Bulletin of Networking, Computing, Systems, and Software – www.bncss.org, ISSN 2186-5140
Volume 10, Number 1, pages 10–16, January 2021

An asynchronous P system using branch and bound for maximum independent set

Kotaro Umetsu Akihiro Fujiwara
Graduate School of Computer Science and Systems Engineering
Kyushu Institute of Technology
Iizuka, Fukuoka, 820-8502, Japan

***Abstract*—Membrane computing is a computational model based on activity of cells. Using the membrane computing, a number of computationally hard problems have been solved in a polynomial number of steps using an exponential number of membranes. However, the number of membranes denotes the number of cells from practical point of view, and the reduction of the number of membranes must be considered for using the membrane computing in real world.**

In this paper, we propose an asynchronous P system with branch and bound for the maximum independent set. In addition, we evaluate validity of the proposed P system using computational simulations. The experimental results show the validity and efficiency of the proposed P system.

***Index Terms*—membrane computing, maximum independent set, branch and bound**

I. Introduction

Natural computing is one of the next-generation computing paradigms. The membrane computing, which has been introduced in [7] as a P system, is a representative computational model in the natural computing. Definition of the P system is based on a feature of cells, and a membrane and an object denote a computing cell and a storage of data, respectively. In addition, each object evolves according to evolution rules, which is associated with the membrane.

Since the exponential number of membranes can be created in the polynomial number of steps using a division rule, which is one of evolution rules on the P system, the computationally hard problem can be solved in a polynomial number of steps. Therefore, there are a number of P systems that solve NP problems [1]–[3], [5], [6], [8]–[14].

In addition, asynchronous parallelism, which assumes asynchronous application of evolution rules, has been considered for the P system. The asynchronous parallelism means that all objects may react to rules with different speeds on the P system. The asynchronous parallelism makes the P system a more realistic computational model because livings cells work independently according to environment.

A number of asynchronous P systems have also been proposed for NP problems [1], [3], [12], [13]. For example, an asynchronous P system for finding the maxmum independent set has been proposed in [13] . The P system for a graph with n vertices works in in $O(n^2 \cdot 2^n)$ sequential steps or $O(n^2)$ parallel steps using $O(n^2)$ kinds of objects.

All of the above P systems solves the computationally hard problems in the polynomial numbers of steps using exponential numbers of membranes. The number of membranes means the number of cells, and reduction of the number of membranes must be considered in case that the P system is implemented using living cells because cells cannot be created exponentially.

Recently, an asynchronous P system using branch and bound has been proposed in [4] for reducing the number of membranes. Branch and bound is a well-known optimization technique, and is used in the P system for omitting partial value assignments that cannot satisfy a given Boolean formula.

In this paper, we propose an asynchronous P system for solving the maxmum independent set problem with branch and bound. In the proposed P system, objects, which denote vertices, are labeled 0 or 1 one by one. In case that the object is labelede 1, adjacency of the labeled objects is checked. If both objects labeled 1 are adjacent, the partial assignment of vertices is discarded as a bounding operation. Since the number of membranes increases according to the number of assignments of vertices, the number can be reduced by omitting partial assignments that are not possible for the maxmum independent set.

We also show that the proposed P system finds the maxmum independent set with the same complexity in [13], that is, the P system solves the maxmum independent set problem for a graph with n vertices in $O(n^2 \cdot 2^n)$ sequential steps or $O(n^2)$ parallel steps using $O(n^2)$ kinds of objects.

In addition, we show the validity of the proposed P system through experimental simulations. In the simulation, various instances are executed on the previous P system [13] and the proposed P system. The results show validity and efficiency of the proposed P system.

The remainder of the paper is organized as follows. In Section 2, we describe the computational model for the membrane computing. In Section 3, we propose the P system with branch and bound for the maxmum independent set, and experimental results for the proposed P system are shown in Section 4. Section 5 concludes the paper.

II. Preliminaries

In the paper, we assume an asynchronous P system proposed in [12]. We briefly explain definition of the P system in the following.

The P system consists mainly of membranes and objects. A membrane, which is a computing cell in the P system, may

contain objects and other membranes, and each membrane is labeled with an integer. An object, which is a memory storage in the P system, stores data in the P system. According to evolution rules for the objects, objects may be evolved into another objects or pass through membranes. The objects may also divide or dissolve a membrane in which the object is stored. We assume that each object is a finite strings over a given set of alphabet.

For example of the membrane and the object, the following expression denotes a membrane structure that consists of two membranes and three objects.

$$[\ [\ \alpha\ ]_2\ [\ \beta\ \gamma\ ]_3\ ]_1$$

In the example, the membrane labeled 1 contains two membranes labeled 2 and 3, and the membrane labeled 2 and 3 contain sets of objects $\{\alpha\}$ and $\{\beta, \gamma\}$, respectively.

Computation of P systems is defined as evolution rules. Each evolution rule is a rewriting rule for membranes and objects. According to the applicable evolution rules, objects and membranes are transformed in parallel in each step of computation. The system stops computation if there is no applicable evolution rule for objects.

A number of types of evolution rules are assumed in the membrane computing. In this paper, we assume the following five rules in [12].

(1) Object evolution rule:

$$[\ \alpha\ ]_h \rightarrow [\ \beta\ ]_h$$

An object α is transformed into another object β.

