The Power of Maximal Parallelism in P Systems*

Oscar H. Ibarra[1], Hsu-Chun Yen[2], and Zhe Dang[3]

[1] Department of Computer Science,
University of California, Santa Barbara, CA, USA
ibarra@cs.ucsb.edu
[2] Department of Electrical Engineering,
National Taiwan University, Taipei, Taiwan, R.O.C.
[3] School of Electrical Engineering and Computer Science,
Washington State University, Pullman, WA, USA

Abstract. We consider the following definition (different from the standard definition in the literature) of "maximal parallelism" in the application of evolution rules in a P system G: Let $R = \{r_1, ...r_k\}$ be the set of (distinct) rules in the system. G operates in maximal parallel mode if at each step of the computation, a maximal subset of R is applied, and at most one instance of any rule is used at every step (thus at most k rules are applicable at any step). We refer to this system as a maximally parallel system. We look at the computing power of P systems under three semantics of parallelism. For a positive integer $n \leq k$, define:

n-Max-Parallel: At each step, nondeterministically select a maximal subset of at most n rules in R to apply (this implies that no larger subset is applicable).

$\leq n$-Parallel: At each step, nondeterministically select any subset of at most n rules in R to apply.

n-Parallel: At each step, nondeterministically select any subset of exactly n rules in R to apply.

In all three cases, if any rule in the subset selected is not applicable, then the whole subset is not applicable. When $n = 1$, the three semantics reduce to the **Sequential** mode.

We focus on two popular models of P systems: multi-membrane catalytic systems and communicating P systems. We show that for these systems, **n-Max-Parallel** mode is strictly more powerful than any of the following three modes: **Sequential**, **$\leq n$-Parallel**, or **n-Parallel**. For example, it follows from the result in [7] that a maximally parallel communicating P system is universal for $n = 2$. However, under the three limited modes of parallelism, the system is equivalent to a vector addition system, which is known to only define a recursive set. These generalize and refine the results for the case of 1-membrane systems recently reported in [3]. Some of the present results are rather surprising. For example, we show that a **Sequential** 1-membrane communicating P

* The research of Oscar H. Ibarra was supported in part by NSF Grants IIS-0101134, CCR-0208595, and CCF-0430945. The research of Zhe Dang was supported in part by NSF Grant CCF-0430531.

system can only generate a semilinear set, whereas with k membranes, it is equivalent to a vector addition system for any $k \geq 2$ (thus the hierarchy collapses at 2 membranes - a rare collapsing result for nonuniversal P systems). We also give another proof (using vector addition systems) of the known result [6] that a 1-membrane catalytic system with only 3 catalysts and (non-prioritized) catalytic rules operating under 3-**Max-Parallel** mode can simulate any 2-counter machine M. Unlike in [6], our catalytic system needs only a *fixed* number of noncatalysts, independent of M.

A simple cooperative system (SCO) is a P system where the only rules allowed are of the form $a \rightarrow v$ or of the form $aa \rightarrow v$, where a is a symbol and v is a (possibly null) string of symbols not containing a. We show that a 9-**Max-Parallel** 1-membrane SCO is universal.

1 Introduction

There has been a flurry of research activities in the area of membrane computing (a branch of molecular computing) initiated five years ago by Gheorghe Paun [13]. Membrane computing identifies an unconventional computing model, namely a P system, from natural phenomena of cell evolutions and chemical reactions. Due to the built-in nature of maximal parallelism inherent in the model, P systems have a great potential for implementing massively concurrent systems in an efficient way that would allow us to solve currently intractable problems (in much the same way as the promise of quantum and DNA computing) once future bio-technology (or silicon-technology) gives way to a practical bio-realization (or chip-realization).

The Institute for Scientific Information (ISI) has recently selected membrane computing as a fast "Emerging Research Front" in Computer Science (*http://esi-topics.com/ erf/october2003.html*). A P system is a computing model, which abstracts from the way the living cells process chemical compounds in their compartmental structure. Thus, regions defined by a membrane structure contain objects that evolve according to given rules. The objects can be described by symbols or by strings of symbols, in such a way that multisets of objects are placed in regions of the membrane structure. The membranes themselves are organized as a Venn diagram or a tree structure where one membrane may contain other membranes. By using the rules in a nondeterministic, maximally parallel manner, transitions between the system configurations can be obtained. A sequence of transitions shows how the system is evolving. Various ways of controlling the transfer of objects from a region to another and applying the rules, as well as possibilities to dissolve, divide or create membranes have been studied. P systems were introduced with the goal to abstract a new computing model from the structure and the functioning of the living cell (as a branch of the general effort of Natural Computing – to explore new models, ideas, paradigms from the way nature computes). Membrane computing has been quite successful: many models have been introduced, most of them Turing complete and/or able to solve computationally intractable problems (NP-complete, PSPACE-complete) in a feasible time (polynomial), by trading space for time. (See the P system website at *http://psystems.disco.unimib.it* for a large collection of papers in the area, and in particular the monograph [14].)

