International Journal of Foundations of Computer Science
© World Scientific Publishing Company

ON ONE-MEMBRANE P SYSTEMS OPERATING IN SEQUENTIAL MODE

ZHE DANG
School of Electrical Engineering and Computer Science, Washington State University
Pullman, WA 99164, USA

and

OSCAR H. IBARRA
Department of Computer Science, University of California
Santa Barbara, CA 93106, USA

Received (received date)
Revised (revised date)
Communicated by Editor's name

ABSTRACT

In the standard definition of a P system, a computation step consists of a parallel application of a "maximal" set of nondeterministically chosen rules. Referring to this system as a parallel P system, we consider in this paper a sequential P system, in which each step consists of an application of a single nondeterministically chosen rule. We show the following:

1. For 1-membrane purely catalytic systems (pure CS's), the sequential version is strictly weaker than the parallel version in that the former defines (i.e., generates) exactly the semilinear sets, whereas the latter is known to define nonrecursive sets.
2. For 1-membrane communicating P systems (CPS's), the sequential version can only define a proper subclass of the semilinear sets, whereas the parallel version is known to define nonrecursive sets.
3. Adding a new type of rule of the form: $ab \rightarrow a_x b_y c_{come} d_{come}$ to the CPS (a natural generalization of the rule $ab \rightarrow a_x b_y c_{come}$ in the original model), where $x, y \in \{here, out\}$, to the sequential 1-membrane CPS makes it equivalent to a vector addition system.
4. Sequential 1-membrane symport/antiport systems (SA's) are equivalent to vector addition systems, contrasting the known result that the parallel versions can define nonrecursive sets.
5. Sequential 1-membrane SA's whose rules have radius 1, (1,1), (1,2) (i.e., of the form $(a, out), (a, in), (a, out; b, in), (a, out; bc, in)$) generate exactly the semilinear sets. However, if the rules have radius 1, (1,1), (2,1) (i.e., of the form $(ab, out; c, in)$), the SA's can only generate a proper subclass of the semilinear sets.

Keywords: Catalytic P system, communicating P system, symport/antiport system, parallel system, sequential system, vector addition system, semilinear set.

1. Introduction

A P system consists of a finite number of membranes, each of which contains a multiset of objects (symbols). The membranes are organized as a Venn diagram or a tree structure where membranes may contain other membranes. The dynamics of the system is governed by a set of rules associated with each membrane. Each rule specifies how objects evolve and move into neighboring membranes. A precise definition can be found in [16, 17, 18]. It has been introduced as a computing model which abstracts from the way live cells process chemical compounds in their "compartmental" (membrane) structure. Various models of P systems have been shown to be equivalent to Turing machines in computing power. For example, recent results in [19, 20, 6] show that P systems with one membrane (i.e., 1-region P systems) using only purely catalytic rules are already able to simulate two counter machines and hence universal [14].

In a P system [16, 17, 18], a step of the computation is an application of a multiset of nondeterministically chosen rules in parallel to the current configuration to obtain the next configuration. (Note that all the rules must be applicable at the same time.) The set of rules is maximal in the sense that no additional rule can be added to the set which still makes the resulting set of rules applicable. To distinguish this system from the one we define below, we call this system a *parallel P system.*

Now define a *sequential P system* as a P system in which a computation step consists of an application of a *single* rule in some membrane. The membrane and rule are chosen nondeterministically.

Consider only P systems that generate objects/symbols. Let $\Sigma = \{a_1, ..., a_n\}$ be a language alphabet. We say that a P system G generates a string $a_1^{i_1}...a_n^{i_n}$ (the order the symbols are written is not important as we are only interested in the multiplicities of each symbol) representing the n-tuple $(i_1, ..., i_n)$ of nonnegative integers if, when started with a fixed initial configuration w of symbols that may contain symbols in Σ and other symbols not in Σ, G can reach a configuration in a specified output membrane that contains $a_1^{i_1}...a_n^{i_n}$ (note that there may be other symbols not in Σ that are present in the configuration). The set of such strings is called the reachability set denoted by $R(G)$. If we are interested only in reachable halting configurations, then we denote such a reachability set by $R_h(G)$. Clearly $R_h(G) \subseteq R(G)$.

For convenience, we will also use the tuple of nonnegative integer representations instead of bounded string representations, i.e., $(i_1, ..., i_n)$ instead of $a_1^{i_1}...a_n^{i_n}$.