(2) Send-in communication rule:

$$\alpha\ [\ ]_h \rightarrow [\ \beta\ ]_h$$

An object α is moved into the inner membrane labeled by h, and transformed into another object β.

(3) Send-out communication rule:

$$[\ \alpha\ ]_h \rightarrow [\ ]_h\ \beta$$

An object α is sent out from the membrane labeled by h, and transformed into another object β.

(4) Dissolution rule:

$$[\ \alpha\ ]_h \rightarrow \beta$$

The membrane that contains an object α is dissolved, and the object α is transformed into another object β. (Note that the outermost membrane cannot be dissolved.)

(5) Division rule:

$$[\ \alpha\ ]_h \rightarrow [\ \beta\ ]_h[\ \gamma\ ]_h$$

The membrane that contains an object α is divided into two membranes with the same label, and the object α is transformed into another objects, β and γ, in each of the divided membranes.

We summarize definition of the P system. The P system consists of the following four components.

- O: The set of objects used in the system.
- μ: A structure of membrane.
- ω_i: A set of objects initially contained in the membrane labeled i.
- R_i: A set of evolution rules for a membrane labeled i.

Using the above components, the P system is defined as follows.

$$\Pi = (O, \mu, \omega_1, \omega_2, \cdots, \omega_m . R_1, R_2, \cdots, R_m)$$

We now consider complexity of the P system. We assume that each of evolution rules can be executed in one step on the computational model, and the complexity of the P system is defined as the number of steps executed on the P system.

On the standard P system, all objects, for which there are applicable evolution rules, are transformed with maximally parallel manner. (In case that there are a number of applicable evolution rules for an object, one of the rules is applied non-deterministically.) Using the maximally parallel manner, we can consider an execution on the P system easily because the execution with the maximally parallel manner is simple. The complexity of the P system with maximally parallel manner is the best case complexity, and we call the number of steps executed with maximal parallel manner as *the number of parallel steps*.

In this paper, we also consider asynchronous parallelism [12] in the P system. Under the assumption of the asynchronous parallelism, any number of applicable evolution rules are applied in parallel. In other words, all objects, for which there are applicable evolution rules, can be transformed in parallel, or only one of the applicable evolution rules is applied in each step of computation. We call the number of of steps in the latter case as *the number of sequential steps*. The number of sequential steps denotes the complexity of the P system in the worst case.

III. An asynchronous P system with branch and bound for maximum independent set

In this section, we explain our proposed asynchronous P system for the maximum independent set. An input and an output of the problem on the P system is shown, and then, an outline and details of the P system are presented. Finally, the complexity of the proposed P system is discussed.

A. Input and output for the maximum independent set

The maximum independent set is a well known computationally hard problem. An input of the problem is an undirected graph $G = (V, E)$. An output of the problem is a subset of vertices such that no pair of vertices in the subset are adjacent. For example, we assume that a graph in Fig. 1 is an input graph for the maximum independent set. Then, a subset, $V' = \{v_1, v_4\}$ is one of the independent sets, and in case of the input graph, the subset is the maximum independent set.

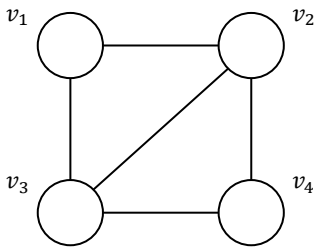

Fig. 1. An example of an input undirected graph

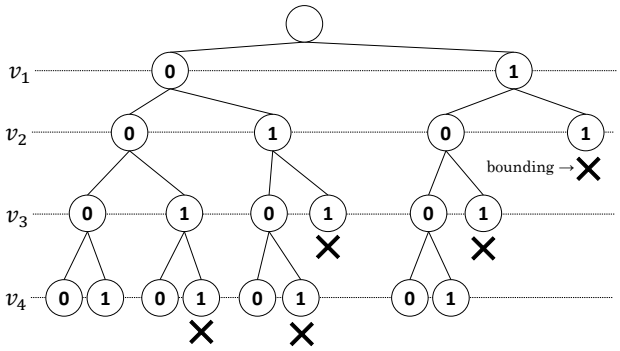

Fig. 2. An example of branch and bound for maximum independent set

For the proposed P system, the following set of objects, O_E, are given as an input graph.

$$O_E = \{\langle e_{i,j}, W\rangle \mid 1 \leq i \leq n, 1 \leq j \leq n, W \in \{T, F\}\}$$

In the above set of input objects, an edge (v_i, v_j) is denoted by an objects $\langle e_{i,j}, W\rangle$. A value W is set to T if an edge (v_i, v_j) is in the graph, otherwise, W is set to F.

For example, the following set of objects is given as an input of the P system, O_E, for a graph in Fig. 1.

$$\begin{aligned}
&\langle e_{1,1}, F\rangle, \langle e_{1,2}, T\rangle, \langle e_{1,3}, T\rangle, \langle e_{1,4}, F\rangle,\\
&\langle e_{2,1}, T\rangle, \langle e_{2,2}, F\rangle, \langle e_{2,3}, T\rangle, \langle e_{2,4}, T\rangle,\\
&\langle e_{3,1}, T\rangle, \langle e_{3,2}, T\rangle, \langle e_{3,3}, F\rangle, \langle e_{3,4}, T\rangle,\\
&\langle e_{4,1}, F\rangle, \langle e_{4,2}, T\rangle, \langle e_{4,3}, T\rangle, \langle e_{4,4}, F\rangle
\end{aligned}$$

We assume that a computation on the P system starts if the above O_E is given from the outside region into the skin membrane.

