P Automata with Controlled Use of Minimal Communication Rules

Rudolf Freund\textsuperscript{1}, Marian Kogler\textsuperscript{1}, and Sergey Verlan\textsuperscript{2}

\textsuperscript{1} Faculty of Informatics, Vienna University of Technology
Favoritenstr. 9, 1040 Vienna, Austria

\textsuperscript{2} LACL, Département Informatique
UFR Sciences et Technologie, Université Paris XII
61, av. Général de Gaulle, 94010 Créteil, France

Email: rudi@emcc.at, marian@emcc.at
and verlan@univ-paris12.fr

NCMA 2009
Overview

Introduction to Membrane Systems
   (Generating) P Systems
   Analyzing P Systems / P Automata

Register Machines

Communication P Automata
   Communication Rules
   Transition Modes

Communication P Automaton with Rule Control

Computational Completeness
   Maximal Parallelism
   (1-Restricted) Minimal Parallelism

Conclusion
Introduction to Membrane Systems

**P Systems**

- introduced by Gheorghe Păun in 1998,
- inspired by cell functioning,
- multisets of objects evolve in parallel,
- in a hierarchical membrane structure.

**References**


Analyzing P Systems / P Automata

introduced 2002:

▶ Analyzing P Systems: R. Freund and M. Oswald;
▶ P Automata: E. Csuhaj-Varjú and Gy. Vaszil;
▶ act as acceptors rather than as generators;
▶ multiset is accepted if and only if the system/automaton halts.


Register Machines

A *deterministic register machine* is a construct $M = (n, B, l_0, l_h, I)$, where

- $n$ is the number of registers,
- $B$ is a set of instruction labels,
- $l_0$ is the start label,
- $l_h$ is the halt label (assigned to HALT only), and
- $I$ is a set of instructions of the following forms:
  - $l_i : (\text{ADD}(r), l_j)$ add 1 to register $r$, and then go to the instruction labeled by $l_j$;
  - $l_i : (\text{SUB}(r), l_j, l_k)$ if register $r$ is non-empty (non-zero), then subtract 1 from it and go to the instruction labeled by $l_j$, otherwise go to the instruction labeled by $l_k$;
  - $l_h : \text{HALT}$ the halt instruction.
Register Machines – Computations

A register machine $M$ accepts a set of (vectors of) natural numbers in the following way:

▶ start with the instruction labeled by $l_0$, with the first registers containing the input as well as all other registers being empty,
▶ apply instructions as indicated by the labels and by the contents of the registers,
▶ accept the input number (vector) if the HALT instruction is reached.

It is known that in this way we can accept all recursively enumerable sets of (vectors of) natural numbers.
A *communication P automaton* is a construct

\[ \Pi = (O, T, \mu, E, w_0, w_1, \ldots, w_d, i_0, R) \]

where

1. \( O \) is a finite alphabet of objects;
2. \( T \subseteq O \) is the alphabet of terminal objects;
3. \( \mu \) is a membrane structure of \( d \) membranes with labels \( i \), \( 1 \leq i \leq d \), the skin membrane always has the index 1; the environment is indicated by 0;
4. \( E \subseteq O \) is the alphabet of objects occurring infinitely often in the environment;
5. \( w_0, w_1, \ldots, w_d \) initial multisets of the environment (\( w_0 \) only contains objects from \( O - E \)) and membranes \( i, 1 \leq i \leq d \);
6. \( i_0, 1 \leq i_0 \leq d \), is the input membrane;
7. \( R \) is a set of communication rules.
**Communication Rules**

**Definition**

*Sympor*t rules in $R$ are of the form $x[i \rightarrow [i]x$ meaning that the multiset $x$ from outside membrane $i$ is moved into the region inside membrane $i$, or $[i]x \rightarrow x[i]$ meaning that the multiset $x$ from inside membrane $i$ is moved into the region surrounding membrane $i$, with $x \in O^+$ and $1 \leq i \leq d$.

For symport rules $x[1 \rightarrow [1]x$, at least one symbol from $x$ has to be from $O - E$.