In this paper, we exhibit P systems where the sequential version is strictly weaker than the parallel version. We show that sequential 1-membrane purely catalytic systems (pure CS's) generate/define exactly the semilinear sets. However, in contrast, it follows from the constructions in [19, 20, 6] that parallel 1-membrane pure CS's can generate nonrecursive sets of tuples. In fact, recently, it was shown that only 3 catalysts are needed [6]. This result is not true when there is only 1 catalyst [11].

Similarly, for communicating P Systems (CPS's) introduced in [19], it follows

from the constructions in [7] that there are parallel 1-membrane CPS's that generate nonrecursive sets. But for sequential 1-membrane CPS's, we show that the sets generated are a proper subclass of the semilinear sets. Interestingly, if we allow the model of a CPS to use a new type of rules of the form: $ab \rightarrow a_x b_y c_{come} d_{come}$ (generalizing the rule $ab \rightarrow a_x b_y c_{come}$ in the original model), we show that the sequential version of this extended model is equivalent to a vector addition system (which is known to be equivalent to a Petri net). We also show that sequential 1-membrane symport/antiport systems (SA's) are equivalent to vector addition systems. Interestingly, sequential 1-membrane SA's whose rules have radius 1, (1,1), (1,2) (i.e., of the form $(a, out), (a, in), (a, out; b, in)$, $(a, out; bc, in)$) generate exactly the semilinear sets. However, if the rules have radius 1, (1,1), (2,1) (i.e., of the form $(ab, out; c, in)$), the SA's can only generate a proper subclass of the semilinear sets.

We note that current digital and bio technologies do not permit a direct implementation of a P system (under the parallel semantics). Therefore, it is important to study the time/space complexity trade-offs between a parallel P system and a sequential one and to further understand the power of the "maximal parallelism" in the definition of P systems. Our results show a number of cases where simulating a parallel P system with a sequential P system using the same types of rules is (im)possible. Currently, we are not able to formulate a naturally restricted and strongest class of P systems such that its parallel model and its sequential model have the same computing power (i.e., generate the same class of reachability sets). Such a restricted class is interesting since, within it, one can further study the description complexity trade-offs between a P system and a sequential one that can simulate the P system.

There has been some related work on P systems operating in sequential mode. For example, sequential variants of P systems have been studied, in a different framework, in [4, 5]. For example, in [4], generalized P systems (GP-systems) were considered and were shown to be able to simulate graph controlled grammars. A comparison between parallel and sequential modes of computation in a restricted model of a P automaton was also recently investigated in [2], where it was shown that the parallel version is equivalent to a linear space-bounded nondeterministic Turing machine (NTM) and the sequential version is equivalent to a simple type of a one-way $log\ n$ space-bounded NTM. A survey of the sequential P systems area can be found in [5].

The paper has five sections in addition to this section. Section 2 compares the computational power of 1-membrane purely catalytic systems that operate in sequential mode to 1-membrane purely catalytic systems that operate in parallel mode. Section 3 does a similar comparison for 1-membrane communicating P systems (CPS). Section 4 introduces a natural extension of 1-membrane CPS, called ECPS, and show that these systems are equivalent to vector addition systems. Section 5 looks at sequential 1-membrane symport/antiport systems. Section 6 is a brief conclusion.

We conclude this section with the following observation:

Observation: Let G be any P system. Suppose that $R_h(G)$ is the parallel halting

reachability set of G (thus we are looking at the parallel version) is nonrecursive. Then it is easy to see that the parallel reachability set $R(G)$ (i.e., all reachable configurations, halting or not) is also be nonrecursive. It follows that P systems that are Turing-universal have nonrecursive reachability sets.

2. 1-Membrane Catalytic Systems

First we recall the definition of a purely catalytic system (pure CS) with *one* membrane operating in parallel. Such a system G operates on two types of symbols: catalytic symbols called *catalysts* (denoted by capital letters C, D, etc) and noncatalytic symbols called *noncatalysts* (denoted by lower case letters a, b, c, d, etc). An evolution rule in G is of the form $Ca \rightarrow Cv$, where C is a catalyst, a is a noncatalyst, and v is a (possibly null) string (an obvious representation of a multiset) of noncatalysts. There are no rules of the form $a \rightarrow v$. Thus, we only consider "purely" catalytic rules. A pure CS G is specified by a finite set of rules together with an initial multiset (configuration) w_0, which is a string of catalysts and noncatalysts. As with the standard semantics of P systems [16, 17, 18], each evolution step of G is a result of applying all the rules in G in a maximally parallel manner. More precisely, starting from the initial configuration, w_0, 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. Note that a rule $Ca \rightarrow Cv$ is applicable if there is a C and an a in the preceding configuration. The result of applying this rule is the replacement of a by v. 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. We require that the chosen subset of rules to apply must be maximally parallel in the sense that no other applicable rule can be added to the subset. Configuration w is reachable if it appears in some execution sequence; w is halting if no rule is applicable on w.

