P automata *†

Erzsébet CSUHAJ-VARJÚ
and
György VASZIL

Computer and Automation Research Institute
Hungarian Academy of Sciences
Kende utca 13-17, 1111 Budapest, Hungary
{csuhaj,vaszil}@sztaki.hu

Abstract

In this paper we introduce the notion of a P automaton with one-way communication, a concept related both to P systems and the traditional concept of automata. A P automaton with one-way communication is a purely communicating accepting P system, where the membranes are allowed only to consume multisets of symbols from their parent membranes, under some conditions. The result of the computation in these systems is the set of multiset sequences consumed by the skin membrane, supposing that the P automaton started functioning in a so-called initial state and after consuming the multiset sequence entered a so-called final state. As a result, we show that any recursively enumerable language can be represented as an image of the set of accepted input multiset sequences of a P automaton, under a certain projection.

1 Introduction

P systems were introduced by Gheorghe Păun in [4] as membrane structures consisting of membranes hierarchically embedded in the outermost skin membrane. Each membrane encloses a region possibly containing other

*Research supported in part by the Hungarian Scientific Research Fund "OTKA" grants no. T 029615, F 037567, and by the MolCoNet project, IST-2001-32008.

†Accepted for the MolCoNet Workshop on Membrane Computing, Curtea de Argeş, 2002.

membranes. A region separated by a membrane may contain other membranes and also specific objects and operators (used in the multiset sense). These operators can be of different types, they can work on the multisets of objects in the regions but also can provide the possibility of transferring the objects and the operators from one region to another one. Since 1998 the theory of P systems or membrane computing has proved to be a successful area of unconventional models of computation: a lot of variants of the basic notion have been introduced and studied proving the power of the framework; the interested reader is referred to [5, 6, 9] for basic information, and to the book [8] to be published soon for a summary of the achievements and open problems in the area. The reader also can consult the P systems web page with a lot of down-loadable papers and information [10].

By introducing the notion of a P automaton, our paper attempts to build a bridge between the theory of P systems and automata theory. According to the original model of a P system, at the beginning of the computation the membranes and the regions of the P system are initialized, that is, they contain multisets of objects and operators. Then, under the given (or evolving) operations, the contents of the regions (the multisets of objects contained in them) change, and these changes define the computation. The result of the computation can be, for example, a multiset or set of objects filtered at some previously fixed stage of the computation.

Our model applies another approach. In this case, the membranes of the system are allowed only to communicate with each other, that is, the only allowed activity is to transfer a multiset of objects from one region to another one, supposing that the regions satisfy some prescribed conditions. The skin membrane is allowed to consume multisets of objects from outside, and this is the only way how new objects can enter the membrane system. The computation starts at some initial state of the P system, that is, the initial contents of the different regions are prescribed, and it ends when the P system is in a final state, that is, the regions contain some previously prescribed multisets of objects. The result of the computation is the sequence of multisets of objects which served as input multisets consumed by the skin membrane under an accepting computation, that is, when the P automaton started its functioning from the initial state and entered a final state.

It is easy to see that the notion is a related concept to the customary notion of automaton: the input multiset sequence of the P automaton corresponds to the word read from the input tape and the membranes and the regions with the objects represent both storage tapes and states of an automaton. Moreover, the notion is a natural concept arising from membrane computing: Starting from the initial state, the work of the system is

influenced by the effects of the external world. Finally it enters an accepting configuration which can either be a balanced situation or a previously prescribed configuration.

The idea of P automaton was motivated by two problems raised by Gheorghe Păun. The first, from [7] is the following. "What about the possibility of considering a class of P systems, meant to compute, where no rule for objects evolution appears, but only rules governing object communication from a region to another one". The second, from [8], Problem $Q32$: "What about using P systems as accepting devices?"

In this paper we discuss P automata with one-way communication, that is, each membrane is allowed to ask a multiset of objects (symbols) only from its parent membrane (its direct predecessor node in the tree representing the membrane structure of the system), supposing that the region of the membrane contains a prescribed multiset of objects. Thus, the communication is of type one-way, top-down. Obviously, a P automaton with two-way communication can also be defined, we shall return to these models in future papers.

The concept of P automata raises several interesting problems. A natural question is, what can we say about the accepted input multiset sequences of a P automaton. We give an answer, at the same time demonstrating how these tools can be used for computation, namely, we show that any recursively enumerable language can be represented as an image of the set of accepted input multiset sequences of a one-way P automaton, under a certain projection.

2 Definitions

Throughout the paper we assume the reader to be familiar with the basics of language theory; for further details confer [2]. Let Σ be an alphabet. Let Σ^* be the set of all words over Σ (including the empty word ε). We denote the length of a word $w \in \Sigma^*$ by $|w|$ and the number of occurrences of a symbol $a \in \Sigma$ in w by $|w|_a$.

A multiset of objects M is a pair $M = (V, f)$, where V is an arbitrary (not necessarily finite) set of objects and f is a mapping $f : V \to N$; f assigns to each object in V its multiplicity in M. The set V is called the support of M. The set of all multisets with support V is denoted by V°. If V is a finite set, then M is called a finite multiset.

The number of objects in a finite multiset of objects $M = (V, f)$, the cardinality of M, denoted by $card(M)$, is defined by $card(M) = \sum_{a \in V} f(a)$.

The reader can easily observe that any finite multiset of objects M with support $V = \{a_1, \ldots, a_n\}$ can be represented as a string w over alphabet V with $|w|_{a_i} = f(a_i)$, $1 \leq i \leq n$. Clearly, all words obtained from w by permuting the letters can also represent M.

In the following we will use this type of representation, and we will denote by $[w]$ the finite multiset of objects M with support V represented by word w over V, the empty multiset will be denoted as $[\varepsilon]$.

Let Σ and V be two alphabets with $\Sigma \subseteq V$. A mapping $h : V^\circ \to 2^\Sigma$ defined by $h(M) = V \cap \Sigma$ for finite multiset $M = (V, f)$ is said to be an l-projection of V° to 2^Σ. Mapping h is extended to sequences of finite multisets $M_1 \ldots M_n$, $M_i \in V^\circ$, $1 \leq i \leq n$, in the following way: $h(M_1 \ldots M_n) = \{x_1 \ldots x_n \mid x_i \in h(M_i) \text{ for } h(M_i) \neq \emptyset, x_i = \varepsilon \text{ otherwise }, 1 \leq i \leq n\}$.