**Definition**

*Antiport* rules in $R$ are of the form $x[;y \rightarrow y[;x$ with $x, y \in O^+$ and $1 \leq i \leq d$, meaning that the multiset $x$ from outside membrane $i$ is exchanged with the multiset $y$ in the region inside membrane $i$. 
Weight of Communication Rules

**Definition**

The **weight** of a symport rule $x[i \rightarrow [i \times]$ or $[i \times \rightarrow x[i]$ is defined as $|x|$, The **weight** of an antiport rule $x[i\ y \rightarrow y[i\ x]$ is defined as $\max(|x|, |y|)$.

**Definition**

If we consider symport rules of any weight, we write $\text{sym}_*$; if we only consider symport rules with weight $\leq n$, we write $\text{sym}_n$; $\text{sym}_2$ rules are also called **minimal symport rules**; finally, $\text{sym}_1$ rules are called **uniport rules**.

If we consider antiport rules of any weight, we write $\text{anti}_*$; if we only consider antiport rules with weights $\leq n$, we write $\text{anti}_n$; $\text{anti}_1$ rules are also called **minimal antiport rules**.
A configuration $C$ of $\Pi$ is a $(d + 1)$-tuple of multisets over $O$ $(u_0, u_1 \ldots, u_d)$; the initial configuration of $\Pi$, $C_0$, is described by $w_0, w_1, \ldots, w_d$, i.e., $C_0 = (w_0, w_1, \ldots, w_d)$. The set of all multisets of rules from $R$ applicable to $C$ is denoted by $\text{Appl}(\Pi, C)$.

To narrow the possible set of multisets of rules that can be applied to a given configuration, we may apply different transition modes. For the transition mode $\vartheta$, the selection of multisets of rules applicable to a configuration $C$ is denoted by $\text{Appl}(\Pi, C, \vartheta)$. 
Transition Modes

**Definition**

For the *maximally parallel* transition mode (*max*), we define

\[
Appl(\Pi, C, \text{max}) = \{ R' | R' \in Appl(\Pi, C) \text{ and there is no } R'' \in Appl(\Pi, C) \text{ with } R'' \nsubseteq R' \}\.
\]

Minimally Parallel Transition Mode

For the *minimally parallel* mode, we need an additional feature for the set of rules $R$, i.e., we consider a partitioning of $R$ into (not necessarily disjoint) subsets $R_1$ to $R_h$. Usually, this partitioning of $R$ may coincide with a specific assignment of the rules to the membranes.

In an informal way, it can be described as applying multisets such that from every set $R_j$, $1 \leq j \leq h$, at least one rule – if possible – has to be used:

**Definition**

For the *minimally parallel* transition mode (*min*), we define

$$\text{Appl}(\Pi, C, \text{min}) = \{ R' \mid R' \in \text{Appl}(\Pi, C) \text{ and there is no } R'' \in \text{Appl}(\Pi, C) \text{ with } R'' \supsetneq R', (R'' - R') \cap R_j \neq \emptyset \text{ and } R' \cap R_j = \emptyset \text{ for some } j, 1 \leq j \leq h \}.$$
**$k$-Restricted Minimally Parallel Transition Mode**

**Definition**

For the $k$-restricted minimally parallel transition mode ($\text{min}_k$), we define

$$\text{Appl} (\Pi, C, \text{min}_k) = \left\{ R' \mid R' \in \text{Appl} (\Pi, C, \text{min}) \text{ and } |R' \cap R_j| \leq k \text{ for all } j, 1 \leq j \leq h \right\}.$$

---


Definition

A *communication P automaton with rule control* is a construct

\[ \Pi' = (O, T, \mu, E, w_0, w_1, ..., w_d, i_0, R, R'_1, ..., R'_m, K) \]

where

1. \( \Pi = (O, T, \mu, E, w_0, w_1, ..., w_d, i_0, R) \) is a communication P automaton with \( O, T, \mu, E, w_0, w_1, ..., w_d, i_0, \) and \( R \) being defined as before;

2. \( R'_1, ..., R'_m \) is a partitioning of \( R \) into (non-empty, but not necessarily disjoint) subsets;

3. \( K \subseteq \{0, 1\}^m \) is a set of control vectors controlling the applicability of multisets of rules from \( R \).
Computations

Definition
For some given transition mode \( \vartheta \),

\[
\text{Appl} (\Pi', C, \vartheta) = \{ R' \mid R' \in \text{Appl} (\Pi, C, \vartheta) \text{ and there exists a vector } \nu \in K \text{ such that for all } j \text{ with } 1 \leq j \leq m \text{ it holds that } \\
\nu (j) = 1 \text{ implies } R' \cap R_j \neq \emptyset \text{ and } \\
\nu (j) = 0 \text{ implies } R' \cap R_j = \emptyset \}.
\]