The output of the P system is the following set of objects, O_C, which denotes a subset of vertices.

$$O_C = \{\langle V_i, A\rangle \mid 1 \leq i \leq n, A \in \{0, 1\}\}$$

In the object $\langle V_i, A\rangle$, v_i. A is set to 1 If vertex v_i is in the subset, otherwise, A is set to 0.

For example, the following O_C is an output of the P system, which denotes maximum independent set for the graph in Fig. 1.

$$O_C \quad = \quad \{\langle V_1, 1\rangle, \langle V_2, 0\rangle, \langle V_3, 0\rangle, \langle V_4, 1\rangle\}$$

B. Branch and bound for maximum independent set

Branch and bound is a well-known computing paradigm for the optimization problem. On the existing P system for solving maximum independent set [13], all subset for vertices are created for an input graph with n vertices, and 2^n assignments are checked as to whether each assignment is valid. However, a partial assignment of vertices can be discarded if the valid assignment of vertices is determined.

Fig. 2 shows an example of a search tree for the above concept. Let the graph in Fig. 1 be an input graph. In the search tree, a selected subset of vertices is denoted by a path from a root node to a leaf node. In this case, a path starting from v_1, which is in V', to v_2, which is also in V', can be bounded because there is an edge between v_1 and v_2 in the input graph.

We now explain an overview of the asynchronous P system with branch and bound for finding the maximum independent set. An initial membrane structure for the computation is $[\ [\ ]_2\ ]_1$. We refer to the membrane labeled 1 as the *outer membrane* and refer to the membranes labeled 2 as the *inner membranes*.

The computation of the P system mainly consists of the following six steps.

- Step 1: Move modified copies of input objects into inner membranes.
- Step 2: Create subsets of vertices by dividing the inner membranes. Then, check whether adjacent vertices are in the selected subset with branch and bound. If adjacent vertices are in the subset, stop for the assignment for vertices.
- Step 3: Sent out all sizes of independent subset from all inner membranes, and compute the size of the maximum independent set in the outer membrane.
- Step 4: By using the computed size of the maximum independent set, create subsets of vertices again by dividing the inner membrane repeatedly. Then, check whether adjacent vertices are in the subset with branch and bound.
- Step 5: In each divided membrane, check whether each subset of vertices is the maximum independent set, and dissolve all inner membranes if the membrane contains a subset that is not the maximum independent set.
- Step 6: Dissolve one of the inner membranes that includes the maximum independent set, and send out the result from the outer membrane.

C. Details of the proposed P system

We explain details of each step of the computation for maximum independent set. In Step 1, modified copies of the input objects are moved into an inner membranes.

Since the P system considered in the present paper is asynchronous, we cannot move the input objects in parallel, and input objects are moved one by one applying the following sets of evolution rules. (In the following description, a set of evolution rules $R_{i,j}$ indicates that the set of rules is used for membrane i in Step j.)

(Evolution rules for the outer membrane)

$$\begin{aligned}
R_{1,1} \;=\; & \{\langle e_{1,1}, F\rangle[]_2 \to [\langle M_{2,1}\rangle\langle e_{1,1}, F\rangle]_2 \\
& \cup\{\langle M_{i,j}\rangle\langle e_{i,j}, W\rangle[]_2 \\
& \quad \to [\langle M_{i+1,j}\rangle\langle e_{i,j}, W\rangle]_2 \mid 1 \le i \le n, \\
& \qquad 1 \le j \le n, W \in \{F, T\}\} \\
& \cup\{\langle M_{n+1,j}\rangle \to \langle M_{1,j+1}\rangle \mid 1 \le j \le n, \\
& \cup\{\langle M_{1,n+1}\rangle[]_2 \to [\langle M_{1,n+2}\rangle]_2\}
\end{aligned}$$

(Evolution rules for inner membrane 2)

$$\begin{aligned}
R_{2,1} \;=\; & \{[\langle M_{i,j}\rangle]_2 \to []_2\langle M_{i,j}\rangle \\
& \quad \mid 1 \le i \le n+1, 1 \le j \le n+1\} \\
& \cup\{[\langle M_{1,n+2}\rangle]_2 \to [\langle S_1\rangle\langle enter, 0\rangle]_2[\langle D\rangle]_2\}
\end{aligned}$$

In the above evolution rules, object $\langle e_{1,1}, F\rangle$ starts the computation, and input objects, $\langle e_{i,j}, W\rangle$, are moved into the inner membrane by object $\langle M_{i,j}\rangle$. After all input objects, $\langle e_{i,j}, W\rangle$, are moved into the inner membrane, object $\langle M_{i,j}\rangle$ is changed into object $\langle M_{1,n+2}\rangle$. At the end of Step 1, objects $\langle D\rangle$, $\langle S_1\rangle$ and $\langle enter, 0\rangle$ are created by division rules. The object $\langle D\rangle$ is used for the computation of Step 5, and two objects, $\langle S_1\rangle$ and $\langle enter, 0\rangle$, trigger the computation of Step 2.

In Step 2, subsets of vertices are created by dividing the inner membranes. Then, the subset is checked whether adjacent vertices are in the subset with branch and bound. If adjacent vertices are in the subset, the assignment is stopped by bounding. Step 2 is executed by applying the following set of evolution rules. In the evolution rules, $\langle v_i, 1\rangle$ denotes that v_i is contained in the subset of vertices.

