and $(K_0 \setminus \{X\}) \cup \{Y\} = M(w'')$, i.e., up to the order of elements a derivation step of G is simulated.

If H_1 is used, then we get

$$C_1 = (K_0 \setminus \{A, X\}, \{A_i, X_j\}, \emptyset, \emptyset, \dots, \emptyset) \tag{1}$$

or

$$C_1 = (K_0 \setminus \{A\}, \{A_i\}, \emptyset, \emptyset, \dots, \emptyset)\,. \tag{2}$$

We first discuss (1). By the definition of tr we now have to choose a k, $1 \le k \le q_1$, and $(2k-1)$-times in succession to apply H_1 for the checking of the sentential forms. Assume that this leads to the derivation

$$C \Longrightarrow C_1 \Longrightarrow C_2 \Longrightarrow C_3 \Longrightarrow \ldots \Longrightarrow C_{2k} \Longrightarrow C_{2k+1} \tag{3}$$

with $H_1(C_u) = ok$ for $1 \leq u \leq 2k$ and $H_3(C_{2k+1}) = ok$. Let $i < j$ and $i \leq k$. Then C_{i+1} contains a letter A'_i in region i and X_j in region i+1. This contradicts $H_1(C_{i+1}) = ok$. Analogously we get a contradiction if $j < i$ and $j \leq k$. If $k < i$ and $k < j$, then C_{2k+1} contains two of the letters A_i, A'_i, X_j, X'_j in some regions different from region 0. This is a contradiction to $H_3(C_{2k+1}) = ok$. Thus, the only possible situation is $i = j = k$. Moreover, no letter in region 0 is changed during the derivation (3). This leads to $C_{2k+1} = ((K_0 \setminus \{A, X\}) \cup M(x) \cup \{Y\}, \emptyset, \emptyset, \ldots, \emptyset)$ where $m_i = (X \to Y, A \to x)$, i.e., we have simulated the application of a matrix of G, again.

In the case of (2) we can analogously show that the derivation is blocked.

Therefore $Ps(\Pi) = Ps(L(G)) = Ps(L)$.

Since any checker has at most 3 states and we use (at most) 3 checkers (if no symbol is used in rules of F, then we have only two checkers), $Ps(L) \in PsDP(*, 3, 3)$. Thus $PsRE \subseteq PsDP(*, 3, 3)$. The converse inclusion is obvious. Hence $PsRE = PsDP(*, 3, 3)$.

If we construct from H_2 and H_3 a checker

$$\begin{aligned} H'_2 = (&\{s_0, s_3, s_4, s_5\}, O, q_1, s_0, \{s_3, s_4\}, \\ &\{s_0[_1X'_i]_1 \to [_1X'_i]_1 s_3 \mid X \in N_1,\ q_1 + 1 \leq i \leq q_2\} \\ &\cup \{s_0[_jX'_i]_j \to [_jX'_i]_j s_4 \mid X \in N_1,\ q_2 + 1 \leq i \leq q\} \\ &\cup \{s_i[_0C]_0 \to [_0C]_0 s_i \mid C \in (N_2 \setminus \{B_{i-2}\}) \cup T, 3 \leq i \leq 4\} \\ &\cup \bigcup_{A \in N_1 \cup N_2 \cup T} \{s_0[_0A]_0 \to [_0A]_0 s_5,\ s_5[_0A]_0 \to [_0A]_0 s_5\}), \end{aligned}$$

then we get as above that the Darwinian P system

$$\Pi' = (O, \mu, \{X, A\}, \emptyset, \emptyset, \ldots, \emptyset, R_0, R_1, \ldots, R_{q_1}, 0, H_1, H'_2, ((H'_2 \cup \sum_{i=1}^{q_1} H_1^{2i}) H'_2)^*)$$

which generates $Ps(L)$. Consequently, $PsDP(*, 2, 4) = PsRE$.

If we combine in the same way H_1, H_2, and H_3, we get a checker with 6 states and then $PsDP(*, 1, 6) = PsRE$.