Now we recall some basic notions from membrane computing. For further details the reader is referred to [5, 6].

A membrane structure μ is represented by a Venn diagram and it is identified by a string of correctly mathching parentheses, with a unique external pair of parentheses. This external pair of parentheses corresponds to the external membrane, called the skin membrane.

A membrane structure μ can also be represented as a tree. The root of the tree corresponds to the skin membrane, and if a node labelled by k is a direct predecessor of a node labelled by i in the tree, then membrane i is contained in membrane k in the system and there is no other membrane j, $j \neq i, k$, such that membrane i is contained in membrane j and membrane j is contained in membrane k. In this case we say that membrane k is the parent membrane of membrane i and we use notation $par(i) = k$. Membrane i is called a child membrane of membrane k. The skin membrane is always denoted by 1.

As we have mentioned in the Introduction, each membrane encloses a region possibly containing other membranes - prescribed by the membrane structure μ - and also specific objects (used in the multiset sense). Analogously to the membranes, we speak about parent and child regions. The parent region of the skin membrane is the outer region.

In the following we shall define the *contents of a region* as the multiset of all objects that are contained by the region but not contained by any of its child regions. If we consider a multiset of objects which are contained by the region, but not by any of its child regions, then we shall use the term that the region contains this multiset of objects.

The reader can observe that our notation $[w]$, used for denoting a finite multiset of objects M with support V represented by a word w over V is reasonable, it represents a membrane systems consisting only from the skin

membrane with contents identified by w. However, we should note that this notation slightly differs from the customary one in the literature, where brackets $[,]$ are used for describing the membrane structure. The reader should be aware of this difference.

Now we define the notion of a one-way P automaton.

Definition 1 A *one-way P automaton* with n membranes is a construct $\Gamma = (V, \mu, ([w_1], P_1, F_1), \ldots, ([w_n], P_n, F_n)), n \geq 1$, where

- V is an alphabet of objects,
- μ is a membrane structure of n membranes (and n regions),
- $[w_i] \in V^\circ, 1 \leq i \leq n$, is the initial contents (state) of region i, that is, it is the multiset of all objects contained by region i,
- $P_i, 1 \leq i \leq n$, is a finite set of communication rules associated to region i. These rules have the form $x : y \rightarrow in$ where $x, y \in V^+ \cup \{\emptyset\}$. For $x, y \neq \emptyset$ strings x and y represent finite multisets $[x], [y] \in V^\circ$. The rule $x : y \rightarrow in$ means the following: For $x, y \neq \emptyset$, if $[x]$ is c ontained in region i and $[y]$ is contained in its parent region, then the objects of $[y]$ must leave the parent region and enter region i. If $x = \emptyset$, then the region i must be empty, if $y = \emptyset$, then no object is requested form the parent region.
- $F_i \subseteq V^\circ, 1 \leq i \leq n$, is either a finite set of multisets over V or $F_i = \emptyset$, with $F_j \neq \emptyset$ for at least one j, $1 \leq j \leq n$. F_i is called the set of final states of region i.

Notice that in the case of communication rules for the same condition x more than one requests areallowed, that is, the rule set of region i can have rules $x : y \rightarrow in$ and $x : z \rightarrow in$ for $y \neq z$. Note also that the productions require two conditions to be satisfied: the multisets represented by x and y must be contained by region i and by the parent region, $par(i)$, respectively.

Definition 2 The n-tuple of multisets of objects present in the n regions of Γ describes the *configuration* of the system. $([w_1], \ldots, [w_n])$ is the initial configuration.

Now we define the way of functioning of the one-way P automaton. At any moment of time, each region asks its parent (the skin membrane asks the outer region) for a multiset of objects by applying one of its rules chosen

non-deterministically. If all requests are satisfied, the system enters a new configuration given by the state (contents) of the n regions. If any of the requests can not be satisfied, or there is no production to be applied at a region, then the system aborts.

Definition 3 The transition mapping of a one-way P automaton is a partial mapping $\delta : V^\circ \times (V^\circ)^n \rightarrow (V^\circ)^n$. For two configurations $([u_1], \ldots, [u_n])$, $([u'_1], \ldots, [u'_n])$ and a multiset $[u] \in V^\circ$,

$$\delta([u], ([u_1], \ldots, [u_n])) = ([u'_1], \ldots, [u'_n])$$

holds if for all $i, 1 \leq i \leq n$, there exists a rule $x_i : y_i \rightarrow in \in P_i$ with $[x_i] \subseteq [u_i]$ for $x_i \neq \emptyset$ and $[u_i] = [\varepsilon]$ for $x_i = \emptyset$, and $Y_j \subseteq [u_{par(j)}]$, $2 \leq j \leq n$, $Y_1 = [u]$, and

$$[u'_i] \quad = \quad [u_i] \cup Y_i - \bigcup_{i=par(j)} Y_j, \tag{1}$$

for $Y_i = [y_i]$ if $y_i \neq \emptyset$ and $Y_i = [\varepsilon]$ if $y_i = \emptyset$, $1 \leq i \leq n$.

If (1) cannot be satisfied, then $\delta([u], ([u_1], \ldots, [u_n]))$ is undefined, the system aborts.

The sequence of configurations obtained in the above manner is a computation. The computation ends if the system is in a final configuration, that is, in a configuration having all regions, $1 \leq i \leq n$, with $F_i \neq \emptyset$ in a final state $[\alpha_i] \in F_i$. If for some j, $F_j = \emptyset$, then the automaton can reach a final state regardless of the contents of region j.

The reader can observe that the sequence of multisets of objects requested by the skin membrane of the P automaton from the outer region can be considered as an input sequence, and the regions are related to tapes which change their contents in parallel depending on the input. Thus, we define a sequence of multisets of objects accepted by the P automaton as an input sequence which, after being consumed by the skin membrane, causes the system to enter a final state.