As already mentioned above, in the standard semantics of P systems [13–15], each evolution step of a system G is a result of applying all the rules in G in a maximally parallel manner. More precisely, starting from the initial configuration, w, the system goes through a sequence of configurations, where each configuration is derived from the directly preceding configuration in one step by the application of a multi-set of rules, which are chosen nondeterministically. For example, a catalytic rule $Ca \rightarrow Cv$ in membrane q is applicable if there is a catalyst C and an object (symbol) a in the preceding configuration in membrane q. The result of applying this rule is the evolution of v from a. If there is another occurrence of C and another occurrence of a, then the same rule or another rule with Ca on the left hand side can be applied. Thus, in general, the number of times a particular rule is applied at anyone step can be unbounded. We require that the application of the rules is maximal: all objects, from all membranes, which *can be* the subject of local evolution rules *have to* evolve simultaneously. Configuration z is reachable (from the starting configuration) if it appears in some execution sequence; z is halting if no rule is applicable on z.

In this paper, we study a different definition of maximal parallelism. Let G be a P system and $R = \{r_1, ..., r_k\}$ be the set of (distinct) rules in all the membranes. (Note that r_i uniquely specifies the membrane the rule belongs to.) We say that G operates in maximal parallel mode if at each step of the computation, a maximal subset of R is applied, and at most one instance of any rule is used at every step (thus at most k rules are applicable at any step). For example, if r_i is a catalytic rule $Ca \rightarrow Cv$ in membrane q and the current configuration has two C's and three a's in membrane q, then only one a can evolve into v. Of course, if there is another rule r_j, $Ca \rightarrow Cv'$, in membrane q, then the other a also evolves into v'. Throughout the paper, we will use this definition of maximal parallelism. Here, we look at the computing power of P systems under three semantics of parallelism. For a positive integer $n \leq k$, define:

n-Max-Parallel: At each step, nondeterministically select a maximal subset of at most n rules in R to
apply (this implies that no larger subset is applicable).

$\leq n$-Parallel: At each step, nondeterministically select any subset of at most n rules in R to apply.

n-Parallel: At each step, nondeterministically select any subset of exactly n rules in R to apply.

In all three cases, if any rule in the subset selected is not applicable, then the whole subset is not applicable. When $n = 1$, the three semantics reduce to the **Sequential** mode.

In the next four sections, we investigate the computing power of two popular models of P systems with respect to the above semantics of parallelism – the catalytic P systems and the communicating P systems.

We should mention some related work on P systems operating in sequential and limited parallel modes. Sequential variants of P systems have been studied, in a different framework, in [5]. There, generalized P systems (GP-systems) were considered and were shown to be able to simulate graph controlled grammars. Our notion of limited parallelism seems to correspond to "cooperation modes" in cooperating distributed grammar systems, investigated in [2].

Because of space limitation, many of the proofs are omitted here. They will appear in the journal version of this paper.

2 Multi-membrane Catalytic Systems

2.1 Maximally Parallel CS

First we recall the definition of a catalytic system (CS). The membranes (regions) are organized in a hierarchical (tree) structure and are labeled 1, 2, .., m for some m, with the outermost membrane (the skin membrane) labeled 1. At the start of the computation, there is a distribution of *catalysts* and *noncatalysts* in the membranes (the distribution represents the initial configuration of the system). Each membrane may contain a finite set of catalytic rules of the form $Ca \rightarrow Cv$, where C is a catalyst, a is a noncatalyst, and v is a (possibly null) string of noncatalysts. When this rule is applied, the catalyst remains in the membrane the rule is in, symbol a is deleted from the membrane, and the symbols comprising v (if nonnull) are transported to other membranes in the following manner. Each symbol b in v has a designation or target, i.e., it is written b_x, where x can be *here*, *out*, or in_j. The designation *here* means that the object b remains in the membrane containing it (we usually omit this target, when it is understood). The designation *out* means that the object is transported to the membrane directly enclosing the membrane that contains the object; however, we do not allow any object to be transported out of the skin membrane. The designation in_j means that the object is moved into a membrane, labeled j, that is directly enclosed by the membrane that contains the object.