Finally, in an analogous way we can divide H_1 and H_2 into four checkers H'_1, H''_1, H''_2, and H'''_2 with the state sets $\{s_0, s_1\}$, $\{s_0, s_2\}$, $\{s_0, s_3\}$, and $\{s_0, s_4\}$. We set

$$tr = (H''_2 \cup H'''_2 \cup \bigcup_{i=1}^{q_1} (H'_1)^i (H''_1)^i) H_3$$

and get $PsDP(*, 5, 2) = PsRE$. $\square$

5 A Sharp Bound for the Size of Checkers

In [1] it has been proved that Darwinian P system with one region and checkers with at most 3 states are sufficient in order to generate Parikh set or length set of any recursively enumerable language. We improve this result by showing that the bound for the size of the checkers can be decreased to 2.

Theorem 3. *$PsRE = PsDP(1,*,2)$ and $NRE = NDP(1,*,2)$.*

Proof. Let L be a recursively enumerable language. Then L can be generated by a matrix grammar G with appearance checking in binary normal form as given in Section 2.

We construct the Darwinian P system

$$\Pi = (O, [_1\ ]_1, \{X, A\}, R_1, 1, H_1, H_1', \ldots, H_{q_1}, H_{q_1}', H_{q_1+1}, H_{q_1+2}, \ldots, H_q, H, tr)$$

where $(S \to XA)$ is the initial matrix of G and

$$\begin{aligned}
O &= N_1 \cup N_2 \cup \bigcup_{i=1}^{q_1} \{X_i, X_i', A_i, A_i\} \cup \bigcup_{i=q_1+1}^{q} \{X_i\}\,,\\
R_1 &= \{X \to X_i \mid X = lhs(r_i),\ 1 \le i \le q\} \cup \{A \to A_i \mid A = lhs(s_i),\ 1 \le i \le q_1\}\\
&\quad \cup \bigcup_{i=1}^{q_1} \{X_i \to X_i',\ A_i \to A_i',\ X_i' \to rhs(r_i),\ A_i' \to rhs(s_i)\}\\
&\quad \cup \bigcup_{i=q_1+1}^{q} \{X_i \to rhs(s_i)\} \ \cup\ \{B \to B \mid B \in N_2\}\,,\\
H_i &= (\{s_0, s_1\}, O, 1, s_0, \{s_1\},\\
&\quad \{s_0[_1X_i]_1 \to [_1X_i]_1 s_1\} \cup \{s_1[_1C]_1 \to [_1C]_1 s_1 \mid C \in N_2 \cup T \cup \{A_i\}\})\\
&\quad \text{for } 1 \le i \le p\,,
\end{aligned}$$

$$\begin{aligned}
H_i' &= (\{s_0, s_1\}, O, 1, s_0, \{s_1\},\\
&\quad \{s_0[_1A_i']_1 \to [_1A_i']_1 S_1\} \cup \{s_1[_1C]_1 \to [_1C]_1 s_1 \mid C \in N_2 \cup T \cup \{X_i'\}\})\\
&\quad \text{for } 1 \le i \le q_1\,,\\
H_i &= (\{s_0, s_1\}, O, 1, s_0, \{s_1\},\\
&\quad \{s_0[_1X_i]_1 \to [_1X_i]_1 s_1\} \cup \{s_1[_1C]_1 \to [_1C]_1 s_1 \mid C \in (N_2 \setminus \{B_1\}) \cup T\})\\
&\quad \text{for } q_1 + 1 \le i \le q_2\,,\\
H_i &= (\{s_0, s_1\}, O, 1, s_0, \{s_1\},\\
&\quad \{s_0[_1X_i]_1 \to [_1X_i]_1 s_1\} \cup \{s_1[_1C]_1 \to [_1C]_1 s_1 \mid C \in (N_2 \setminus \{B_2\}) \cup T\})\\
&\quad \text{for } q_2 + 1 \le i \le q\,,\\
H &= (\{s_0\}, O, 1, s_0, \{s_0\}, \{s_0[_1C]_1 \to [_1C]_1 s_0 \mid A \in N_1 \cup N_2 \cup T\})\,,\\
tr &= ((\bigcup_{i=1}^{q_1} H_i H_i' \cup \bigcup_{q_1+1}^{q} H_i) H)^*\,.
\end{aligned}$$

Let us assume that we have a configuration $C = (\{X\} \cup M(w))$ with some $X \in N_1$ and some word $w \in (N_2 \cup T)^*$ (the initial configuration $(\{X, A\})$ has this form). Then we have $H(C) = ok$ and $H_i(C) \neq ok$ since the multiset contains no indexed letter.