Definition 4 Let us extend δ to $\bar{\delta}$, a function mapping the sequences of multisets over V and the configurations $([u_1], \ldots, [u_n])$ of Γ to new configurations. We define $\bar{\delta}$ as

1. $\bar{\delta}([v], ([u_1], \ldots, [u_n])) = \delta([v], ([u_1], \ldots, [u_n]))$ $[v], [u_i] \in V^\circ$, $1 \leq i \leq n$, and

2. $\bar{\delta}([v_1]\dots[v_s],([u_1],\dots,[u_n])) = \delta([v_s],\bar{\delta}([v_1]\dots[v_{s-1}],([u_1],\dots,[u_n]))$, $[v_j],[u_i] \in V^\circ,\ 1 \le i \le n,\ 1 \le j \le s$.

Definition 5 Let Γ be a one-way P automaton as above. The *language accepted by* Γ is the set of multiset sequences

$$\begin{aligned} L_{acc}(\Gamma) \quad = \quad & \{[v_1]\dots[v_s] \mid \bar{\delta}([v_1]\dots[v_s],([w_1],\dots,[w_n])) = ([u_1],\dots,[u_n]) \\ & \text{with } [u_j] \in F_j \text{ for all } j \text{ with } F_j \neq \emptyset,\ 1 \le j \le n, s \ge 1\}. \end{aligned}$$

According to this definition, the P automaton is in a final configuration if all of its regions having a set of final states different from the empty set are in one of these final states.

There might be other ways of defining the accepting configuration. To require that only one region is in a final state, is an example of the numerous possibilities.

Example 1 Consider the following one-way P automaton.

$$\begin{aligned} \Gamma = (\{S_1,S_2,S_3,a,b,c\}, & [\ [\ [\ ]_3\ ]_2\ ]_1, \\ & ([S_1],P_1,\{[\varepsilon]\}),([S_2],P_2,\{[S_1S_2]\}),([S_3],P_3,\emptyset)), \end{aligned}$$

with

$$\begin{aligned} P_1 \quad = \quad & \{S_1 : a \to in, a : a \to in, a : b \to in, b : b \to in, b : c \to in, \\ & c : c \to in, c : \emptyset \to in, \emptyset : \emptyset \to in\}, \end{aligned}$$

$$P_2 \quad = \quad \{S_2 : S_1 \to in, S_1 : a \to in, S_1 : b \to in, S_1 : c \to in, c : \emptyset \to in\},$$

$$P_3 \quad = \quad \{S_3 : \emptyset \to in, S_3 : abc \to in\},$$

This P automaton accepts multiset sequences of the form $[a]^n[b]^n[c]^n,\ n \ge 1$. The work of the machine starts with the initial configuration $([S_1],[S_2],[S_3])$, and after the first transition it is found in $([a],[S_2S_1],[S_3])$. Now by applying the second rules of P_1, P_2 and the first rule of P_3 the automaton reads a sequence of $[a]$s until it reaches the configuration $([a],[S_2S_1a^{n-1}],[S_3])$ and then $([b],[S_2S_1a^n],[S_3])$. Now by applying the fourth rule of P_1 and the third rule of P_2 it reads a sequence of $[b]$s, reaching $([b],[S_2S_1a^nb^{m-1}],[S_3])$, and then $([c],[S_2S_1a^nb^m],[S_3])$. Now a sequence of $[c]$s is read in a similar manner and the automaton reaches $([c],[S_2S_1a^nb^mc^{l-1}],[S_3])$ and then $([\varepsilon],[S_2S_1a^nb^mc^l],[S_3])$. Now by using the rule $S_3 : abc \to in$ of P_3 the automaton compares the number of letters read before, while also the last rule of P_1 and the last rule of P_2 is applied. If $n = m = l$ then the automaton reaches the final state $([\varepsilon],[S_1S_2],[S_3a^nb^mc^l])$.

3 Representation of RE in terms of one-way P automata

In this section we show that any language that can be recognized by a two-counter machine can also be obtained as an l-projection of the language accepted by some one-way P automaton. Moreover, the P automaton can be chosen to have seven membranes only. Since every recursively enumerable language is the accepted language of a two-counter machine, the statement gives a representation of the recursively enumerable language class in terms of P automata.

First we recall the notion of a two-counter machine; for further details the reader is referred to [1, 3].

A two-counter machine is a 3-tape Turing machine, $M = (\Sigma \cup \{Z, B\}, Q, R)$ where Σ is an alphabet – the alphabet of input symbols or input tape symbols –, Q is a set of states with two distinguished elements, $q_0, q_f \in Q$, and R is a set of transition rules. The state q_0 is called the initial state and q_f is called the final (accepting) state of M. The machine has a read only input tape and two semi-infinite storage tapes (the counters). The alphabet of the storage tapes consists of two symbols, Z and B (blank), while the alphabet of the input tape is $\Sigma \cup \{B\}$.

R consists of transition rules of the form $x = \langle b, q, c_1, c_2, q', e_1, e_2, g \rangle$, where $b \in \Sigma \cup \{B\}$ is the symbol scanned on the input tape in state $q \in Q$ and $c_1, c_2 \in \{Z, B\}$ are the symbols scanned on the storage tapes. M enters into state $q' \in Q$, the counters should be modified according to $e_1, e_2 \in \{-1, 0, +1\}$, that is, a symbol B is added to the contents of the counter $(+1)$, the contents of the counter remains unchanged (0), or one symbol B is removed from the counter (-1). (The contents of the counter is the number of blank symbols on the storage tape between the first position of the tape, carrying a Z, and the position under the reading head.) The input head moves according to $g \in \{0, +1\}$. If $g = +1$, then the head moves one cell to the right, if $g = 0$, then the head remains in the same position.

Symbol Z appears on the cells initially scanned by the storage tape heads and it never appears on any other cell. An integer i can be stored by moving a storage tape head i cells to the right of Z (the tape contains ZB^i). A stored number can be incremented or decremented by moving the tape head right or left. The machine is capable of checking whether a stored value is zero or not, by looking at the symbol scanned by the storage tape heads. If the scanned symbol is Z, then the value stored in the corresponding counter is zero.

A word $w \in \Sigma^*$ is accepted by the two counter machine if the input head scanned the last non-blank symbol on the input tape and the machine is in the final (accepting) state. That is, starting from the initial configuration - having an input word on the input tape, being in the initial state, and reading Zs on both of the counter tapes - the two-counter machine enters an accepting configuration, that is, the input head scanned the last non-blank symbol and the machine is in the accepting state.