In our definition, there is no target indication associated with the objects; i.e., we do not allow objects to exit the membrane into the environment.

We denote by $R(G)$ the Parikh map of all reachable configurations with respect to noncatalysts only. (Thus, if $a_1, ..., a_k$ are the noncatalysts, then $R(G) = \{(\#_{a_1}(x), ..., \#_{a_k}(x)) \mid x \text{ is a reachable configuration in } G\}$, where $\#_{a_i}(x)$ is the number of occurrences of noncatalyst a_i in x.) For convenience, when we talk about configurations, we sometimes do not include the catalysts. $R(G)$ is called the reachability set of G. $R_h(G)$ will denote the set of all halting reachable configurations. Let $\mathbf{N}$ be the set of all nonnegative integers and k be a positive integer. It is known that for any set $Q \subseteq \mathbf{N}^k$ that can be accepted by a Turing machine, we can construct a pure CS G such that $R_h(G) = Q$ [19, 20, 6]. In fact, [6] shows that three catalysts (even when each catalyst appears exactly once in the initial configuration) are already sufficient for universality. Thus, in general, a parallel 1-membrane pure CS can define a nonrecursive reachability set.

2.1. Sequential 1-Membrane Pure CS

In a sequential 1-membrane pure CS, each step of the computation consists of an application of a single nondeterministically chosen rule. We show below that sequential 1-membrane pure 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 vectors in $\mathbf{Z}^n$, where $\mathbf{Z}$ is the set of all integers (positive, negative, zero). 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 $x + v_1 + ... + v_i \geq 0\}$. The *halting reachability set* $R_h(G) = \{z \mid z \in R(G), z + v \not\geq 0$ 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 $G = \langle x, T, p_0 \rangle$, where p_0 is the starting state.

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.

Next, we recall the definition of a semilinear set. A set $S \subseteq \mathbf{N}^n$ is a *linear set* if there exist vectors $v_0, v_1, ..., v_k$ in $\mathbf{N}^n$ such that $S = \{v \mid v = v_0 + a_1 v_1 + ... + a_t v_k,\ a_i \in \mathbf{N}\}$. The vectors v_0 (referred to as the *constant vector*) and $v_1, v_2, ..., v_k$ (referred to as the *periods*) are called the *generators* of the linear set S. A set $S \subseteq \mathbf{N}^n$ is *semilinear* if it is a finite union of linear sets. The empty set is a trivial (semi)linear set, where the set of generators is empty. Every finite subset of $\mathbf{N}^n$ is semilinear – it is a finite union of linear sets whose generators are constant vectors. Clearly, semilinear sets are closed under union and projection. It is also known that semilinear sets are closed under intersection and complementation.

We summarize the following known results concerning VAS and VASS [21, 8, 1, 9, 13]:

Theorem 1 *Let G be an n-dimensional VASS. (1) We can effectively construct an $(n+3)$-dimensional VAS G' that simulates G. (2) If G is a 2-dimensional VASS G, 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 G, 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. They are equivalent to communication-free Petri nets, which are also equivalent to commutative context-free grammars [3, 10]. It is known that they have effectively computable semilinear

reachability sets [3].

Lemma 1 *Any sequential 1-membrane pure CS can be converted to one where there is only one catalyst.*

Proof. Let S be a sequential 1-membrane pure CS with noncatalysts $a_1, ..., a_k$. Suppose that the start configuration of S has t catalysts $C_1, ..., C_t$, where each catalyst can appear a multiple number of times in the start word. Since the operation of S is sequential (i.e., only one catalytic rule can be applied at any step), it is easy to see that S is equivalent to a 1-membrane pure CS with only one catalyst C in the start configuration, in which all rules of the form $C_i a \to C_i v$ are written as $Ca \to Cv$. □

The proof of the next theorem uses ideas in [11].

Theorem 2 *Every sequential 1-membrane pure CS can be simulated by a communication-free VAS G, and vice versa.*

Proof. Let S be a sequential 1-membrane pure CS with noncatalysts $a_1, ..., a_k$. We may assume from Lemma 1 that S has only one catalyst, C. Suppose Cw is the initial configuration.

First assume that each rule in S has the form $Ca \to Cv$, where a is not contained in v. If $Ca_m \to Ca_1^{i_1} ... a_{m-1}^{j_{m-1}} a_{m+1}^{j_{m+1}} ... a_k^{j_k}$ is a rule in S, then the following transition is in G: $(j_1, ..., j_{m-1}, -1, j_{m+1}, ..., j_k)$. The start vector of G is a k-dimensional vector x, where the components correspond to the multiplicities of the a_i's in w. It is easy to see that G simulates S.