It is important to note that our definition of catalytic system is different from what is usually called catalytic system in the literature. Here, we do not allow rules without catalysts, i.e., rules of the form $a \rightarrow v$. Thus our systems use only purely catalytic rules.

Suppose that S is a CS with m membranes. Let $\{a_1, ..., a_n\}$ be the set of noncatalyst symbols (objects) that can occur in the configurations of S. Let $w = (w_1, ..., w_m)$ be the initial configuration, where w_i represents the catalysts and noncatlysts in membrane i. (Note that w_i can be null.) Each reachable configuration of S is an nm-tuple $(v_1, ..., v_m)$, where v_i is an n-tuple representing the multiplicities of the symbols $a_1, ..., a_n$ in membrane i. Note that we do not include the catalysts in considering the configuration as they are not changed (i.e., they remain in the membranes containing them, and their numbers remain the same during the computation). Hence the set of all reachable configurations of S, denoted by $R(S)$ is a subset of $\mathbf{N}^{mn}$. The set of all halting reachable configurations is denoted by $R_h(S)$.

2.2 Sequential CS

In a sequential multi-membrane CS, each step of the computation consists of an application of a single nondeterministically chosen rule, i.e., the membrane and rule within the membrane to apply are chosen nondeterministically. We show below that sequential multi-membrane CS's define exactly the semilinear sets.

We need the definition of a vector addition system. An n-dimensional *vector addition system* (VAS) is a pair $G = \langle x, W \rangle$, where $x \in \mathbf{N}^n$ is called the *start point* (or *start vector*) and W is a finite set of transition vectors in $\mathbf{Z}^n$, where $\mathbf{Z}$ is the set of all integers (positive, negative, zero). Throughout this paper, for a $w \in \mathbf{Z}^n$ we write $w \geq 0$ to mean that w has only nonnegative components (i.e., $w \in \mathbf{N}^n$). The *reachability set* of the VAS $\langle x, W \rangle$ is the set $R(G) = \{z \mid$ for some j, $z = x + v_1 + ... + v_j$, where, for all $1 \leq i \leq j$, each $v_i \in W$ and $x+v_1+...+v_i \geq 0\}$. Note that $R(G)$ is the smallest set satisfying the following two properties: (1) $x \in R(G)$, and (2) whenever $z \in R(G), v \in W$, and $z + v \in \mathbf{N}^n$, then $z+v \in R(G)$. The *halting reachability set* $R_h(G) = \{z \mid z \in R(G), z+v \ngeq 0$ for every v in $W\}$.

An n-dimensional *vector addition system with states* (VASS) is a VAS $\langle x, W \rangle$ together with a finite set T of transitions of the form $p \rightarrow (q, v)$, where q and p are states and v is in W. The meaning is that such a transition can be applied at point y in state p and yields the point $y+v$ in state q, provided that $y+v \geq 0$. The VASS is specified by $G = \langle x, W, T, p_0 \rangle$, where p_0 is the starting state. Assuming that the set of states of G is $\{p_0, ..., p_k\}$ (for some $k \geq 0$), the *reachability set* of VASS G is $R(G)=\{(i, w) \in \mathbf{N}^{n+1} \mid 1 \leq i \leq k, (p_i, w)$ is reachable from $(p_0, x)\}$. The *halting reachability set* $R_h(G)=\{(i, w) \mid (i, w) \in R(G)$ and no transition is applicable in $(p_i, w)\}$.

The *reachability problem* for a VASS (respectively, VAS) G is to determine, given a vector y, whether y is in $R(G)$. The *equivalence problem* is to determine given two VASS (respectively, VAS) G and G', whether $R(G) = R(G')$. Similarly, one can define the reachability problem and equivalence problem for halting configurations.

The following summarizes the known results concerning VAS and VASS [17, 8, 1, 9, 12]:

Theorem 1.

1. *Let G be an n-dimensional VASS. We can effectively construct an $(n+3)$-dimensional VAS G' that simulates G.*
2. *If G is a 2-dimensional VASS, then $R(G)$ is an effectively computable semilinear set.*
3. *There is a 3-dimensional VASS G such that $R(G)$ is not semilinear.*
4. *If G is a 5-dimensional VAS, then $R(G)$ is an effectively computable semilinear set.*
5. *There is a 6-dimensional VAS G such that $R(G)$ is not semilinear.*
6. *The reachability problem for VASS (and hence also for VAS) is decidable.*
7. *The equivalence problem for VAS (and hence also for VASS) is undecidable.*

Clearly, it follows from part 6 of the theorem above that the halting reachability problem for VASS (respectively, VAS) is decidable.