If we apply R_1 to C, we produce a configuration $C' = (K)$ where K contains a letter X_i, $1 \leq i \leq q$, and some or no letters A_j, $1 \leq j \leq q_1$. By the definition of tr we have to continue with an application of H_i for some i, $1 \leq i \leq q$.

Assume that $1 \leq i \leq q_1$. If we use a checker H_t, $1 \leq t \leq q$, $t \neq i$, then $H_t(C') \neq ok$. Therefore, we have to use the checker H_i which only accepts C' if K contains – beside X_i – some letters A_i and all other letters of K are non-indexed. In the following derivation step $C' \Longrightarrow C''$ we have to replace X_i by X_i' and all A_i by A_i'. Moreover, since $H_i'(C'') = ok$ has to hold, we check that – besides X_i' – exactly one occurrence of A_i' exists and that all other letters remain unchanged. Now we replace X_i' by $rhs(r_i)$ and A_i' by $rhs(s_i)$. Furthermore, since $H(C''') = ok$ has to hold, all other letters remain unchanged, again. Thus we have simulated the application of the matrix m_i.

If $q_1 + 1 \leq i \leq q_2$, then $H_t(C) \neq ok$ for all t with $1 \leq t \leq q$ and $t \neq i$ and the check by H_i is successful only if X has been replaced by an indexed letter and that no letter B_1 occurs in the configuration. Then we replace X_i by $Y = rhs(r_i)$ which simulates the application of $m_i = (X \to Y, B_1 \to \#)$ in the appearance checking mode.

Analogously, if $q_2 + 1 \leq i \leq q$, then we simulate an application of $m_i = (X \to Y, B_2 \to \#)$.

If the configuration does not contain a symbol of N_1, then the derivation is blocked since the application of R_1 leads to a configuration which is rejected by all checkers H_i, $1 \leq i \leq q$, and one of these checkers has to be used.

Thus – up to the order of the letters – we simulate a derivation in G which proves $Ps(\Pi) = Ps(L(G)) = Ps(L)$ and $N(\Pi) = N(L(G)) = N(L)$.

By the definition of Π, $Ps(L) \in PsDP(1, *, 2)$ and $N(L) \in NDP(1, *, 2)$. Thus $PsRE \subseteq PsDP(1, *, 2)$ and $NRE \subseteq NDP(1, *, 2)$. The converse inclusions are obvious. Hence $PsRE = PsDP(1, *, 2)$ and $NRE = NDP(1, *, 2)$. □

We now discuss Darwinian P systems where the checkers have only one state. We note that such a checker with state s has only rules of the form $s[_iA]_i \to [_iA]_is$. This means that the letter A is allowed in the i-th region. Therefore, with any region i, any checker associates a set of letters allowed in this region.

Lemma 2. *For any ET0L system $G = (V, T, P_1, P_2, \ldots, P_r, w_0)$, there is a Darwinian P system H such that all checkers of H have only one state and $Ps(\Pi) = Ps(L(G))$ and $N(\Pi) = N(L(G))$.*

Proof. We set $V' = \{b' \mid b \in V\}$. With any word $w \in V^*$, $w = b_1b_2 \ldots b_t$, $t \geq 0$, $b_i \in V$ for $1 \leq i \leq t$, we associate the word $w' = b_1'b_2' \ldots b_t' \in (V')^*$.

We construct the Darwinian P system

$$\Pi = (V \cup V', \mu, M(w_0'), \emptyset, \emptyset, \ldots, \emptyset, P_0, P_1', P_2', \ldots, P_r', 0, H_0, H_1, \ldots, H_{r+1}, tr)$$

where:

$$\begin{aligned}
\mu &= [_0[_1\ ]_1[_2\ ]_2 \cdots [_r\ ]_r]_0\,, \\
P_0 &= \{b' \to (b', in) \mid b \in V\} \cup \{b' \to (b, here) \mid b \in T\}\,, \\
P_i' &= \{b' \to (b_1', out)(b_2', out) \dots (b_t', out) \mid b \to b_1 b_2 \dots b_t \in P_i\} \text{ for } 1 \le i \le r, \\
H_i &= (\{s\}, O, m, s, \{s\}, \{s[_i b']_i \to [_i b']_i s \mid b \in V\}\}) \text{ for } 0 \le i \le r, \\
H_{r+1} &= (\{s\}, O, m, s, \{s\}, \{s[_0 a]_0 \to [_0 a]_0 s \mid a \in T\})\,, \\
tr &= (\{H_1, H_2, \dots, H_r\} H_0)^* H_{r+1}\,.
\end{aligned}$$