Two counter machines are computationally complete, that is, they are just as powerful as the Turing machines [1, 3].

Now we present our theorem.

Theorem 1 *Any recursively enumerable language $L \subseteq \Sigma^*$ can be obtained as an l-projection of the language $L_{acc}(\Gamma)$ of some one-way P automaton Γ with seven membranes.*

Proof. Let $L \in \Sigma^*$ be a recursively enumerable language accepted by a two-counter machine $M = (\Sigma \cup \{Z, B\}, Q, R)$. We first construct a one-way P automaton $\Gamma = (V, \mu, (w_1, P_1, F_1), \ldots, (w_7, P_7, F_7))$ such that the accepted input sequences of Γ describe transition sequences of M which starting from an initial configuration, lead to an accepting configuration. The idea of the construction of Γ is based on the following considerations.

A transition $x = \langle b, q, c_1, c_2, q', e_1, e_2, g \rangle$ of M, where $b \in \Sigma$, $q, q' \in Q$, $c_1, c_2 \in \{B, Z\}$, $e_1, e_2 \in \{+1, 0, -1\}$, $g \in \{0, +1\}$, can be performed by M if the input symbol to be scanned by the input tape head is b, the machine is in state q, the symbols scanned on the counter tapes are c_1 and c_2. If these hold, then the contents of the counters are modified according to e_1 and e_2 and the input tape head moves according to g.

To describe and simulate these activities, dedicated membranes and regions of Γ are constructed. The region of the first membrane, that of the skin membrane of Γ is responsible for simulating the scanning of the actual input symbol, in this case symbol b, the second and fourth, respectively the third and fifth regions simulate the checking procedure to establish whether or not the symbol scanned on the counter tape corresponds to c_1, respectively c_2, and they also simulate the change in the contents of the first and second counter according to e_1 and e_2, respectively, all prescribed by the simulated transition. According to the construction of the P automaton, when starting the simulation of transition x of M, then the second, respectively the third region of Γ contains as many dedicated symbols A as many blank symbols can be found in the corresponding counter of M at that moment of the computation. The sixth and the seventh regions are for collecting

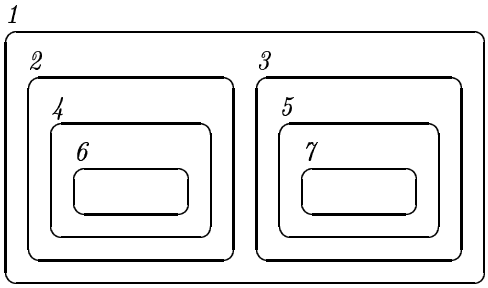

Figure 1: The membrane structure of Γ.

symbols that are unnecessary for the further computation steps in Γ. The change of state q to q' is simulated with the help of special symbols that maintain the synchronized actions of the membranes under the simulation of the execution of transition x in Γ. These symbols are also for simulating parameter g describing the position of the input tape head.

The membrane structure, μ, is as follows: the parent of the sixth (seventh) membrane is the fourth (fifth) membrane, the parent of the fourth (fifth) membrane is the second (third) membrane, and the skin membrane is the parent of the second (third) membrane (see Figure 1). Moreover, Γ is with one-way communication.

To help the reader in understanding how the automaton is designed, we explain the simulation of the execution of $x = \langle b, q, c_1, c_2, q', e_1, e_2, g \rangle$, a transition of M, with components defined above.

At the beginning of the simulation, the contents of the region of the skin membrane are occurrences of symbols x and $\bar{x}$, which indicate that the simulation of transition x will follow. The second, respectively the third region contains symbols referring to the simulated transition - either x' and F_x or x', depending whether the prescribed counter's contents is empty or not - and as many occurrences of symbol A as many blank symbols are in the corresponding counter of M at that moment of the computation in M. The fourth, respectively the fifth region contains symbols identifying the simulated transition (x'') and several As. Finally, the sixth and the seventh region, both, contain one occurrence of symbol D and they contain possibly several occurrences of symbols which either refer to a transition or they are input symbols of M.

Then Γ performs a computation step, where the symbols referring to transition x move from the corresponding region to its child region via communication and an input multiset of Γ is consumed from outside by the

skin membrane. But, the successful transition of Γ is possible only, if the input multiset consists of occurrences of the input symbol b of the simulated transition, x.

At the next step, the symbols referring to transition x and the input symbol b of x move one region down and the skin membrane consumes symbols from outside which identify the next transition of M to be applied (y''s) and as many occurrences of As as it is necessary to perform the modification of the contents of the counters of M according to parameters e_1 and e_2 under transition x. Moreover, this is the step where the check whether or not the contents of the counters correspond to c_1 and c_2 is executed. If c_i, $i = 1, 2$, prescribes a nonempty counter, then a symbol A has to move from the second, respectively from the third region to the fourth, respectively to the fifth region. If c_i, $i = 1, 2$ prescribes an empty counter, then the move of the input symbol b and that of symbol $\overline{x}$ from the regiion of the skin membrane to the second, respectively to the third region , is possible only if the region is empty. These communication steps can only be performed if the prescribed conditions in transition x hold in the corresponding computation step of M.

After the successful transition in Γ, at the next step, the skin membrane consumes further symbols from outside which identify the next transition of M to be performed (y's) and the second and the third regions are communicated with as many occurrences of As as it is necessary to perform the modification of the contents of the counters according to e_1 and e_2. Furthermore, symbols b and $\overline{x}$ from the second, respectively from the third region move to the fourth, respectively to the fifth region. The symbols referring to transition x move from the fourth, respectively from the fifth region move to the sixth, respectively to the seventh region. This step can be performed if the previous steps had been successfully executed.

Finally, at the last step, symbols y and $\bar{y}$ are consumed from outside by the skin membrane, to indicate that a new transition, y, will be simulated, and symbols y' and y'' move to the second, respectively, to the third region and to the fourth, respectively to the fifth region, symbols b and $\overline{x}$ move to the sixth and the seventh region.

The obtained configuration of Γ is of the same form that it was at the beginning of the simulation of transition x. The move of the reading head of the input tape (g) is taken into account at the initialization of the simulation of the following transition.