(Evolution rules for inner membrane)

$$\begin{aligned}
R_{2,2} \;=\; & \{[\langle S_i\rangle\langle enter, h\rangle]_2 \\
& \quad \to [\langle S_{i+1}\rangle\langle v_i, 0\rangle\langle enter, h\rangle]_2 \\
& \qquad [\langle L_{i,1}\rangle\langle v_i, 1\rangle\langle enter, h+1\rangle]_2\} \\
& \qquad \mid 1 \le i \le n, 1 \le h \le n-1\} \\
& \cup\{\langle L_{i,j}\rangle\langle v_j, 0\rangle\langle e_{i,j}, W\rangle \to \langle L_{i,j+1}\rangle\langle v_j, 0\rangle\langle e_{i,j}, W\rangle \\
& \quad \mid 2 \le i \le n, 1 \le j \le i-1, W \in \{F, T\}\} \\
& \cup\{\langle L_{i,j}\rangle\langle v_j, 1\rangle\langle e_{i,j}, F\rangle \\
& \quad \to \langle L_{i,j+1}\rangle\langle v_j, 1\rangle\langle e_{i,j}, F\rangle \mid 2 \le i \le n, 1 \le j \le i-1\} \\
& \cup\{\langle L_{i,j}\rangle\langle v_j, 1\rangle\langle e_{i,j}, T\rangle \\
& \quad \to \langle Failed, n-i\rangle\langle v_j, 1\rangle\langle e_{i,j}, T\rangle\langle d_{i,1}, down\rangle \\
& \quad \langle d_{i+1,1}, up\rangle \mid 2 \le i \le n, 1 \le j \le i-1\} \\
& \cup\{\langle L_{i,i}\rangle \to \langle S_{i+1}\rangle \\
& \quad \mid 1 \le i \le n\} \\
& \cup\{\langle S_{n+1}\rangle\langle enter, h\rangle \to \langle Succeed, 0, h\rangle\langle d_{n,1}, down\rangle \\
& \quad \mid 0 \le h \le n\}
\end{aligned}$$

In the above evolution rules, vertex v_i is set into subset using the first set of rules. The vertices of v_i $(1 \le i \le n)$ are checked for v_i using object $\langle L_{i,j}\rangle$ if v_i and v_j are adjacent or not. Then, the object $\langle enter, h\rangle$ denotes the number of vertices in subset for the membrane.

On the other hand, for the case in which v_i and v_j are adjacent, and the vertices are in the subset at the same time, objects $\langle Failed, n-i\rangle$, $\langle d_{i,1}, down\rangle$ and $\langle d_{i+1,1}, up\rangle$, which denote a failure of assignment, are created. The object $\langle Failed, n-i\rangle$ denotes that $n-i$ vertices are remained in the case of the failure of the assignment, and $\langle d_{i,1}, down\rangle$ $\langle d_{i+1,1}, up\rangle$ are objects used in Step 3 for deleting partial assignment in the membrane. In addition, these objects trigger the computation of Step 3.

In Step 3, in each divided membrane, the number of vertices is sent out to the outer membrane in the case that the subset is an independent set. Then, the size of the maximum independent set is computed in the outer membrane.

Step 3 is executed by applying the following set of evolution rules.

(Evolution rules for the outer membrane)

$$\begin{aligned}
R_{1,3} \;=\; & \{\langle Failed, k\rangle\langle Failed, k\rangle \to \langle Failed, k+1\rangle \\
& \quad \mid 0 \le k \le n-2\} \\
& \cup\{\langle Failed, k\rangle\langle Succeed, k, h\rangle \\
& \quad \to \langle Succeed, k+1, h\rangle \\
& \quad \mid 0 \le k \le n-1, 0 \le h \le n\} \\
& \cup\{\langle Succeed, k, a\rangle\langle Succeed, k, b\rangle \\
& \quad \to \langle Succeed, k+1, a\rangle \\
& \quad \mid 0 \le k \le n-1, 1 \le a \le n, 0 \le b \le a\} \\
& \cup\{\langle Succeed, n, h\rangle \to \langle search, h\rangle \\
& \quad \mid 1 \le h \le n\}
\end{aligned}$$

(Evolution rules for inner membrane)

$$\begin{aligned}
R_{2,3} \;=\; & \{\langle d_{i,j}, down\rangle\langle e_{i,j}, W\rangle \to \langle d_{i,j+1}, down\rangle \\
& \quad \mid 1 \le i \le n, 1 \le j \le n-1, W \in \{F, T\}\} \\
& \cup\{\langle d_{i,n}, down\rangle\langle e_{i,n}, W\rangle\langle v_i, a\rangle \to \langle d_{i-1,1}, down\rangle \\
& \quad \mid 1 \le i \le n, W \in \{F, T\}, a \in \{0, 1\}\}
\end{aligned}$$

$$\cup\{\langle d_{i,j}, up\rangle\langle e_{i,j}, W\rangle \to \langle d_{i,j+1}, up\rangle \mid 1 \le i \le n, 1 \le j \le n-1, W \in \{F,T\}\}$$
$$\cup\{\langle d_{i,n}, up\rangle\langle e_{i,n}, W\rangle \to \langle d_{i+1,1}, up\rangle \mid 1 \le i \le n, W \in \{F,T\}\}$$
$$\cup\{[\langle Failed, i\rangle\langle d_{0,1}, down\rangle\langle d_{n+1,1}, up\rangle\langle enter, k\rangle]_2 \to \langle Failed, i\rangle \mid 0 \le i \le n-1, 1 \le k \le n\}$$
$$\cup\{[\langle Succeed, 0, h\rangle\langle d_{0,1}, down\rangle]_2 \to \langle Succeed, 0, h\rangle \mid 0 \le h \le n\}$$