Obviously, for $0 \le i \le r$ and a configuration C, $H_i(C) = ok$ if and only if all letters of C are in the i-th region, and $H_{r+1}(C) = ok$ if and only if all letters of C are in the region 0 and belong to T. Thus, any successful derivation is of the form

$$D_0 \Longrightarrow C_{i_1} \Longrightarrow D_1 \Longrightarrow C_{i_2} \Longrightarrow D_2 \dots \Longrightarrow C_{i_k} \Longrightarrow D_k \Longrightarrow D$$

where:

- $D_0 = (M(w_0'), \lambda, \lambda, \dots, \lambda)$,
- for $0 \le j \le k-1$, the multisets of C_{i_j} in the regions $t \ne i_j$ are empty and the multiset of C_{i_j} in the region i_j is $M(w_i')$,
- for $0 \le j \le k$, the multisets of D_j in the regions $p \ne 0$ are empty and the multiset of D_j in the region 0 is $M(w_{i+1}')$,
- for $0 \le i \le k-1$, $w_i \Longrightarrow_{P_{i_j}} w_{i+1}$ in G since the evolution rules of P_{i_j}' simulate the application of P_{i_j},
- $D = (w_k, \lambda, \lambda, \dots, \lambda)$ and $w_k \in T^*$ obtained by the parallel application of rules of the form $b' \to (b, here)$ to all elements of w_k'.

Thus we have simulated the derivation $w_0 \Longrightarrow w_1 \Longrightarrow \dots \Longrightarrow w_k$ in G with $w_k \in T^*$. Therefore $Ps(\Pi) = Ps(L(G))$ and $N(\Pi) = N(L(G))$. □

Lemma 3. *Let Π be a Darwinian P system such that all checkers of Π have only one state. Then there is an ET0L system G such that $N(L(G)) = N(\Pi)$ and $Ps(L(G)) = Ps(\Pi)$.*

Proof. Let

$$\Pi = (O, \mu, K_1, K_2, \dots, K_m, R_1, R_2, \dots, R_m, i_0, H_1, H_2, \dots, H_n, tr)$$

be a Darwinian P system where any checker H_i, $1 \le i \le n$, has only one state. Thus, for $1 \le i \le n$ and $1 \le j \le m$, there is a set $M_{i,j} \subseteq O$ such that H_i only checks whether or not all letters in the j-th region belong to $M_{i,j}$.

For $1 \le i \le m$, let

$$O^{(i)} = \{b^{(i)} \mid b \in O\}$$

(by the superscript i we want to indicate that the letter is in the i-th region). For a word $w = b_1 b_2 \dots b_t$, $b_i \in O$ for $1 \le i \le t$, and $1 \le j \le m$, we set

$w^{(j)} = b_1^{(j)} b_2^{(j)} \ldots b_t^{(j)}$. With a configuration $C = (M_1, M_2, \ldots, M_m)$ we associate the word

$$w(C) = (w(M_1))^{(1)}(w(M_2))^{(2)} \ldots (w(M_m))^{(m)} \in \bigcup_{i=1}^{m} O^{(i)} .$$

Conversely, with a word $w \in \bigcup_{i=1}^{m} O^{(i)}$ we associate the configuration

$$C(w) = (M(w_1), M(w_2), \ldots, M(w_m)) ,$$

where, for $1 \leq i \leq m$, w_i is the scattered subword of w built by all letters from $O^{(i)}$ in w.

Assume that $s_1, s_2, \ldots, s_k$ are the inner regions of the i-th region, and let t be the outer region of the i-th region. Then we define the substitution τ_i by

$$\begin{aligned}&\tau_i((b, here)) = b^{(i)},\ \tau_i((b, out)) = b^{(t)} \text{ and}\\ &\tau_i((b, in)) = \{b^{(s_1)}, b^{(s_2)}, \ldots, b^{(s_k)}\} \text{ for } b \in O.\end{aligned}$$

With a rule $p = b \to w \in R_i$, we associate the set R_p of all rules $b^{(i)} \to w'$ with $w' \in \tau_i(w)$.