Now we show how to convert a pure CS S to another system S' which satisfies the property above. The system S' has many catalysts and simulates S. Number the rules of S by $1, ..., s$. Pure CS S' will have catalysts $C, Q_1, ..., Q_s$ and noncatalysts $a_1, ..., a_k, d_1, ..., d_s$. Its initial configuration is $CwQ_1...Q_s$. The rules of S' are defined as follows:

Case 1: Suppose that $Ca_i \to Cv$ is a rule in S, and v does not contain a_i. Then this rule is in S'.

Case 2: Suppose that $Ca_i \to Ca_i^j v$ is rule number r in S, where $j \geq 1$ and v does not contain a_i. Then we have the following rules in S':

$Ca_i \to Cd_r^j v$ and $Q_r d_r \to Q_r a_i$.

We say that d_r is a_i-related since it is associated with a rule with a_i on the LHS (left hand side). It is easily verified that any reachable configuration in S' with the property that for each i, the number of occurrences of a_i + the number of occurrences of a_i-related d_r's (over all instruction number r) – call the sum m – uniquely corresponds to a reachable configuration in S where the number of occurrences of a_i is equal to m. (Note that reachable configurations in S are over $a_1, ..., a_k$.) Thus, S' simulates S.

Since in S' every rule $Xb \to Xv$ (where X is a catalyst, b is a noncatalyst, and v a string of noncatalysts) has the property that v does not contain b, S' can be converted to a communication-free VAS G.

Conversely, let G be a communication-free VAS. We construct a sequential 1-membrane pure CS S which has one catalyst C, noncatalysts $\#, a_1, ..., a_k$, and start-

ing configuration $C\#w$, where w corresponds to the starting vector of G. Suppose $(j_1, ..., j_{m-1}, j_m, j_{m+1}, ..., j_k)$ 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:

$Ca_m \rightarrow Ca_1^{j_1} ... a_{m-1}^{j_{m-1}} a_{m+1}^{j_{m+1}} ... a_k^{j_k}$.

Case 2: All the j_i's are nonnegative. Then the following rule is in S:

$C\# \rightarrow C\# a_1^{j_1} ... a_k^{j_k}$.

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

Corollary 1 *(1) If S is a sequential 1-membrane pure 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 1-membrane pure CS's is NP-complete.*

Proof. For part 1, let G be the communication-free VAS constructed in the first part of the proof above. Then $R(G) \subseteq \mathbf{N}^{k+s}$ is semilinear (since a communication-free VAS has a semilinear reachability set). Now the first k components of each vector in $R(G)$ correspond to $a_1, ..., a_k$, and the remaining s components correspond to $d_1, ..., d_s$. Clearly, from the proof above, $R(S) = proj_k(R(G) \cap (\mathbf{N}^k \times \{0\}^s))$, where $proj_k$ is the projection of the tuples on the first k coordinates. Since $\mathbf{N}^k \times \{0\}^s$ is semilinear and semilinear sets are closed under intersection and projection, it follows that $R(S)$ is semilinear.

For $R_h(S)$, without loss of generality (by simple relabeling), assume that $a_t, ..., a_k$ ($t \geq 1$) be all the noncatalysts for which there is some rule with a_i on the LHS, $t \leq i \leq k$. Clearly, a reachable configuration in $R(S)$ is halting if coordinates $t, ..., k$ are zero. Hence $R_h(S) = R(S) \cap (\mathbf{N}^{t-1} \times \{0\}^{k-t+1})$, which is semilinear.

Part 2 follows from the NP-completeness of the reachability problem for communication free Petri nets (which are equivalent to commutative context-free grammars) [10, 3]. □

3. Sequential 1-Membrane Communicating P Systems

Consider the model of a communicating P system (CPS) with only *one* membrane [19]. The rules are in the following forms:

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 be *out* or *here*. The first means that the symbol is exported into the environment, while the second means that the symbol remains in the membrane. (The designation *here* is usually omitted when understood.) The third rule brings in an object c from the environment to the membrane (note that there is only one membrane). 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). Hence, the decidability of reachability in VAS implies the decidability of reachability for CPS. The converse is not true as we show next.