A *communication-free VAS* is a VAS where in every transition, at most one component is negative, and if negative, its value is -1. Communication-free VAS's are equivalent to communication-free Petri nets, which are also equivalent to commutative context-free grammars [4, 10]. It is known that they have effectively computable semilinear reachability sets [4].

Our first result shows that a sequential CS is weaker than a maximally parallel CS.

Theorem 2. *Every sequential multi-membrane CS S can be simulated by a communication-free VAS G, and vice versa.*

Proof. Let S be an m-membrane CS with noncatalysts $a_1, ..., a_n$. Suppose that the start configuration of $w = (w_1, ..., w_m)$ has k catalysts $C_1, ..., C_k$. We may assume, without loss of generality by adding new catalysts and rules if necessary, that each C_i occurs at most once in w_i ($1 \leq i \leq m$). Number all the rules in S by $1, ..., s$. Note that the rule number uniquely determines the membrane where the rule is applicable.

We first transform S to a new system S' by modifying the rules and the initial configuration w. S' will now have catalysts $C_1, ..., C_k, Q_1, ..., Q_s$ and noncatalysts $a_1, ..., a_n, d_1, ..., d_s$. The component w_q of the initial configuration in membrane q will now be w_q plus each Q_h for which rule number h is in membrane q. The rules of S' are defined as follows:

Case 1: Suppose that $C_j a_i \to C_j v$ is a rule in membrane q of S, and a_i does not appear in v with designation (target) *here*. Then this rule is in membrane q of S'.

Case 2: Suppose that $C_j a_i \to C_j a_i^t v$ is rule number r and $t \geq 1$. Suppose that this rule is in membrane q, with the target of each a_i in a_i^t being *here*, and v does not contain any a_i with target *here*. Then the following rules are in membrane q of S': $C_j a_i \to C_j d_r^t v$ and $Q_r d_r \to Q_r a_i$. In the above rules, the target for a_i and each d_r in the right-hand side of the rules is *here*.

Clearly, S' simulates S, and S' has the property that in each rule $Xb \to Xv$ (where X is a catalyst, b is a noncatalyst, and v a string of noncatalysts), v does not contain a b with target *here*. It is now obvious that each rule $Xb \to Xv$ in S' can be transformed to a VAS transition rule of $mn + s$ components, where the component of the transition corresponding to noncatalyst b is -1, and the other components (corresponding to the target designations in v) are nonnegative. Thus, the VAS is communication free.

Conversely, let G be a communication-free VAS. We construct a sequential 1-membrane CS S which has one catalyst C, noncatalysts $\#, a_1, ..., a_k$, and starting configuration $C\#w$, where w corresponds to the starting vector of G. Suppose that $(j_1, ..., j_{m-1}, j_m, j_{m+1}, ..., j_k)$ is a transition in G.

Case 1: $j_m = -1$ and all other j_i's are nonnegative. Then the following rule is in S: $Ca_m \to Ca_1^{j_1}...a_{m-1}^{j_{m-1}}a_{m+1}^{j_{m+1}}...a_k^{j_k}$.

Case 2: All the j_i's are nonnegative. Then the following rule is in S: $C\# \to C\#a_1^{j_1}...a_k^{j_k}$.

Clearly, S simulates G. In fact, $R(G) = R(S) \times \{1\}$. □

Corollary 1.

1. *If S is a sequential multi-membrane CS, then $R(S)$ and $R_h(S)$ are effectively computable semilinear sets.*
2. *The reachability problem (whether a given configuration is reachable) for sequential multi-membrane CS's is NP-complete.*

Since a communication-free VAS can be simulated by a sequential 1-membrane CS (from conversely in the proof of Theorem 2), we have:

Corollary 2. *The following are equivalent: communication-free VAS, sequential multi-membrane CS, sequential 1-membrane CS.*

2.3 CS Under Limited Parallelism

Here we look at the computing power of the multi-membrane CS under three semantics of parallelism (namely, n-**Max-Parallelism**, $\leq n$-**Parallelism**, and n-**Parallelism**) defined in Section 1. We can show the following:

Theorem 3. *For $n = 3$, a 1-membrane CS operating under the* n-**Max-Parallel** *mode can define any recursively enumerable set. For any n, a multi-membrane CS operating under* $\leq n$-**Parallel** *mode or* n-**Parallel** *mode can be simulated by a VASS.*

2.4 3-Max-Parallel 1-Membrane CS

As noted above, it is known that a 3-**Max-Parallel** 1-membrane CS is universal [6] in that it can simulate any 2-counter machine M. We show in this section that the universality result of [6] can be obtained in terms of communication-free VAS. Later we improve this result by showing that, in fact, the 1-membrane CS need no more than k noncatalysts for some *fixed* k, independent of M.