A computation in \( \Pi' \) checking for the acceptance of a multiset \( w \) consists of a sequence of transitions starting from the initial configuration \( C'_0 = (w'_0, w'_1, \ldots, w'_d) \) with \( w'_i = w_i \) for \( 0 \leq i \leq m \) and \( i \neq i_0 \) and \( w'_i = w_i + w \) for \( i = i_0 \). A multiset \( w \) is accepted if and only if there exists a halting computation of \( \Pi' \) on \( C'_0 \).
Example

Let $\Pi' = (O, T, [1]_1, E, w_0, w_1, 1, R, R_1, R_2, R_3, R_4, R_5, K)$ be a communication $P$ automaton with rule control with

$$
\begin{align*}
O &= \{ a, b, p_1, p_2 \}, \\
T &= \{ a \}, \\
E &= \{ b \}, \\
w_0 &= \{ \}, \\
w_1 &= \{ p_1 p_2 \}, \\
R &= R_1 \cup R_2 \cup R_3 \cup R_4 \cup R_5; \\
R_1 &= \{ [1p_1a \rightarrow p_1a][1] \}, \\
R_2 &= \{ [1p_2a \rightarrow p_2a][1] \}, \\
R_3 &= \{ p_1b[1 \rightarrow [1p_1b] \}, \\
R_4 &= \{ p_2[1 \rightarrow [1p_2] \}, \\
R_5 &= \{ p_1[1p_2 \rightarrow p_2[1p_1, p_2[1p_1 \rightarrow p_1[1p_2] \}, \\
K &= \{ (1, 1, 0, 0, 0), (1, 0, 0, 0, 0), (0, 0, 1, 1, 0), (0, 0, 0, 0, 1) \}.
\end{align*}
$$
Example

In the maximally parallel mode, this automaton starts with the multiset $p_1 p_2 a^n$ for some $n \geq 0$ (where $a^n$ is the input) and first applies the rules in $R_1$ and $R_2$ in parallel, thereby fulfilling $(1, 1, 0, 0, 0)$, exporting two symbols $a$ from the skin membrane into the environment; if only one symbol $a$ is available, then only $[1p_1a \rightarrow p_1a]_1$ is applied thereby fulfilling $(1, 0, 0, 0, 0)$. If both $p_1$ and $p_2$ have been moved out, then the rules in $R_3$ and $R_4$ are executed at the same time according to $(0, 0, 1, 1, 0)$; these two steps have replaced two symbols $a$ by one symbol $b$. The P automaton $\Pi'$ repeats this process until at most one symbol $a$ is left in the skin membrane. If no $a$ is left, i.e., $n$ has been an even number, the automaton terminates with having $p_1 p_2 b^{n/2}$ in the skin membrane. If one $a$ is left, i.e., $n$ has been an odd number, then only $R_1$ is applicable and the automaton ends up in an infinite loop with $p_1[1p_2 \rightarrow p_2[1p_1]$ and $p_2[1p_1 \rightarrow p_1[1p_2]$. The automaton therefore exactly accepts $a^{2m}$ for any $m \geq 0$. 
Definition

For some given transition mode \( \vartheta \), by

\[ NO_l P_d K_m (\vartheta) [\text{rule types}] (PsO_l P_d K_m (\vartheta) [\text{rule types}] ) \]

we define the sets of (vectors of) natural numbers accepted by communication P automata with rule control working in the transition mode \( \vartheta \) in \( d \) membranes using \( l \) objects and a partitioning with \( m \) rule sets, allowing rules of the types specified in [rule types]; if any of the numbers \( d, m, l \) are unbounded, we write \( * \) instead.
Computational Completeness

By simulating deterministic register machines, we can show that communication $P$ automata with rule control using only minimal symport rules ($sym_2$ rules) or minimal antiport rules ($anti_1$ rules) together with uniport rules ($sym_1$ rules) in only one membrane are computationally complete.