Consider a 1-membrane CPS H. Let $\Sigma = \{a_1, ..., a_k\}$ be all the symbols in H. Consider a configuration

$$\alpha = a_1^{n_1} ... a_k^{n_k} \tag{1}$$

where each n_i (the multiplicity of a_i) is a nonnegative integer. The summation $\sum_i n_i$ is called the *weight* of the configuration α. Let $A \subseteq \Sigma$. We use $\alpha[A]^+$ to denote the set of configurations β where the multiplicity of each symbol a_i is equal to n_i (resp. is greater than n_i) when $a_i \notin A$ (resp. $a_i \in A$). That is, $\alpha[A]^+$ defines the following set

$$\{a_1^{l_1} ... a_k^{l_k} \mid \text{for each } 1 \leq i \leq k, \text{if } a_i \in A \text{ then } l_i > n_i \text{ else } l_i = n_i\}.$$

When the weight of α is W, the set $\alpha[A]^+$ is called an *upper-closed set with basic weight* W.

Suppose that H starts from a fixed configuration α_0 whose weight is W_0. We use $\mathcal{U}$ to denote all the sets u such that u is an upper-closed set with basic weight at most W_0. The following theorem gives a characterization of the set R_H of reachable configurations of H, under the sequential semantics of H.

Theorem 3 *R_H is a union of some elements in $\mathcal{U}$.*

Proof. Notice that, since the initial configuration α_0 is fixed, so is W_0. Therefore, there are only finitely many distinct upper-closed sets with basic weight at most W_0. Therefore, $\mathcal{U}$ is also finite.

Before we proceed further, a claim is needed. Let $u \in \mathcal{U}$; i.e., u is an upper-closed set with basic weight at most W_0. Let r be a rule in H. We use $r(u)$ (resp. $r^+(u)$) to denote the set of all possible resulting configurations by applying r, under the sequential semantics, once (resp. at least once) on some configuration in u. We use $r^*(u)$ to denote the union of u and $r^+(u)$. We claim that

(Claim 1) $r^*(u)$ is a union of some elements in $\mathcal{U}$.

Let u be $\alpha[A]^+$ for some A and α (with weight $\leq W_0$). We prove the claim by considering each possible form of rule r. Without loss of generality, we only show the claim for r taking the following form: $ab \rightarrow a_{here} b_{out} c_{come}$ where a, b, c are all distinct. All the other forms are either similar or simpler.

Case 1 $a \notin \alpha$ and $a \notin A$, or, $b \notin \alpha$ and $b \notin A$. In this case, r is not firable on any configuration in u. Hence, $r^*(u) = u$. The claim holds.

Case 2 $a \in \alpha$ or $a \in A$, and, $b \in \alpha$ but $b \notin A$. In this case, $r(u) = \alpha'[A]^+$ where α' is the result of replacing one b with c in multiset α. Observe that $r(u)$ is

in $\mathcal{U}$ since the weight of α' is also bounded by that of α. Following this line, one can show that $r^*(u)$ is the finite union of all these $\alpha'[A]^+$ in $\mathcal{U}$ where α' is the result of replacing 0 or more instances of b's with the same number of c's. The claim holds.

Case 3 $a \in \alpha$ or $a \in A$, and, $b \in \alpha$ and $b \in A$. In this case, each configuration in u has at least two instances of b's. The set $r^*(u)$ of configurations after firing r for at least once from a configuration in u is the union of all the sets in the following form:

- $\alpha'[A]^+$ where α' is the result of replacing 0 or more b's in α with the same number of c's;
- $\alpha'[A \cup \{c\}]^+$ where α' is the result of first dropping a b from α and then replacing 0 or more b's with the same number of c's.

In this way, the weight of each α' is still bounded by that of α. Hence the claim holds.

Case 4 $a \in \alpha$ or $a \in A$, and, $b \in A$ but $b \notin \alpha$. In this case, $r^*(u)$ is the union of u, and the following two sets:

- $\alpha[(A \cup \{c\}) - \{b\}]^+$,
- $\alpha[A \cup \{c\}]^+$.

Clearly, the claim holds.

This completes the proof of the claim.

Closely looking at the proof reveals that the set of elements in (Claim 1) can actually be computed from the given u and r; we use $U(r,u) \subseteq \mathcal{U}$ to denote the set which obviously satisfies that the union of all the elements in $U(r,u)$ equals $r^*(u)$. Based upon this, one can define an iterative procedure to compute R_H as follows, noticing that $\{\alpha_0\}$ itself is an element in $\mathcal{U}$.

Let $U := \{\{\alpha_0\}\}$;
Repeat
  $U' := U$;
  For each rule r in H and each $u \in U'$
    $U := U \cup U(r,u)$;
Until U equals U';
The desired R_H is then the union of all the elements in U.