Consider an n-dimensional communication-free VAS $G = \langle x, W \rangle$ with its set of addition vectors W partitioned into three disjoint sets W_1, W_2 and W_3. Under the 3-**Max-Parallel** mode, at each step G nondeterministically applies a maximal set of at most 3 addition vectors simultaneously to yield the next vector; however, from each set $W_i, 1 \leq i \leq 3$, at most one addition vector can be chosen.

Acting as either *acceptors* or *generators*, the following result shows the equivalence of 2-counter machines and communication-free VAS operating under the 3-**Max-Parallel** mode.

Theorem 4. *Let M be a 2-counter machine with two counters C_1 and C_2. There exist an n-dimensional VAS $G = \langle x, (W_1, W_2, W_3)\rangle$ under the* **3-Max-Parallel** *mode and a designated coordinate l such that M accepts on initial counter values $C_1 = m$ and $C_2 = 0$ iff*

1. *(generator:) $m \in \{v(l) \mid v \in R_h(G)\}$;*
2. *(acceptor:) from start vector x with $x(l) = m$, $R_h(G) \neq \emptyset$, i.e., G has a halting computation.*

Here n is bounded by a function of the number of states of M.

By assigning for each $1 \leq i \leq 3$, a catalyst C_i for the set of addition vectors W_i, defining a distinct noncatalyst symbol for each position in the addition vector, and converting each vector in W_i to a rule of the form $C_i a \to C_i v$, where a is a noncatalyst and v is a (possibly null) string of noncatalysts, the following result can easily be obtained from Theorem 4.

Corollary 3. *Let M be a 2-counter machine with two counters. There exists a 1-membrane 3-***Max-Parallel** *CS S with catalysts C_1, C_2, C_3 and n noncatalysts with a designated noncatalyst symbol a_l such that M accepts on initial counter values m and 0, respectively, iff*

1. *(generator:) $m \in \{\#_{a_l}(y) \mid y \in R_h(S)\}$;*
2. *(acceptor) if S starts with initial configuration $(a_l)^m y$, for some y not containing a_l, then $R_h(S) \neq \emptyset$.*

We note that in the corollary above, the CS operates in 3-**Max-Parallel** mode. Now the catalyst C_1 (resp. C_2) is needed to make sure that at most one addition vector in W_1 (resp. W_2) is simulated by the CS at each step. However, catalyst C_3 is not really needed in that we can convert each addition vector in W_3 to a rule of the form $a \to v$, i.e., a *noncooperative rule* (without a catalyst). Thus, the system can be constructed to have only *two* catalysts with catalytic rules and noncooperative rules. This was also shown in [6]. However, the degree of maximal parallelism in the system is no longer 3 (because now more than one noncooperative rule may be applicable at each step). It can be shown that at any point, no more than 3 noncooperative rules are applicable. This in turn implies that the degree of maximal parallelism now becomes 5 (two catalysts plus 3 noncooperative rules). Note also that n, the dimension of the communication-free VAS, which translates to the number of noncatalysts for the system, is also a function of the number of states, hence is unbounded.

We can improve the above results. We need the following lemma.

Lemma 1. *There exists a 2-counter machine U with counters C_1 and C_2 that is universal in the following sense. When U is given a description of an arbitrary 2-counter machine M as a positive integer in C_2 and an input m in C_1, U accepts iff M with input m on its first counter and 0 on its other counter accepts.*

Therefore, we have:

Corollary 4. *There exists a* fixed positive integer n *such that if* $L \subseteq \mathbf{N}$ *is* any *recursively enumerable set of nonnegative integers, then:*

1. *L can be generated (accepted) by a 1-membrane* 3-**Max-Parallel** *CS with 3 catalysts and n noncatalysts.*
2. *L can be generated (accepted) by a 1-membrane* 5-**Max-Parallel** *P system with 2 catalysts and n noncatalysts with catalytic and noncooperative rules.*

2.5 9-Max-Parallel 1-Membrane CS with One Catalyst

We now look at a model of a 1-membrane CS with only *one* catalyst C with initial configuration $C^k x$ for some string x of noncatalysts (thus, there are k copies of C). The rules allowed are of the form $Ca \rightarrow Cv$ or of the form $Caa \rightarrow Cv$, i.e., C catalizes two copies of an object. Clearly the system operates in maximally parallel mode, but uses no more than k rules in any step. We call this system 1GCS. This system is equivalent to a restricted form of cooperative P system [13, 14]. A simple cooperative system (SCO) is a P system where the rules allowed are of the form $a \rightarrow v$ or of the form $aa \rightarrow v$. Moreover, there is some fixed integer k such that the system operates in maximally parallel mode, but uses no more than k rule instances in any step. We can show the following:

Theorem 5. *1GCS (hence, also SCO) operating under the* 9-**Max-Parallel** *mode is universal.*

3 Sequential 1-Membrane Communicating P Systems

Consider the model of a communicating P system (CPS) with only *one* membrane, called the skin membrane [16]. The rules are of the form: (1) $a \rightarrow a_x$, (2) $ab \rightarrow a_x b_y$, (3) $ab \rightarrow a_x b_y c_{come}$, where a, b, c are objects, x, y (which indicate the directions of movements of a and b) can only be *here* (i.e., the object remains in the membrane) or *out* (i.e., the object is expelled into the environment). The third rule brings in an object c from the environment into the skin membrane. In the sequel, we omit the designation *here*, so that objects that remain in the membrane will not have this subscript. There is a fixed finite set of rules in the membrane. At the beginning, there is a fixed configuration of objects in the membrane.

Assume that the computation is *sequential*; i.e., at each step there is only one application of a rule (to one instance). So, e.g., if nondeterministically a rule like $ab \rightarrow a_{here} b_{out} c_{come}$ is chosen, then there must be at least one a and one b in the membrane. After the step, a remains in the membrane, b is thrown out of the membrane, and c comes into the membrane. There may be several a's and b's, but only one application of the rule is applied. Thus, there is *no* parallelism involved. The computation halts when there is no applicable rule. We are interested in the multiplicities of the objects when the system halts.

One can show that a 1-membrane CPS can be simulated by a vector addition system (VAS) (this is a special case of a theorem in the next section). However, the converse is not true – it was shown in [3] that a sequential 1-membrane CPS can only define a semilinear set.

4 Sequential 1-Membrane Extended CPS (ECPS)

Recall that 1-membrane CPS's operating sequentially define only semilinear sets. In contrast, we shall see in the next section that sequential 2-membrane CPS's are equivalent to VASS.

There is an interesting generalization of a 1-membrane CPS, we call extended CPS (or ECPS) – we add a fourth type of rule of the form: $ab \rightarrow a_x b_y c_{come} d_{come}$. That is, two symbols can be imported from the environment. We shall see below that ECPS's are equivalent to VASS's.

Let G be an n-dimensional VASS. Clearly, by adding new states, we may assume that all transitions in G have the form: $p_i \rightarrow (p_j, +1_h)$, $\;p_i \rightarrow (p_j, -1_h)$. The above is a short-hand notation. The $+1_h$ is addition of 1 to the h-th coordinate, and -1_h is subtraction of 1 from the h-th coordinate. All other coordinates are unchanged. Note at each step, the state uniquely determines whether it is a '+1 transition' or a '-1 transition'.

For constructing the ECPS S equivalent to G, we associate symbol p_i for every state of the VASS, a_h for every coordinate (i.e., position) h in the transition. We also define a new special symbol c. So the ECPS has symbols $p_1, ...p_s$ (s is the number of states), $a_1, ..., a_n$ (n is dimension of the VASS), and c.

Then a transition of the form $p_i \rightarrow (p_j, +1_h)$ in G is simulated by the following rule in S: $p_i c \rightarrow p_{i(out)} c_{here} p_{j(come)} a_{h(come)}$

A transition of the form $p_i \rightarrow (p_j, -1_h)$ in G is simulated by the following rule in S: $p_i a_h \rightarrow p_{i(out)} a_{h(out)} p_{j(come)}$.

If the VASS G has starting point $\langle p_1, v\rangle$, where $v = (i_1, ..., i_n)$ and p_1 is the start state, then ECPS S starts with the word $p_1 a_1^{i_1} ... a_n^{i_n} c$. Clearly, S simulates G.