In Step 3, $R_{2,3}$ is applied in each divided membrane. In each divided membrane, the objects $\langle d_{i,j}, down\rangle$ and $\langle d_{i,j}, up\rangle$ erase all objects in each inner membranes except for $\langle Succeed, 0, h\rangle$ and $\langle Failed, i\rangle$. Then, the objects, $\langle Succeed, 0, h\rangle$ and $\langle Failed, i\rangle$, are sent out to the outer membrane by dissolving the membrane.

The object $\langle Succeed, k, h\rangle$ denotes that there are 2^k successful assignments of vertices with h vertices, and the object $\langle Failed, i\rangle$ denotes that there are 2^i failure assignments of vertices. For counting the number of vertices in the subset, object $\langle search, h\rangle$ is created. The $\langle search, h\rangle$ denotes that the check for the membrane is finished and the maximum size of the subset in each membrane.

In Step 4, by using the computed size of the maximum independent set, subsets of vertices are created again by dividing the inner membrane repeatedly. Then, check whether adjacent vertices are in the subset with branch and bound. Step 4 is executed by applying the following sets of evolution rules.

(Evolution rules for the outer membrane)

$$R_{1,4} = \{\langle search, h\rangle[]_2 \to [\langle search, h\rangle]_2 \mid 1 \le h \le\}$$

(Evolution rules for inner membrane)

$$\begin{aligned}
R_{2,4} = {} & \{\langle search, h\rangle\langle D\rangle \to \langle 2S_1\rangle\langle search, h\rangle\langle 2enter, 0\rangle\} \\
& \quad \mid 1 \le h \le n\} \\
& \cup\{[\langle 2S_i\rangle\langle 2enter, h\rangle]_2 \\
& \to [\langle Lcheck, n-1\rangle\langle 2v_i, 0\rangle\langle 2enter, h\rangle]_2 \\
& \quad [\langle 2L_{i,1}\rangle\langle 2v_i, 1\rangle\langle 2enter, h+1\rangle]_2\} \\
& \quad \mid 1 \le i \le n, 0 \le h \le n-1\} \\
& \cup\{\langle 2L_{i,i}\rangle \to \langle 2S_{i+1}\rangle\} \mid 1 \le i \le n\} \\
& \cup\{\langle search, h\rangle\langle 2S_{n+1}\rangle\langle 2enter, h\rangle \\
& \quad \to \langle 2Succeed, 0, h\rangle\langle 2d_{1,1}, up\rangle\} \mid 0 \le h \le n\} \\
& \cup\{\langle 2L_{i,j}\rangle\langle 2v_j, 0\rangle\langle e_{i,j}, W\rangle \to \langle 2L_{i,j+1}\rangle\langle v_j, 0\rangle\langle e_{i,j}, W\rangle \\
& \quad \mid 2 \le i \le n, 1 \le j \le i-1, W \in \{F,T\}\} \\
& \cup\{\langle 2L_{i,j}\rangle\langle 2v_j, 1\rangle\langle e_{i,j}, F\rangle \to \langle 2L_{i,j+1}\rangle\langle 2v_j, 1\rangle\langle e_{i,j}, F\rangle \\
& \quad \mid 2 \le i \le n, 1 \le j \le i-1\} \\
& \cup\{\langle 2L_{i,j}\rangle\langle 2v_j, 1\rangle\langle e_{i,j}, T\rangle \\
& \quad \to \langle 2Failed, n-i\rangle\langle 2v_j, 1\rangle\langle e_{i,j}, T\rangle \\
& \quad \quad \langle 2d_{i,1}, down\rangle\langle 2d_{i+1,1}, up\rangle \\
& \quad \mid 2 \le i \le n, 1 \le j \le i-1\} \\
& \cup\{\langle search, a\rangle\langle 2enter, h\rangle\langle Lcheck, a-h-1\rangle \\
& \quad \to \langle search, a\rangle\langle 2enter, h\rangle\langle 2Failed, a-h-1\rangle \\
& \quad \langle 2d_{n-a+h+1,1}, down\rangle\langle 2d_{n-a+h+2,1}, up\rangle \\
& \quad \mid 1 \le a \le n, 0 \le h \le a-1\} \\
& \cup\{\langle search, a\rangle\langle 2enter, h\rangle\langle Lcheck, k\rangle \\
& \quad \to \langle search, a\rangle\langle 2enter, h\rangle\langle 2S_{n-k+1}\rangle \\
& \quad \mid 1 \le a \le n-1, 0 \le h \le a, a-h \le k \le n-h\}
\end{aligned}$$

In the above evolution rule for the outer membrane, object $\langle search, h\rangle$, which is created in Step 3, is moved into the inner membrane. On the other hand, in the above evolution rules for the inner membrane, objects $\langle 2S_1\rangle$ and $\langle 2enter, 0\rangle$ are created, and the objects triggers a division rule. In addition, if the number of vertices is not the meximumn h of $\langle search, h\rangle$ using $\langle Lcheck, i\rangle$, which denotes the number of vertices not yet assigned, the divided membrane is bounded. Then, an assignment of vertices are created similar to Step 2, and objects $\langle 2d_{i,j}, down\rangle$ and $\langle 2d_{i,j}, up\rangle$, which triggers the computation of Step 5, are created in each divided membrane.