Obviously, in addition to the rules associated with the transitions of M, the regions of Γ contain other rules to guarantee the correct simulation. After this explanation, we define the P automaton Γ in details.

We first note that instead of simulating the functioning of M, we construct Γ simulating the functioning of the two-counter machine M' which accepts $\$_1 L \$_2$, where $\$_1, \$_2$ are two new symbols not in Σ.

M' has the states of M and it has some further new states. It is with initial state q_s, with final state q_{fin}, both different from the initial state and the final state of M. Moreover, in addition to q_f M has a new state q'_f. The transition set of M' consists of the transitions of M and some further transitions given as follows: $\langle \$_1, q_s, Z, Z, q_0, 0, 0, 1\rangle$, $\langle \$_2, q'_f, c_1, c_2, q_{fin}, 0, 0, 0\rangle$, where $c_1, c_2 \in \{B, Z\}$, and for each transition $\langle b, q, c_1, c_2, q_f, e_1, e_2, g\rangle$ in M there is a transition in M' of the form $\langle b, q, c_1, c_2, q'_f, e_1, e_2, 1\rangle$.

Thus, M' first reads symbol $\$_1$ and then continues reading word $w \in \Sigma^*$ in the same way as M, and it enters the accepting state if M enters the accepting state.

Now we define the components of Γ. First, let us denote $\Sigma_T = \{x \mid x = \langle b, q, c_1, c_2, q', e_1, e_2, g\rangle$ is a transition of $M\}$. Before continuing, we have an observation.

We say that transition $y = \langle a, q_y, c_{1,y}, c_{2,y}, q'_y, e_{1,y}, e_{2,y}, g_y\rangle$ is applicable after transition $x = \langle b, q_x, c_{1,x}, c_{2,x}, q'_x, e_{1,x}, e_{2,x}, g_x\rangle$ in M' if $q'_x = q_y$ holds. Moreover, if $g_x = 0$, then $a = b_e$.

For technical reasons, when simulating the transitions of M' with Γ, we have to indicate whether or not the reading input head of M' was at the same position at the previous transition. For this purpose we define a new alphabet $\Sigma^e = \{b_e \mid b \in \Sigma\}$.

Let $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x\rangle$ be a transition in M such that $g_x = 0$, that is, the reading head at the input tape does not move after performing the transition. Suppose that M has a transition $y = \langle b, q', c_{1,y}, c_{2,y}, q'', e_{1,y}, e_{2,y}, g_y\rangle$ applicable after x. Then, for each transition y with this property we add a new transition $y_e = \langle b_e, q', c_{1,y}, c_{2,y}, q'', e_{1,y}, e_{2,y}, g_y\rangle$ to the set of transitions of M'. Let us denote by Σ^e_T the set of these new transitions.

Now we define the alphabet of Γ as $V = \Sigma \cup \Sigma_T \cup \Sigma'_T \cup \Sigma''_T \cup \overline{\Sigma}_T \cup \Sigma^e \cup \Sigma^e_T \cup \Sigma^{e\,\prime}_T \cup \Sigma^{e\,\prime\prime}_T \cup \overline{\Sigma^e_T} \cup \{F_x \mid x \in \{\Sigma_T \cup \Sigma^e_T\} \cup \{D, B, A, Z, \$_1, \$_2\}$, where $\overline{\Sigma}_T$, Σ'_T, $\Sigma_T{}''$ $\overline{\Sigma^e_T}$, $\Sigma^{e\,\prime}_T$, $\Sigma^{e\,\prime\prime}_T$ are the over-lined, primed and double-primed versions of the corresponding alphabets.

Let Γ have the following membrane structure $\mu = [\,[\,[\,[\,]_6\,]_4\,]_2\,[\,[\,[\,]_7\,]_5\,]_3\,]_1$, as indicated in Figure 1, and let us define the initial states of the corresponding regions as follows: $w_1 = \overline{x_s x_s}$, $w_2 = w_3 = x_s{}' F_{x_s}$, $w_4 = w_5 = x_s{}''$, where $x_s = \langle \$_1, q_s, Z, Z, q_0, 0, 0, 1\rangle$, and let $w_6 = w_7 = D$.

We define the final states of the regions as follows. F_1 consists of the following multisets: $\$_2 \$_2 \overline{yy}$ for $y = \langle \$_2, q'_f, c_1, c_2, q_{fin}, 0, 0, 0\rangle$ with $c_1, c_2 \in$

$\{B, Z\}$, and let $F_i = \emptyset$ for $i = 2, 3, 4, 5, 6, 7$.

Thus, Γ starts its work by consuming an input multiset with symbols $\$_1$ from outside and stops after consuming an input multiset with symbols $\$_2$ by the skin membrane from outside.

Now we define the rule sets of the different membranes.

P_1 consists of all productions of the following forms:

- $xx\overline{xx} : bb \to in$ for $x = \langle b, q, c_{1,x}c_{2,x}, q', e_{1,x}, e_{2,x}, g_x\rangle \in \Sigma_T \cup \Sigma_T^e, c_{1,x} = c_{2,x} = B$, and
- $x\overline{xx} : bb \to in$ for $x = \langle b, q, c_{1,x}c_{2,x}, q', e_{1,x}, e_{2,x}, g_x\rangle \in \Sigma_T \cup \Sigma_T^e, Z \in \{c_{1,x}, c_{2,x}\}$, and $c_{1,x} \neq c_{2,x}$,
- $\overline{xx} : bb \to in$ for $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x\rangle \in \Sigma_T \cup \Sigma_T^e, c_{1,x} = c_{2,x} = Z$.

These rules are for checking whether the input symbols from Σ consumed from outside by the skin membrane of Γ correspond to the input symbol of M' prescribed by its corresponding transitions.