**Theorem**

For $X \in \{N, Ps\}$,

\[
XRE = XO_\ast P_1 K_\ast (max) [sym_2] \\
XO_\ast P_1 K_\ast (max) [anti_1, sym_1].
\]
Computational Completeness – Proof Ideas

ds_\text{sym}(2):

\begin{itemize}
  \item \[ p_i : (\text{ADD}(r), p_j) \]
    
    \[
    \begin{align*}
    [1p_i \rightarrow p_i[1 \text{ and }]_{1p_i} p_i'' &\rightarrow p_i' p_i''[1] \\
    p_i' p_j[1] &\rightarrow [1p_i' p_j \text{ and }]_{1p_i} a_r[1] \rightarrow [1p_i'] a_r
    \end{align*}
    \]

  \item \[ p_i : (\text{SUB}(r), p_j, p_k) \in I \]
    
    \[
    \begin{align*}
    [1p_i'' p_i \rightarrow p_i'' p_i[1 \text{ and }]_{1p_i' a_r} &\rightarrow p_i' a_r[1] \\
    p_i' p_j[1] &\rightarrow [1p_i' p_j \text{ and }]_{1p_i''}[1] \rightarrow [1p_i'] \text{ or } \\
    p_i''[1] &\rightarrow [1p_i'' \text{ and }]_{1p_i} p_i''' \rightarrow p_i' p_i'''[1] \\
    p_i' p_k[1] &\rightarrow [1p_i' p_k \text{ and }]_{1p_i''}[1] \rightarrow [1p_i'']
    \end{align*}
    \]
\end{itemize}

\text{sym}(1), \text{ anti}(1):

\begin{itemize}
  \item \[ p_i : (\text{ADD}(r), p_j) \]
    
    a_r[1p_i \rightarrow p_i[1 a_r \text{ and }]_{1p_i} p_j[1] \rightarrow [1p_j]

  \item \[ p_i : (\text{SUB}(r), p_j, p_k) \]
    
    \[
    \begin{align*}
    p_i''[1] p_i &\rightarrow p_i'' p_i[1 \text{ and }]_{1p_i'} a_r \rightarrow a_r[1p_i' \\
    p_j[1] p_i'' &\rightarrow p_i'' p_j[1 \text{ and }]_{1p_i} p_i'[1] \rightarrow p_i'[1] \\
    p_i''[1] p_i'' &\rightarrow p_i'' p_i''[1 \text{ and }]_{1p_i'} p_i' \rightarrow p_i'[1] \rightarrow [1p_i'] \\
    p_k[1] p_i''' &\rightarrow p_i''' p_k[1 \text{ and }]_{1p_i'} p_i' \rightarrow p_i'[1]
    \end{align*}
    \]
sym(2):
The “garbage” – the symbols $p'_i$, $p''_i$, $p'''_i$ – cannot be removed from the skin membrane, which does not matter for the accepting case of a P automaton; taking the same construction for generating P systems, this is a challenging question - either we do not care about these “garbage” symbols or we have to add an output membrane.

sym(1), anti(1):
Only the final label $p_h$ remains in the skin membrane when the P automaton accepts by halting, hence, by adding the rule $[1p_h \rightarrow p_h]$ we even end up with an empty skin membrane. In other words, we could also consider these P automata with minimal antiport and uniport rules as generating mechanisms yielding their results as the numbers of objects in the skin membrane, without having any additional garbage.
Any partition of rules used for the control of the rules to be applied together consists of only one rule; hence, we can use the same partitioning of rules for the definition of minimal parallelism as well as of 1-restricted minimal parallelism.

**Corollary**

For $X \in \{N, Ps\}$, $X \in \{min, min_1\}$,

$$XRE = XO \ast P_1 K_\ast (Y) [sym_2]$$

$$XO \ast P_1 K_\ast (Y) [anti_1, sym_1].$$
Conclusion

Communication P automata with rule control can accept any recursively enumerable set of (vectors of) natural numbers

► working in the transition modes \( \text{max}, \text{min}, \text{min}_1 \) with
► \( \text{sym}(2) \)-rules or \( \text{sym}(1), \text{anti}(1) \) rules
► in only one membrane.

Future Research

► consider the transition modes \( \text{min}_k \) for \( k \geq 2 \);
► consider other kinds of rules;
► investigate corresponding generating cases, especially check whether “garbage” symbols can be avoided when using \( \text{sym}(2) \) rules;
► ....
THANK YOU FOR YOUR ATTENTION!