The procedure always terminates since $U \subseteq \mathcal{U}$ is a finite set and is nondecreasing after each `Repeat` loop. Clearly, according to the sequential semantics of H, the obtained R_H from the procedure is exactly the set of all reachable configurations of H from α_0. □

It is straightforward to show,

Corollary 2 *The set of halting configurations of H is the union of some elements from $\mathcal{U}$.*

Notice that elements in $\mathcal{U}$ are all upper-closed sets. Indeed, every such set is semilinear. But the converse is not true. For instance, sets like $\{a^i b^j \mid i = j\}$ can not be the union of some elements in $\mathcal{U}$. Therefore, they can not be the halting configuration set of a CPS.

Conversely, for each upper-closed set u, one can construct a CPS H (on a larger alphabet than Σ) whose halting configuration set, after projection on Σ, is exactly u. To see this, let $u = \alpha[A]^+$ for some α and A. Let α_0 to be the multiset after adding into α one instance for each symbol in A. Now, we use $t^1, ..., t^q$ to denote all the instances in multiset α_0. Of course, if $q = 0$, then we are done since H can be constructed without any rules in it and starts from α_0. When $q \geq 1$, for each t^j, we associate two new symbols y^j and z^j. H, defined in below, works on alphabet $\Sigma \cup \{y^j, z^j \mid 1 \leq j \leq q\}$. Initially, H starts from configuration β that contains one instance of y^j and one instance of z^j for each $1 \leq j \leq q$, only. The rules in H are as follows:

for each $1 \leq j \leq q$, there is a rule $y^j z^j \rightarrow y^j_{out} z^j_{out} t^j_{come}$,

for each $a \in A$, there is a rule $y^1 z^1 \rightarrow y^1_{here} z^1_{here} a_{come}$.

It can be shown that H has the desired property. We are not able to show that when we are given two upper-closed sets u and u', there is a CPS H starting from one initial configuration whose halting configurations are exactly $u \cup u'$. But, if we allow a CPS to start with a finite set of initial configurations, one can easily establish the following theorem, using the above results:

Theorem 4 *(1) The halting configuration set of a sequential 1-membrane CPS H starting from a finite set of initial configurations is a finite union of upper-closed sets. (2) Any finite union of upper-closed sets is the halting configuration set of some 1-membrane CPS H starting from a finite set of initial configurations.*

From Theorem 3 and the fact mentioned earlier that $\{a^i b^j \mid i = j\}$ can not be the union of upper-closed sets, we have

Corollary 3 *Sequential 1-membrane CPS's can only generate a proper subclass of the semilinear sets.*

Moreover, since every sequential 1-membrane CPS can be simulated by a vector addition system (or, equivalently, by a Petri net), and vector addition systems are able to define semilinear sets, it follows that sequential 1-membrane CPS's are strictly weaker than vector addition systems.

4. Sequential 1-Membrane Extended CPS

We have seen in the previous section that 1-membrane CPS's operating sequentially define only semilinear sets. Interestingly, if we generalize the rules of a 1-membrane CPS slightly the extended system becomes equivalent to a VASS. Define an extended CPS (ECPS) by allowing rules of the form:

1. $a \rightarrow a_x$,
2. $ab \rightarrow a_x b_y$,
3. $ab \rightarrow a_x b_y c_{come}$,
4. $ab \rightarrow a_x b_y c_{come} d_{come}$

(i.e., by adding rules of type 4).

Let G be an n-dimensional VAS. Clearly, by adding new states, we may assume that all transitions in G have the form:

$p_i \to (p_j, +1_h)$,

$p_i \to (p_j, -1_h)$.

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 $p_i \to (p_j, +1_h)$ in G is simulated by the following rule in S:

$p_i c \to p_{i(out)} c_{here} p_{j(come)} a_{h(come)}$.

A transition of the form $p_i \to (p_j, -1_h)$ in G is simulated by the following rule in S:

$p_i a_h \to p_{i(out)} a_{h(out)} p_{j(come)}$

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 $< R_0, v_0 >$, 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. $R_0 \to (R_i, zero)$ 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 $a_h \to a_{hx}$, then the following are transitions:

   $R_i \to (R'_i, -1_h)$

   $R'_i \to (R_j, d_{hx})$ 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, $-1_h, 0, +1_h$ 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 $a_h a_r \to a_{hx} a_{ry}$, then the following are transitions:

   $R_i \to (R'_i, -1_h, -1_r)$

   $R'_i \to (R_j, d_{hx}, d_{ry})$ 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 $a_h a_r \to a_{hx} a_{ry} a_{s(come)}$, then the following are transitions:

   $R_i \to (R'_i, -1_h, -1_r)$

$R_i' \rightarrow (R_j, d_{hx}, d_{ry}, +1_s)$ 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 5 *Sequential 1-membrane ECPS and VASS are equivalent.*

We can generalize rules of an ECPS further as follows:

$a_{i_1} ... a_{i_h} \rightarrow a_{i_1 x_1} ... a_{i_1 x_h}$,

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

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 5, we can show ECPS+ is still equivalent to a VASS. Thus, we have:

Corollary 4 *The following systems are equivalent: Sequential 1-membrane ECPS, sequential 1-membrane ECPS+, and VASS.*

5. Sequential 1-Membrane Symport/Antiport Systems

Theorem 5 can easily be shown for sequential 1-membrane symport/antiport systems (SA) [12, 15]. These systems have rules of the form:

$(x, out; y, in)$ (**antiport rules**),

(x, out) or (x, in) (**symport rules**),

where x, y are strings of symbols. The *radius* of an antiport rule is $(|x|, |y|)$. For a symport rule, the radius is $|x|$.

Theorem 6 *(1) Every VASS can be simulated by a sequential 1-membrane SA all of whose rules are antiport with radius (1,2) or (2,1). (2) Every sequential 1-membrane SA can be simulated by a VASS.*

Proof. Let G be a VASS. We construct the sequential 1-membrane SA S simulating G as follows. Referring to the first part of the proof of Theorem 5, a transition of the form $p_i \rightarrow (p_j, +1_h)$ in G is simulated by the following antiport rule: $(p_i, out; p_j a_h, in)$. A transition of the form $p_i \rightarrow (p_j, -1_h)$ in G is simulated by the following antiport rule: $(p_i a_h, out; p_j, in)$. 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 S starts with the word $p_1 a_1^{i_1} ... a_n^{i_n} c$. Clearly, S simulates G.

The proof of the second part is similar to part 2 of the proof of Theorem 5. □

The next two results show that Part 1 of Theorem 6 is best possible.

Theorem 7 *Sequential 1-membrane SA's whose rules have radius 1, (1,1) or (2,1) (i.e., the rules can only be of the form: $(a, out), (a, in), (a, out; b, in), (ab, out; c, in)$) can only generate a proper subclass of the semilinear sets.*

Proof. Clearly the rules above can be simulated by a CPS with rules of the form: $a \to a_{out}$, $XY \to XYa_{come}$, $Xa \to Xa_{out}b_{come}$, $ab \to a_{out}b_{out}c_{come}$, respectively, where X, Y are two new symbols. Then, from Corollary 3, sequential 1-membrane CPS's can only generate a proper subclass of the semilinear sets. □

Theorem 8 *(1) Every semilinear set can be generated by a sequential 1-membrane SA whose rules have radius (1,2). (2) Every sequential 1-membrane SA whose rules have radius 1, (1,1), or (1,2) generates only a semilinear set.*

Proof. The first part follows from the observation that every semilinear set can be defined by a VASS with transitions of the form $p_i \to (p_j, +1_h)$, which translate into rules $(p_i, out; p_j a_h, in)$ of radius (1,2).

For the second part, it is clear that rules of the form $(a, out), (a, in), (a, out; b, in)$, (a, out, bc, in) can be simulated by catalytic rules $Ca \to C, CX \to CXa, Ca \to Cb, Ca \to Cbc$, respectively, where C is the (only) catalyst, and X is a new noncatalyst. Then, from Corollary 1, this catalytic system generates a semilinear set. □

From the preceding two theorems, we have:

Corollary 5 *A sequential 1-membrane SA whose rules have radius (1,2) is more powerful than a sequential 1-membrane SA whose rules have radius 1, (1,1), or (2,1).*

Part 2 of Theorem 8 can be generalized to the case when the sequential 1-membrane SA's have symport rules of radius > 1.

Corollary 6 *Every sequential 1-membrane SA whose rules are either symport rules (with any radius) or antiport rules with radius (1,1) or (1,2) generates only a semilinear set.*

Proof. Similar to the proof of Part 2 of Theorem 8, antiport rules $(a, out; b, in)$ and (a, out, bc, in) can be simulated by catalytic rules $Ca \to Cb$ and $Ca \to Cbc$, respectively. A symport rule of the form (x, in) can be simulated by catalytic rule $CX \to CXx$ where X is a new noncatalyst. A symport rule of the form (x, out) can be simulated by *degenerating* rule $x \to \epsilon$. We use S to denote the obtained system, S_1 (resp. S_2) to denote the result of dropping degenerating rules (resp. catalytic rules) from S, and $(S_1; S_2)$ to denote the sequential composition of S_1 and S_2. Observe that both S and $(S_1; S_2)$ generate the same reachability set. S_1 is a pure CS and hence generates a semilinear set R (Corollary 1). We use $T(u, v)$ to denote the fact that u can reach v in S_2. From the definition of degenerating rules, one can verify that T defines a semilinear set. The result follows, since S generates the set $\{v \mid \exists u, R(u) \wedge T(u, v)\}$, which is semilinear. □