- $bb\overline{xx} : y''y''wz \to in$ for $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x\rangle \in \Sigma_T \cup \Sigma_T^e$, where $y = \langle a, q', c_{1,y}, c_{2,y}, q'', e_{1,y}, e_{2,y}, g_y\rangle \in \Sigma_T$ for $g_x = 1$ and $y \in \Sigma_T^e$ for $g_x = 0$, and $a = b_e$ if $g = 0$ and $b \in \Sigma$ and $a = b$ if $g = 0$ and $b \in \Sigma^e$. Furthermore, $w = AA$ for $e_{1,x} = 1$, $w = A$ for $e_{1,x} = 0$, $w = \varepsilon$ for $e_{1,x} = -1$ and $c_{1,x} \neq Z$ and $w = A$ for $e_{1,x} = 1$ and $w = \varepsilon$ for $e_{1,x} = 0$, and $c_{1,x} = Z$, and $z = AA$ for $e_{2,x} = 1$, $w = A$ for $e_{2,x} = 0$, $w = \varepsilon$ for $e_{2,x} = -1$ and $c_{2,x} \neq Z$ and $z = A$ for $e_{2,x} = 1$ and $w = \varepsilon$ for $e_{2,x} = 0$, and $c_{2,x} = Z$,
- $y''y'' : y'y' \to in$ for any $y = \langle b, q, c_{1,y}, c_{2,y}, q', e_{1,y}, e_{2,y}, g_y\rangle \in \Sigma_T \cup \Sigma_T^e$, where $Z \notin \{c_{1,y}, c_{2,y}\}$, and
- $y''y'' : y'y'F_y \to in$ for any $y = \langle b, q, c_{1,y}, c_{2,y}, q', e_{1,y}, e_{2,y}, g_y\rangle \in \Sigma_T \cup \Sigma_T^e$, with $Z \in \{c_{1,y}, c_{2,y}\}$, and $c_{1,y} \neq c_{2,y}$, and
- $y''y'' : y'y'F_yF_y \to in$ for any $y = \langle b, q, c_{1,y}, c_{2,y}, q', e_{1,y}, e_{2,y}, g_y\rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{1,y} = c_{2,y} = Z$, and fin ally,
- $y'y' : yy\overline{yy} \to in$ for any $y = \langle b, q, c_{1,y}, c_{2,y}, q', e_{1,y}, e_{2,y}, g_y\rangle \in \Sigma_T \cup \Sigma_T^e$, with $Z \notin \{c_{1,y}, c_{2,y}\}$, and
- $y'y' : y\overline{yy} \to in$ for any $y = \langle b, q, c_{1,y}, c_{2,y}, q', e_{1,y}, e_{2,y}, g_y\rangle \in \Sigma_T \cup \Sigma_T^e$, with $Z \in \{c_{1,y}, c_{2,y}\}$, and $c_{1,y} \neq c_{2,y}$, and

- $y' : \overline{yy} \to in$ for any $y = \langle b, q, c_{1,y}, c_{2,y}, q', e_{1,y}, e_{2,y}, g_y \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{1,y} = c_{2,y} = Z$.

These rules are for initializing the simulation of the next transition to be performed after a transition of M'.

Rule sets P_2 and P_3 are defined in the same way. Let $P_i, i = 2, 3$, consist of the following rules:

- $x' : x \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{i-1,x} \neq Z$ for $i = 2, 3$, and
- $x'F_x : \emptyset \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{i-1,x} = Z$ for $i = 2, 3$.

These rules assist to synchronizing the activity of the regions while simulating a transition in M'.

- $x : b\overline{x} \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{i-1,x} \neq Z$ for $i = 2, 3$, and
- $\emptyset : b\overline{x} \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{i-1,x} = Z$ for $i = 2, 3$.

These rules assist in checking whether or not the actual contents of the counters of M' represented in the second, respectively in the third region in an appropriate manner.

- $b : y''w \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$, where $y = \langle a, q', c_{1,y}, c_{2,y}, q'', e_{1,y}, e_{2,y}, g_y \rangle \in \Sigma_T$ for $g_x = 1$, and $y \in \Sigma_T^e$ for $g_x = 0$, and $a = b_e$ if $g = 0$ and $b \in \Sigma$ and $a = b$ if $g = 0$ and $b \in \Sigma^e$. Furthermore, $w = AA$ if $e_{i-1,x} = 1$, $w = A$ if $e_{i-1,x} = 0$, and $w = \varepsilon$ if $e_{i-1,x} = -1$ and $c_{i-1,x} \neq Z$, and $w = A$ if $e_{i-1,x} = 1$, $w = \varepsilon$ if $e_{i-1,x} = 0$, and $c_{i-1,x} = Z$.
- $y'' : y' \to in$ for any $y = \langle b, q, c_{1,y}, c_{2,y}, q', e_{1,y}, e_{2,y}, g_y \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{i-1,y} \neq Z$ for $i = 2, 3$, and
- $y'' : y'F_y \to in$ for any $y = \langle b, q, c_{1,y}, c_{2,y}, q', e_{1,y}, e_{2,y}, g_y \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{i-1,y} = Z$ for $i = 2, 3$.

These rules assist in initializing the simulation of the next transition to be performed after a transition in M'.

Rule sets P_4 and P_5 are defined in the same way. Let $P_i, i = 4, 5$, consist of the following rules:

- $x'' : x' \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{i-3,x} \neq Z$ for $i = 4, 5$.
- $x'' : x'F_x \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{i-3,x} = Z$ for $i = 4, 5$.

These rules contribute to maintaining the synchronized activity of the regions when simulating a certain transition in M'.

- $x' : Ax \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{i-3,x} \neq Z$ for $i = 4, 5$, and
- $x'F_x : \emptyset \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{i-3,x} = Z$ for $i = 4, 5$.

These rules take part in checking whether or not the actual contents of the counters of M' are represented in the second, respectively in the third region in an appropriate manner.

- $x : b\overline{x} \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{i-3,x} \neq Z$ for $i = 4, 5$, and
- $F_x : b\overline{x} \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{i-3,x} = Z$ for $i = 4, 5$, and
- $b : y'' \to in$ for $y = \langle b, q, c_{1,y}, c_{2,y}, q', e_{1,y}, e_{2,y}, g_y \rangle \in \Sigma_T \cup \Sigma_T^e$.

These rules help in maintaining the synchronized activity of the regions under simulating a certain activity and also take part in initializing the simulation of the next transition to be performed after a transition in M'.

Rule sets P_6 and P_7 are defined in the same way. Let $P_i, i = 6, 7$, consist of the following rules:

- $D : x'' \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$,
- $D : x' \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$,
- $D : x \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{i-5,x} \neq Z$, and
- $D : F_x \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$, with $c_{i-5,x} = Z$, and
- $D : b\overline{x} \to in$ for any $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x \rangle \in \Sigma_T \cup \Sigma_T^e$.