In Step 5, all inner membrane are deleted except for the membrane that contains the maximum independent set. Step 5 is executed by applying the following set of evolution rules.

(Evolution rules for the outer membrane)

$$\begin{aligned}
R_{1,5} = {} & \{\langle 2Failed, k\rangle\langle 2Failed, k\rangle \to \langle 2Failed, k+1\rangle \\
& \quad \mid 0 \le k \le n-2\} \\
& \cup\{\langle 2Failed, k\rangle\langle 2Succeed, k, h\rangle \\
& \quad \to \langle 2Succeed, k+1, h\rangle \\
& \quad \mid 0 \le k \le n-1, 1 \le h \le n\} \\
& \cup\{\langle 2Succeed, k, a\rangle\langle 2Succeed, k, b\rangle \\
& \quad \to \langle 2Succeed, k+1, a\rangle \\
& \quad \mid 0 \le k \le n-1, 1 \le a \le n, 0 \le b \le a\} \\
& \cup\{\langle 2Succeed, n, h\rangle \to \langle output\rangle \mid 1 \le h \le n\}
\end{aligned}$$

(Evolution rules for inner membrane)

$$\begin{aligned}
R_{2,5} = {} & \{\langle 2d_{i,j}, down\rangle\langle e_{i,j}, W\rangle \to \langle 2d_{i,j+1}, down\rangle \\
& \quad \mid 1 \le i \le n, 1 \le j \le n-1, W \in \{F,T\}\} \\
& \cup\{\langle 2d_{i,n}, down\rangle\langle e_{i,n}, W\rangle\langle 2v_i, a\rangle \\
& \quad \to \langle 2d_{i-1,1}, down\rangle \\
& \quad \mid 1 \le i \le n, W \in \{F,T\}, a \in \{0,1\}\} \\
& \cup\{\langle 2d_{i,j}, up\rangle\langle e_{i,j}, W\rangle \to \langle 2d_{i,j+1}, up\rangle \\
& \quad \mid 1 \le i \le n, 1 \le j \le n-1, W \in \{F,T\}\} \\
& \cup\{\langle 2d_{i,n}, up\rangle\langle e_{i,n}, W\rangle \to \langle 2d_{i+1,1}, up\rangle \\
& \quad \mid 1 \le i \le n, W \in \{F,T\}\} \\
& \cup\{[\langle 2Failed, i\rangle\langle 2d_{0,1}, down\rangle\langle 2d_{n+1,1}, up\rangle\langle 2enter, k\rangle]_2 \\
& \quad \to \langle 2Failed, i\rangle \mid 0 \le i \le n-1, 0 \le k \le n\} \\
& \cup\{[\langle 2Succeed, 0, h\rangle\langle 2d_{n+1,1}, up\rangle]_2 \\
& \quad \to \langle 2Succeed, 0, h\rangle \mid 1 \le h \le n\}
\end{aligned}$$

In Step 5, $R_{2,5}$ is applied in each divided membrane. In each failed membrane, the objects $\langle 2d_{i,j}, down\rangle$ and $\langle 2d_{i,j}, up\rangle$ delete all objects in each inner membranes except for $\langle 2Succeed, 0, h\rangle$ and $\langle 2Failed_i\rangle$. In other words, the membrane that contains the maximum independent sets is not deleted. Then, the objects, $\langle 2Succeed, 0, h\rangle$ and $\langle 2Failed_i\rangle$, are sent out to the outer membrane with dissolving the membrane.

For counting the number of vertices in the subset, object $\langle 2Succeed, n, h\rangle$ is created. The $\langle 2Succeed, n, h\rangle$ denotes that the check for the membrane is finished and the maximum size of the subset is h. Then, the object $\langle output\rangle$, which is created from $\langle 2Succeed, n, h\rangle$, is sent out to outer membrane. The $\langle output\rangle$ triggers the computation of Step 6.

In Step 6, one of the inner membranes, which includes the maximum independent set, is dissolved, and the result is sent out from the outer membrane. Step 6 is executed by applying the following set of evolution rules.

(Evolution rules for the outer membrane)

$$\begin{aligned} R_{1,6} &= \{\langle output\rangle[]_2 \to \langle output\rangle \\ &\cup\{\langle chain, i\rangle \to \langle chain, i+1\rangle\langle output, i+1\rangle \\ &\quad \mid 1 \le i \le n-2\} \\ &\cup\{\langle chain, n-1\rangle \to \langle output, n\rangle \\ &\cup\{[\langle output, i\rangle\langle 2v_i, a\rangle]_0 \to []_0\langle V_i, a\rangle \\ &\quad \mid 1 \le i \le n, a \in \{0,1\}\} \end{aligned}$$

(Evolution rules for the inner membrane)

$$R_{2,6} = \{[\langle output\rangle]_2 \to \langle chain, 1\rangle\langle output, 1\rangle$$

At the beginning of Step 6, object $\langle output\rangle$ is moved into one of the inner membranes, which is selected non-deterministically. Then, the membrane is dissolved by the object, and objects $\langle chain, 1\rangle$ and $\langle output, 1\rangle$ are created in the inner membrane. Next, a set of output objects $\{\langle V_i, W\rangle | \ W \in \{0,1\}\}$ is sent out from the outer membrane to the outside region by auxiliary objects $\langle chain, i\rangle$ and $\langle output, i\rangle$.