We now define the regularly controlled ET0L system

$$G = (V, O, P, P_1, P_2, \ldots, P_n, P', w((K_1, K_2, \ldots, K_m)), R)$$

with

$$N = O \cup \{\#\} \cup \bigcup_{i=1}^{m} O^{(i)}$$

(# is a trap symbol, once introduced, it is never removed),

$$P = \{a \to a \mid a \in O\} \cup \{\# \to \#\} \cup \bigcup_{i=1}^{m} \{b^{(i)} \to b^{(i)} \mid b \notin dom(R_i)\} \cup \bigcup_{i=1}^{m} \bigcup_{p \in R_i} R_p$$

(the first three sets ensure the completeness condition for ET0L systems; the sets R_p ensure that the application of P simulates a derivation step of Π),

$$\begin{aligned}P_j = &\{a \to a \mid a \in 0\} \cup \{\# \to \#\}\\ &\cup \bigcup_{i=1}^{m} \{b^{(i)} \to b^{(i)} \mid b \in M_{i,j}\} \cup \bigcup_{i=1}^{m} \{b^{(i)} \to F \mid b \notin M_{i,j}\}\end{aligned}$$

(the application of table P_i introduces no letter # only if all letters in the i-th region belong to the set $M_{i,j}$, i.e., if P_i is applied to w, then no # is introduced if and only if $C(w)$ contains in the j-th region only letters from $M_{i,j}$; thus it simulates the check done by H_i),

$$P' = \{a \to a \mid a \in O\} \cup \{\# \to \#\} \cup \{b^{(i)} \to \lambda \mid b \notin dom(R_i), 1 \leq i \leq m, i \neq i_0\}$$
$$\cup \{b^{(i_0)} \to a \mid b \notin dom(R_{i_0})\} \cup \bigcup_{i=1}^{m} \{b^{(i)} \to \# \mid b \in dom(R_i)\}$$

(the application of P' does not introduce the letter # if it is applied to a word corresponding to a halting configuration; moreover, it filters out the word in the region i_0 in a non-indexed version),

$$R = \{PP_{i_1}PP_{i_2} \ldots PP_{i_r}P' \mid H_{i_1}H_{i_2} \ldots H_{i_r} \in tr\}$$

(since tr is regular, R is regular, too). By the definition of R we alternately simulate a derivation step of H and the checkers which have to be used at those places. Thus, we simulate a derivation in Π. Finally, we check by P' whether or not we obtained a halting configuration. Thus we stop with a word w such $w = w(M_{i_0})$ for some halting configuration $C = (M_1, M_2, \ldots, M_m)$ or with a word which contains the letter # (and does not belong to $L(G)$.

Consequently, $Ps(L(G)) = Ps(\Pi)$ and $N(L(G)) = N(\Pi)$.

By Lemma 1, we obtain $Ps(\Pi) = Ps(L(G'))$ and $N(\Pi) = N(L(G'))$ for some ET0L system G'. □

As a consequence from the two preceding lemmas we get the following statement.

Corollary 1. *NDP*$(*, *, 1) =$ *NET0L and PsDP*$(*, *, 1) =$ *PsET0L.*

By Corollary 1 the assertion of Theorem 3 with respect to the size of the checkers (as well as with respect to the number of regions) is optimal.

References

1. E. Csuhaj-Varjú, C. Martín-Vide, Gh. Păun, A. Salomaa: From Watson-Crick L-systems to Darwinian P systems. *Natural Computing*, 2 (2003), 299–218.
2. J. Dassow, Gh. Păun: *Regulated Rewriting in Formal Language.* Springer-Verlag, 1989.
3. J. Dassow, Gh. Păun: On the power of membrane computing. *J. Universal Comp. Sci.*, 5 (1999), 33–49.
4. R. Freund, Gh. Păun: On the number of non-terminal symbols in graph-controlled, programmed and matrix grammars. In *Machines,Computations, and Universality 2001* (M. Margenstern, Y. Rogozhin, eds.), LNCS 2055, Springer-Verlag, 2001, 214–225.
5. S. Ginsburg, G. Rozenberg: T0L schemes and control sets. *Inform. Control*, 27 (1974), 109–125.
6. Gh. Păun: *Membrane Computing: An Introduction.* Springer-Verlag, 2002.
7. G. Rozenberg, A. Salomaa: *The Mathematical Theory of L Systems.* Academic Press, 1980.
8. G. Rozenberg, A. Salomaa, eds.: *Handbook of Formal Languages.* Vol. I–III. Springer-Verlag, 1997.