These rules assist in maintaining the synchronized activity of the regions under simulating a certain transition of M' by collecting symbols unnecessary for further computation steps.

Now we prove that transition sequences of Γ correspond to transition sequences of M'.

Suppose that at some stage of the computation, Γ is in configuration

$$(t, t'_{1,x}\alpha', t'_{2,x}\beta', x''\gamma, x''\delta, Du, Dv),$$

where $t = xx\overline{xx}$ if $Z \notin \{c_{1,x}, c_{2,x}\}$, $t = x\overline{xx}$ if $Z \in \{c_{1,x}, c_{2,x}\}$ and $c_{1,x} \neq c_{2,x}$, and $t = \overline{xx}$ if $c_{1,x} = c_{2,x} = Z$, for some $x = \langle b, q, c_{1,x}, c_{2,x}, q', e_{1,x}, e_{2,x}, g_x\rangle \in \Sigma_T \cup \Sigma_T^e$. Furthermore, $t'_{i,x} = x'$, $\alpha' = \alpha \in A^*$ if $c_{1,x} \neq Z$, $\alpha' = F_x\alpha$ with $\alpha \in A^*$ for $c_{1,x} = Z$, $\beta' = \beta \in A^*$ if $c_{2,x} \neq Z$, $\beta' = F_x\beta$ with $\beta \in A^*$ for $c_{2,x} = Z$, $\gamma \in A^*, \delta \in A^*, u, v \in (\Sigma \cup \Sigma^e \cup \Sigma_T \cup \Sigma_T^e \cup \Sigma'_T \cup \Sigma''_T \cup \overline{\Sigma}_T \cup \Sigma_T^e)^*$.

Moreover, α' and β', respectively α and β, contain as many occurrences of A as many blank symbols can be found in the first, respectively in the second, counter of M' at the beginning of the simulation of the next transition, x.

Thus, the contents of the first region refers to the transition x of M' which will be simulated, and this plan is also indicated by the second, the third, the fourth and the fifth region. As we have said before, the sixth and the seventh regions collect the symbols which are unnecessary for the further steps of the computation.

In this configuration the only 7-tuple of rules that can be applied is the following.

$$(t : bb \to in, r'_1 : s_1 \to in, r'_2 : s_2 \to in, x'' : r'_1 \to in, \\ x'' : r'_2 \to in, D : x'' \to in, D : x'' \to in),$$

where t and b are defined above and for $i = 1, 2$, $r'_i = x'$ for $c_{i,x} \neq Z$ and $r'_i = x'F_x$ for $c_{i,x} = Z$, $s_i = x$ for $c_{i,x} \neq Z$ and $s_i = \emptyset$ for $c_{i,x} = Z$.

By the application of these rules, the skin membrane checks whether the input multiset identifies the input symbol b in transition x of M' and makes preparations for checking whether or not the contents of the second and the third region correspond to the required contents of the corresponding counters of M' to successfully perform transition x. If rule $t : bb \to in$ cannot be applied, then the computation aborts. After successfully applying the above 7-tuple of rules, Γ enters configuration

$$(bb\overline{xx}, \alpha'', \beta'', \gamma', \delta', x''Du, x''Dv),$$

where $\alpha'' = x\alpha$ for $c_{1,x} \neq Z$, and $\alpha'' = \alpha$ for $c_{1,x} = Z$, $\beta'' = x\beta$ for $c_{2,x} \neq Z$, $\beta'' = \beta$ for $c_{2,x} = Z$, $\gamma' = x'\gamma$ for $c_{1,x} \neq Z$, and $\gamma' = x'F_x\gamma$ for $c_{1,x} = Z$, and $\delta' = x'\delta$ for $c_{2,x} \neq Z$, and $\delta' = x'F_x\delta$ for $c_{2,x} = Z$.

Thus, the second, respectively the third region contains as many occurrences of A as the number stored in the first, respectively in the second counter of M' at this step of the computation. Moreover, if a counter is not empty, and only in this case, the corresponding region contains an occurrence of x.

Now the only 7 tuple of rules which can be applied is as follows.

$$(bb\overline{x}\overline{x} : y''y''wz \to in, r_1'' : b\overline{x} \to in, r_2'' : b\overline{x} \to in, r_3'' : s_3 \to in, \\ r_4'' : s_4 \to in, D : x' \to in, D : x' \to in),$$

where $r_i'' = x$ for $c_{i,x} \neq Z$ and $r_i'' = \emptyset$ for $c_{i,x} = Z$, $i = 1, 2$, and $r_j'' = x'$ for $c_{j-2,x} \neq Z$ and $r_j'' = x'F_x$ for $c_{j-2,x} = Z$, and $s_j = Ax$ for $c_{j-2,x} \neq Z$ and $s_j = \emptyset$ for $c_{j-2,x} = Z$, $j = 3, 4$. Moreover, we know that for $y = \langle a, q', c_{1,y}, c_{2,y}, q'', e_{1,y}, e_{2,y}, g_y \rangle$, $a = b_e$ holds if $g_x = 0$.

These rules simulate the checking of the applicability of transition x of M', and initialize the simulation of the next transition of M'. The initialization is done by applying rule $bb\overline{x}\overline{x} : y''y''wz \to in$ in the first region.

The checking of the contents of the counters of M' is simulated by applying rules of the second, third, fourth and fifth regions as follows. If the first, respectively the second counter of M' must be empty according to the transition, then the second region, respectively the third region must be empty. This is the only case when rule $\emptyset : b\overline{x} \to in$ can be successfully applied. If the region is not empty, then it contains several As without any other symbol, so no rule in P_2 (in P_3) can be applied, thus, the computation aborts.

If the first, respectively the second counter must not be empty, then the second, respectively the third region must contain symbol x and at least one occurrence of A, and this is the only case when the rule $x' : Ax \to in$ can be successfully applied in the fourth, respectively in the fifth region. Otherwise, the computation of Γ ends with abnormal termination.