We now summarize the asynchronous P system Π_{MIS} for finding the maximum independent set as follows:

$$\Pi_{\mathrm{MIS}} = (O, \mu, \omega_1, \omega_2, R_1, R_2)$$

$$\begin{aligned} O &= \{\langle e_{i,j}, W\rangle \mid 1 \le i \le n, 1 \le j \le n, W \in \{F, T\}\} \\ &\cup\{\langle M_{i,j}\rangle \mid 1 \le i \le n, 1 \le j \le n\} \\ &\cup\{\langle S_i\rangle \mid 1 \le i \le n+1\} \\ &\cup\{\langle enter, h\rangle \mid 1 \le h \le n\} \\ &\cup\{\langle v_i, W\rangle \mid 1 \le i \le n, W \in \{0,1\}\} \\ &\cup\{\langle Failed, k\rangle \mid 0 \le k \le n-1\} \\ &\cup\{\langle Succeed, k, h\rangle \mid 0 \le k \le n, 1 \le h \le n\} \\ &\cup\{\langle d_{i,j}, down\rangle \mid 0 \le i \le n, 1 \le j \le n+1\} \\ &\cup\{\langle d_{i,j}, up\rangle \mid 1 \le i \le n+1, 1 \le j \le n+1\} \\ &\cup\{\langle search, h\rangle \mid 1 \le h \le n\} \\ &\cup\{\langle D\rangle\} \\ &\cup\{\langle 2S_i\rangle \mid 1 \le i \le n+1\} \\ &\cup\{\langle 2enter, h\rangle \mid 1 \le h \le n\} \\ &\cup\{\langle 2v_i, W\rangle \mid 1 \le i \le n, W \in \{0,1\}\} \\ &\cup\{\langle 2Failed, k\rangle \mid 0 \le k \le n-1\} \\ &\cup\{\langle 2Succeed, k, h\rangle \mid 0 \le k \le n, 1 \le h \le n\} \\ &\cup\{\langle 2d_{i,j}, down\rangle \mid 0 \le i \le n, 1 \le j \le n+1\} \\ &\cup\{\langle 2d_{i,j}, up\rangle \mid 1 \le i \le n+1, 1 \le j \le n+1\} \\ &\cup\{\langle Lcheck, i\rangle \mid 1 \le i \le n-1\} \\ &\cup\{\langle output\rangle\} \\ &\cup\{\langle chain, i\rangle \mid 1 \le i \le n-1\} \\ &\cup\{\langle output, i\rangle \mid 1 \le i \le n\} \\ &\cup\{\langle V_i, W\rangle \mid 1 \le i \le n, W \in \{0,1\}\} \\ \mu &= [\ [\ ]_2\ ]_1 \\ \omega_1 &= \omega_2 = \phi \\ R_1 &= R_{1,1} \cup R_{1,2} \cup \cdots \cup R_{1,6} \\ R_2 &= R_{2,1} \cup R_{2,2} \cup \cdots \cup R_{2,6} \end{aligned}$$

D. Complexity of the P system

The complexity of the proposed P system Π_{MIS} is considered as follows. In Step 1, $O(n^2)$ objects are moved from the outer membrane into the inner membrane sequentially. Then, Step 1 works in $O(n^2)$ parallel or sequential steps. The sizes of the evolution rules and objects are $O(n^2)$ and $O(n^2)$, respectively.

In Step 2, $O(2^n)$ membranes are created, and Step 2 works in $O(n^2)$ parallel steps and $O(2^n)$ sequential steps. The sizes of the evolution rules and objects are $O(n^2)$ and $O(n^2)$, respectively.

Step 3 is executed in each divided membrane. Therefore, Step 3 works in $O(n^2)$ parallel steps and $O(n^2)$ sequential steps. The size of the evolution rules is $O(n^3)$.

In Step 4 and Step 5, the procedure is almost the same as that of Step 2 and Step 3. Therefore, the numbers of sequential and parallel steps of Step 4 are $O(2^n)$ and $O(n^2)$, respectively, and the numbers of sequential and parallel steps of Step 5 are $O(n^2)$ and $O(n^2)$, respectively.

In the final step, the computation is executed in the outer membrane, and Step 6 works in $O(n)$ parallel or sequential steps. The size of the evolution rules is $O(n)$.

From the above discussion, the following theorem is obtained for the proposed asynchronous P system Π_{MIS}.

Theorem 1: The asynchronous P system Π_{MIS}, which solves the maximum independent set for a graph with n vertices, works in $O(2^n)$ sequential steps or $O(n^2)$ parallel steps by using $O(n^2)$ types of objects and evolution rules of size $O(n^3)$. □

IV. Experimental simulations

We compare the numbers of membranes used on an existing P system [13] and the number of membranes used on the

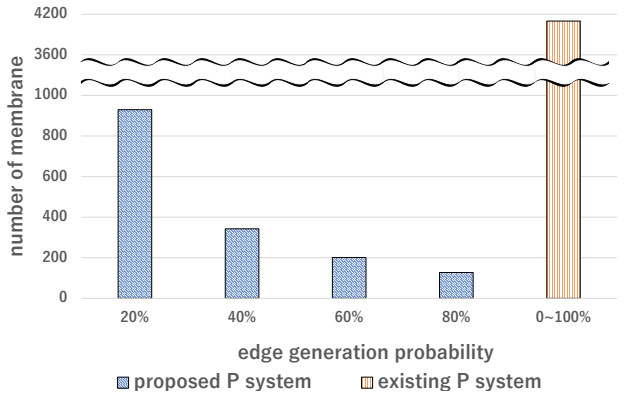

Fig. 3. Experimental results

proposed P system membranes for the maximum independent set. The simulations are executed on our custom simulator, which is programmed in Python, for asynchronous P systems.