For a positive integer k, consider sequential 1-membrane SA's whose rules are either symport rules with radius at most k or antiport rules with radius (1,1) or (2,1). For what values of k is it the case that such SA's can only generate semilinear sets? From Theorem 7, it is the case when $k = 1$. However, the question is open even for the case $k = 2$.

6. Conclusion

We showed in this paper that P systems that compute in sequential mode are strictly weaker than systems that operate in "maximal parallelism" for three classes of systems: 1-membrane purely catalytic systems, 1-membrane communicating P systems, and 1-membrane symport/antiport systems. In fact these systems can be characterized by vector addition systems or semilinear sets. An interesting topic for further research is to characterize the computing power of sequential multi-membrane systems. We hope to report on this in the future.

Acknowledgements

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

References

1. H. G. Baker, "Rabin's proof of the undecidability of the reachability set inclusion problem for vector addition systems," C.S.C. Memo 79, Project MAC, MIT, 1973.
2. E. Csuhaj-Varju, O. Ibarra, and G. Vaszil, "On the computational complexity of P automata," in *Proc. Tenth International Meeting on DNA Computing*, eds. C. Ferretti, G. Mauri, C. Zandron (2004), pp. 97–106.
3. J. Esparza, "Petri nets, commutative context-free grammars, and basic parallel processes," *Fundamenta Informatica* **31** (1997) 13–26.
4. R. Freund, "Sequential P-systems," Available at *http://psystems.disco.unimib.it*, 2000.
5. R. Freund, "Asynchronous P Systems," in *Pre-Proc. Fifth Workshop on Membrane Computing*, eds. G. Mauri, Gh. Paun, and C. Zandron (2004).
6. R. Freund, L. Kari, M. Oswald, and P. Sosik, "Computationally universal P systems without priorities: two catalysts are sufficient," *Theoretical Computer Science* **330** (2005) 251–266.
7. R. Freund and A. Paun, "Membrane systems with symport/antiport rules: universality results," in *Proc. Workshop on Membrane Computing*, eds. Gh. Paun and C. Zandron (Springer, Berlin, 2003), pp. 270–287.
8. M. H. Hack, "The equality problem for vector addition systems is undecidable," 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** (1979) 135–159.
10. D.T. Huynh, "Commutative grammars: The complexity of uniform word problems," *Information and Control* **57** (1983) 21–39.
11. O. Ibarra, Z. Dang, and O. Egecioglu, "Catalytic P systems, semilinear sets, and vector addition systems," *Theoretical Computer Science* **11** (2004) 167–181.
12. C. Martin-Vide, A. Paun, and Gh. Paun, "On the power of P systems with symport rules," *Journal of Universal Computer Science* **8** (2002) 317–331.
13. E. Mayr, "Persistence of vector replacement systems is decidable," *Acta Informatica* **15** (1981) 309–318.
14. M. Minsky, "Recursive unsolvability of Post's problem of Tag and other topics in the theory of Turing machines," *Ann. of Math.* **74** (1961) 437–455.

15. A. Paun and Gh. Paun, "The power of communication: P systems with symport/antiport," *New Generation Computing* **20** (2002) 295–306.

16. Gh. Paun, "Computing with membranes," *Journal of Computer and System Sciences* **61** (2000) 108–143.

17. Gh. Paun, *Membrane Computing: An Introduction* (Springer, Berlin, 2002).

18. Gh. Paun and G. Rozenberg, "A guide to membrane computing," *Theoretical Computer Science* **287** (2002) 73–100.

19. P. Sosik, "P systems versus register machines: two universality proofs," in *Proc. Workshop on Membrane Computing*, eds. Gh. Paun and C. Zandron (Springer, Berlin, 2003), pp. 371–382.

20. P. Sosik and R. Freund, "P systems without priorities are computationally universal," in *Proc. Workshop on Membrane Computing*, eds. Gh. Paun and C. Zandron (Springer, Berlin, 2003), pp. 400–409.

21. J. van Leeuwen, "A partial solution to the reachability problem for vector addition systems," in *Proc. ACM Symposium on Theory of Computing* (ACM Press, New York, 1974), pp. 303–309.