After applying this 7-tuple of rules, the obtained configuration of Γ is as follows:

$$(y''y''wz, b\overline{x}\alpha''', b\overline{x}\beta''', \gamma'', \delta'', x'x''Du, x'x''Dv),$$

where $\alpha''' \in A^*, |\alpha'''| = |\alpha| - 1$ for $c_{1,x} \neq Z$, and $\alpha''' = \varepsilon$ for $c_{1,x} = Z$, $\beta''' \in A^*, |\beta'''| = |\beta| - 1$ for $c_{2,x} \neq Z$, $\beta''' = \varepsilon$ for $c_{2,x} = Z$, $\gamma'' = xA\gamma'$ for $c_{1,x} \neq Z$, and $\gamma'' = F_x\gamma'$ for $c_{1,x} = Z$, and $\delta'' = xA\delta'$ for $c_{2,x} \neq Z$, and $\delta'' = F_x\delta'$ for $c_{2,x} = Z$.

Thus, the first region contains two occurrences of the symbol y'' which identify the next transition of M' to be simulated. Moreover, the region of the skin membrane contains occurrences of A calculated as follows. If the respective counter had to be nonempty before performing the transition, then this region contains one occurrence of A more than the number of blank symbols prescribed by the transition to modify the actual contents of the counter, if the counter had to be empty, the the number of occurrences of A is exactly the same as it is prescribed by the transition. This is calculated for both counters of M'.

Then, the next 7 tuple of rules which can be applied is the following.

$$(y''y'' : t'_{1,y}t'_{2,y} \to in, b : y''w \to in, b : y''z \to in, r'''_3 : b\overline{x} \to in, \\ r'''_4 : b\overline{x} \to in, D : r'''_3 \to in, D : r'''_4 \to in),$$

where $t'_{i,y} = y'$ for $c_{i,y} \neq Z$ and $t'_{i,y} = y'F_y$ for $c_{i,y} = Z$ for $i = 1, 2$, $r'''_3 = x$ for $c_{1,x} \neq Z$ and $r'''_3 = F_x$ for $c_{1,x} = Z$, and $r'''_4 = x$ for $c_{2,x} \neq Z$ and $r'''_4 = F_x$ for $c_{2,x} = Z$.

Then the contents of the second region and that of the third region are modified to correspond to the contents of the respective counters of M' after the successful application of transition x and the initialization of the simulation of the next transition continues.

The new configuration of Γ is as follows.

$$(t'_{1,y}t'_{2,y}, y''\alpha^{iv}, y''\beta^{iv}, b\overline{x}\gamma^{iv}, b\overline{x}\delta^{iv}, r_6x'x''Du, r_7x'x''Dv),$$

where $\alpha^{iv} \in A^*$ with as many occurrences of A as the value stored in the first counter of M' after performing transition x, $\beta^{iv} \in A^*$, with as many occurrences of A as the value stored in the second counter of M' after performing transition x, γ^{iv} and δ^{iv} consist of several occurrences of As, $r_6 = x$ for $c_{1,x} \neq Z$ and $r_6 = F_x$ for $c_{1,x} = Z$, $r_7 = x$ for $c_{2,x} \neq Z$ and $r_7 = F_x$ for $c_{2,x} = Z$.

The simulation of transition x and the initialization of the simulation of transition y is finished by applying the 7-tuple of rules

$$(t'_{1,y}t'_{2,y} : t_y \to in, y'' : t'_{1,y} \to in, y'' : t_{2,y} \to in, \\ b : y'' \to in, b : y'' \to in, D : b\overline{x} \to in, D : b\overline{x} \to in),$$

where $t'_{1,y}$ and $t'_{2,y}$ are defined as above, and $t_y = yy\overline{yy}$ for $Z \notin \{c_{1,y}, c_{2,y}\}$, $t_y = y\overline{yy}$ for $Z \in \{c_{1,y}, c_{2,y}\}$, $c_{1,y} \neq c_{2,y}$, and $t_y = \overline{yy}$ for $c_{1,y} = c_{2,y} = Z$. Moreover, this is the only 7-tuple of rules that can be applied at this step of the computation.

As a result, Γ enters configuration

$$(t_y, t'_{1,y}\alpha^{iv}, t'_{2,y}\beta^{iv}, y''\gamma^v, y''\delta^v, br_6x'x''Du, br_7x'x''Dv),$$

Which is of the same form as the configuration we have started from. It is easy to see that Γ simulates the functioning of M', and only that. Thus, for any accepting computation in M' there is an accepting computation in Γ and reversely. If we define l-projection h with $h([u]) = V \cap \Sigma$ for $[u] = (V, f) \in V^\circ$, then we can easily see that for any accepted input multiset sequence $[a_1] \dots [a_n]$ of Γ it holds that $h([a_1] \dots [a_n]) = w$, where $w \in L$. Hence the result. □

4 Final remarks

The aim of our paper was to introduce the notion of a P automaton, a P system defined as an accepting device using communication rules only, according to two of the problems raised by Gheorghe Păun in [7] and in [8]. Obviously, similarly to the large number of different types of automata, a wide variety of P automata can be defined and studied, with different types of communication or accepting. It is also an interesting problem area how to define complexity measures for these devices. We shall return to these topics in future papers.

References

[1] P.C. Fischer: Turing machines with restricted memory access. Information and Control 9(1966), 364-379.

[2] Handbook of Formal Languages. (G. Rozenberg, A. Salomaa, eds.), Volumes I-III. Springer, 1997.

[3] J.E. Hopcroft, J.D. Ullman: Introduction to Automata Theory, Languages and Computation. Addison-Wesley, Reading, Mass., 1979.

[4] Gh. Păun: Computing with Membranes. Turku Centre for Computer Science - TUCS Report No 208, 1998.

[5] Gh. Păun: Computing with Membranes. An Introduction. Bulletin of the EATCS 67 (1999), 139-152.

[6] Gh. Păun: Computing with Membranes. Journal of Computer and System Sciences 61(1) (2000), 108-143.

[7] Gh. Păun: Computing with membranes (P Systems): Twenty Six Research Topics. CDMTCS Technical Report 119, Univ. of Auckland, 2000, 203-217.

[8] Gh. Păun: Membrane Computing. An Introduction. Manuscript, 429 pages, to appear. Springer Verlag, Berlin-Heidelberg, 2002.

[9] Gh. Păun, G. Rozenberg: A Guide to Membrane Computing. Theoretical Computer Science, to appear.

[10] P systems web page at `http://dna.bio.disco.unimib.it/psystems`