Conversely, suppose we are given an ECPS S over symbols $a_1, ..., a_n$ with initial configuration w and rules $R_1, ..., R_k$. The VASS G has states $R_0, R_1, R_1', ...R_k, R_k'$ and starting point $\langle R_0, v_0\rangle$, where v_0 is the n-dimensional vector in $\mathbf{N}^n$ representing the multiplicities of the symbols in the initial configuration w. The transitions of G are defined as follows:

1. $R_0 \rightarrow (R_i, zero)$ for every $1 \leq i \leq k$ is a transition, where $zero$ represents the zero vector.
2. If R_i is a rule of the form $a_h \rightarrow a_{hx}$, then the following are transitions:
   $R_i \rightarrow (R_i', -1_h)$
   $R_i' \rightarrow (R_j, d_{hx})$ for every $1 \leq j \leq k$, where $d_{hx} = 0_h$ if $x = (out)$ and $d_{hx} = +1_h$ if $x = (here)$.
   (As before, $-1_h, 0_h, +1_h$ mean subtract 1, add 0, add 1 to h, respectively; all other coordinates are unchanged.)
3. If R_i is a rule of the form $a_h a_r \rightarrow a_{hx} a_{ry}$, then the following are transitions:
   $R_i \rightarrow (R_i', -1_h, -1_r)$
   $R_i' \rightarrow (R_j, d_{hx}, d_{ry})$ for every $1 \leq j \leq k$, where d_{hx} and d_{ry} are as defined above.
   (Note that if $h = r$, then $(R_i', -1_h, -1_r)$ means $(R_i', -2_h)$, i.e., subtract 2 from coordinate h.)

4. If R_i is a rule of the form $a_h a_r \rightarrow a_{hx} a_{ry} a_{s(come)}$, then the following are transitions:
   $R_i \rightarrow (R_i', -1_h, -1_r)$
   $R_i' \rightarrow (R_j, d_{hx}, d_{ry}, +1_s)$ for every $1 \leq j \leq k$, where d_{hx} and d_{ry} are as defined above.
5. If R_i is a rule of the form $a_h a_r \rightarrow a_{hx} a_{ry} a_{s(come)} a_{t(come)}$, then the following are transitions:
   $R_i \rightarrow (R_i', -1_h, -1_r)$
   $R_i' \rightarrow (R_j, d_{hx}, d_{ry}, +1_s, +1_t)$ for every $1 \leq j \leq k$, where d_{hx} and d_{ry} are as defined above.

It follows from the construction above that G simulates S. Thus, we have:

Theorem 6. *Sequential 1-membrane ECPS and VASS are equivalent.*

We can generalize rules of an ECPS further as follows:

1. $a_{i_1} ... a_{i_h} \rightarrow a_{i_1 x_1} ... a_{i_1 x_h}$
2. $a_{i_1} ... a_{i_h} \rightarrow a_{i_1 x_1} ... a_{i_1 x_h} c_{j_{1come}} ... c_{j_{lcome}}$

where $h, l \geq 1$, and $x_m \in \{here, out\}$ for $1 \leq m \leq h$, and the a's and c's are symbols. Call this system ECPS+. Generalizing the constructions in the proof of Theorem 6, we can show ECPS+ is still equivalent to a VASS. Thus, we have:

Corollary 5. *The following systems are equivalent: Sequential 1-membrane ECPS, sequential 1-membrane ECPS+, and VASS.*

Using rules of types of 1 and 2 above, we can define the three versions of parallelism as in Section 2.3, and we can prove the following result. Note that the first part was shown in [7].

Theorem 7. *For $n = 2$, a 1-membrane CPS (and, hence, also 1-membrane ECPS+) operating under the n-**Max-Parallel** mode can define a recursively enumerable set. For any n, a 1-membrane ECPS+ operating under $\leq n$-**Parallel** mode or n-**Parallel** mode is equivalent to a VASS.*

5 Multi-membrane CPS and ECPS

In this section, we look at CPS and ECPS with multiple membranes. Now the subscripts x, y in the CPS rules $a \rightarrow a_x$, $ab \rightarrow a_x b_y$, $ab \rightarrow a_x b_y c_{come}$ (and $ab \rightarrow a_x b_y c_{come} d_{come}$ in ECPS) can be *here*, *out*, or in_j. As before, *here* means that the object remains in the membrane containing it, *out* means that the object is transported to the membrane directly enclosing the membrane that contains the object (or to the environment if the object is in the skin membrane), and *come* can only occur within the outermost region (i.e., skin membrane). The designation in_j means that the object is moved into a membrane, labeled j, that is directly enclosed by the membrane that contains the object.

In Section 3, we saw that a sequential 1-membrane CPS can only define a semilinear set. We can show that if the system has two membranes, it can simulate a vector addition system. Thus, we have:

Theorem 8. *A sequential 2-membrane CPS S can simulate a VASS G.*

In Theorem 6, we saw that a sequential 1-membrane ECPS can be simulated by a VASS. The construction can be extended to multi-membrane ECPS. Recall that we now allow rules of the form: $ab \rightarrow a_x b_y c_{come} d_{com}$.