Since the simulator executes P systems with asynchronous parallelism, all of the evolution rules are applied in a fully asynchronous manner. In other words, any number of applicable evolution rules is applied in each step of execution in the simulator. Therefore, the applied evolution rules differ between executions on the simulator, and the output of the simulation may differ for the same input. We first implement the proposed P system for the maximum independent set on the simulator, and execute the proposed P systems for various inputs on the simulator. In all of the simulations, valid results are obtained for any input.

Next, we compare the number of membranes used on the existing P system and the proposed P system for the maximum independent set. For example, in case of twelve vertices, ten random graphs are created so that each edge is connected with probability $p(p \in \{\frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}\})$, and the input graphs are given to the existing and proposed P systems.

Fig. 3 shows an example of the results that denote average values of the number of membranes for $n = 12$. Since the simulation for the existing P system for $n = 12$ did not finish in a valid execution time, the number of the membranes is calculated theoretically for this case. The number of membranes of the existing P system needs more than $4,000$ for twelve vertices graph. In this case, the number of membranes on the proposed P system is 3 percent of the number of membranes on the existing P system for $p = \frac{4}{5}$.

V. Conclusions

In this paper, we proposed the asynchronous P system for the maximum independent set. The P system with branch and bound reduces the number of membranes by discarding partial assignments of vertices in which the neighboring vertices are in the same subset.

The proposed P system is fully asynchronous and works in a polynomial number of steps in a maximally parallel manner, and also works sequentially. Although asymptotic complexity of the P system is the same as the known P system, the proposed P system reduces the number of membranes by discarding a partial assignment of vertices in case that the assignment is invalid.

We showed that the proposed P system outputs valid results and also showed that the number of membranes in the proposed P system is effectively less than the number of membranes in the existing P system for the maximum independent set.

In our future research, we intend to consider a reduction in the number of objects and the size of evolution rules used in the proposed P system. We also intend to consider asynchronous P systems with branch and bound for other computationally hard problems.

VI. Acknowledgments

The work was partially supported by JSPS KAKENHI, Grant-in-Aid for Scientific Research (C), 20K11681.

References

[1] R. Freund. Asynchronous P systems and P systems working in the sequential mode. In *International workshop on Membrane Computing*, pages 36–62, 2005.

[2] M. A. Gutiérrez-Naranjo, M. J. Pérez-Jiménez, and F. J. Romero-Campero. A uniform solution to SAT using membrane creation. *Theoretical Computer Science*, 371(1-2):54–61, 2007.

[3] J. Imatomi and A. Fujiwara. An asynchronous P system for MAX-SAT. In *8th International Workshop on Parallel and Distributed Algorithms and Applications*, pages 572–578, 2016.

[4] Y. Jimen and A. Fujiwara. Asynchronous P systems for solving the satisfiability problem. *International Journal of Networking and Computing*, 8(2):141–152, 2018.

[5] A. Riscos-Nnez M. J. Prez-Jimnez. A linear-time solution to the knapsack problem using P systems with active membranes. *Proc. International Workshop on Membrane Computing*, pages 250–268, 2003.

[6] L. Q. Pan and A. Alhazov. Solving HPP and SAT by P systems with active membranes and separation rules. *Acta Informatica*, 43(2):131–145, 2006.

[7] G. Păun. Computing with membranes. *Journal of Computer and System Sciences*, 61(1):108–143, 2000.

[8] G. Păun. P systems with active membranes: Attacking NP-complete problems. *Journal of Automata, Languages and Combinatorics*, 6(1):75–90, 2001.

[9] M. J. Pérez-Jiménez and A. Riscos-Núñez. Solving the subset-sum problem by P systems with active membranes. *New Generation Computing*, 23(4):339–356, 2005.

[10] M. J. Pérez-Jiménez and F.J. Romero-Campero. Solving the BIN PACKING problem by recognizer P systems with active membranes. In *The Second Brainstorming Week on Membrane Computing*, pages 414–430, 2004.

[11] M. J. Pérez-Jiménez, A. Romero-Jiménez, and F. Sancho-Caparrini. A polynomial complexity class in P systems using membrane division. *Journal of Automata, Languages and Combinatorics*, 11(4):423–434, 2003.

[12] H. Tagawa and A. Fujiwara. Solving SAT and Hamiltonian cycle problem using asynchronous p systems. *IEICE Transactions on Information and Systems (Special section on Foundations of Computer Science)*, E95-D(3), 2012.

[13] K. Tanaka and A. Fujiwara. Asynchronous P systems for hard graph problems. *International Journal of Networking and Computing*, 4(1):2–22, 2014.

[14] C. Zandron, C. Ferretti, and G. Mauri. Solving NP-complete problems using P systems with active membranes. In *Unconventional Models of Computation*, pages 289–301, 2000.