Suppose that S has m membranes. Let $\{a_1, ..., a_n\}$ be the set of symbols (objects) that can occur in the configurations of S. Then each reachable configuration of S is an mn-tuple $(v_1, ..., v_m)$, where v_q is an n-tuple representing the multiplicities of the symbols $a_1, ..., a_n$ in membrane q. Then the set of all reachable configurations of S is a subset of $\mathbf{N}^{mn}$. Let $R_1, ..., R_k$ be the rules in S. Note that R_i not only gives the rule but also the membrane where it appears. The construction of the VASS G simulating S is similar to the construction described in the second part of the proof of Theorem 6. In fact the construction also works for ECPS+. Since a sequential 2-membrane CPS can simulate a VASS, we have:

Theorem 9. *The following are equivalent: VASS, sequential 2-membrane CPS, sequential 1-membrane ECPS, sequential multi-membrane ECPS, and sequential multi-membrane ECPS+.*

Finally, we observe that Theorem 7 extends to multi-membrane CPS:

Theorem 10. *For any n, a multi-membrane ECPS+ operating under $\leq n$-***Parallel** *mode or n-***Parallel** *mode is equivalent to a VASS.*

6 Conclusion

We showed in this paper that P systems that compute in sequential or limited parallel mode are strictly weaker than systems that operate with maximal parallelism for two classes of systems: multi-membrane catalytic systems and multi-membrane communicating P systems. Our proof techniques can be used to show that many of the P systems that have been studied in the literature (including ones with membrane dissolving rules) operating under sequential or limited parallelism with unprioritized rules can be simulated by vector addition systems.

References

1. H. G. Baker. Rabin's proof of the undecidability of the reachability set inclusion problem for vector addition systems. In *C.S.C. Memo 79, Project MAC, MIT*, 1973.
2. E. Csuhaj-Varju, J. Dassow, J. Kelemen, and Gh. Paun. *Grammar Systems: A Grammatical Approach to Distribution and Cooperation.* Gordon and Breach Science Publishers, Inc., 1994.
3. Z. Dang and O. H. Ibarra. On P systems operating in sequential and limited parallel modes. In *Pre-Proc. 6th Workshop on Descriptional Complexity of Formal Systems*, 2004.

4. J. Esparza. Petri nets, commutative context-free grammars, and basic parallel processes. In *Proc. Fundamentals of Computer Theory*, volume 965 of *Lecture Notes in Computer Science*, pages 221–232. Springer, 1995.
5. R. Freund. Sequential P-systems. Available at *http://psystems.disco.unimib.it*, 2000.
6. R. Freund, L. Kari, M. Oswald, and P. Sosik. Computationally universal P systems without priorities: two catalysts are sufficient. Available at *http://psystems.disco.unimib.it*, 2003.
7. R. Freund and A. Paun. Membrane systems with symport/antiport rules: universality results. In *Proc. WMC-CdeA2002*, volume 2597 of *Lecture Notes in Computer Science*, pages 270–287. Springer, 2003.
8. M. H. Hack. The equality problem for vector addition systems is undecidable. In *C.S.C. Memo 121, Project MAC, MIT*, 1975.
9. J. Hopcroft and J.-J. Pansiot. On the reachability problem for 5-dimensional vector addition systems. *Theoretical Computer Science*, 8(2):135–159, 1979.
10. D.T. Huynh. Commutative grammars: The complexity of uniform word problems. *Information and Control*, 57:21–39, 1983.
11. O. Ibarra, Z. Dang, and O. Egecioglu. Catalytic P systems, semilinear sets, and vector addition systems. *Theoretical Computer Science*, 312(2-3): 379–399, 2004.
12. E. Mayr. Persistence of vector replacement systems is decidable. *Acta Informat.*, 15:309–318, 1981.
13. Gh. Paun. Computing with membranes. *Journal of Computer and System Sciences*, 61(1):108–143, 2000.
14. Gh. Paun. *Membrane Computing: An Introduction.* Springer-Verlag, 2002.
15. Gh. Paun and G. Rozenberg. A guide to membrane computing. *Theoretical Computer Science*, 287(1):73–100, 2002.
16. P. Sosik. P systems versus register machines: two universality proofs. In *Pre-Proceedings of Workshop on Membrane Computing (WMC-CdeA2002), Curtea de Arges, Romania*, pages 371–382, 2002.
17. J. van Leeuwen. A partial solution to the reachability problem for vector addition systems. In *Proceedings of STOC'74*, pages 303–309.