On Communication Complexity of Some Hard Problems in ECPe Systems with Priority

Sherlyne L. Francia
slfrancia@up.edu.ph

Denise Alyssa A. Francisco
dafrancisco@up.edu.ph

Richelle Ann B. Juayong
rbjuayong@up.edu.ph

Henry N. Adorna
hnadorna@dcs.upd.edu.ph

Algorithms and Complexity Laboratory
University of the Philippines Diliman
Quezon City, Philippines

ABSTRACT

In this paper, the Vertex Cover Problem and 3-Satisfiability Problem are non-confluently decided using Evolution-Communication P systems with Energy (ECPe) that impose priority on either evolution or communication. Additionally, the communication resources, i.e. number of communication steps, communication rules and energy objects, used in each system is analyzed. It is then shown through comparison that the ECPe systems that put priority on evolution utilize the greatest amount of resources. Those that put priority on communication, however, not only use the least amount of resources but also employ the least amount of membranes.

1. INTRODUCTION

Nearly two decades has passed since Georghe Păun [10] devised the P system computational model inspired by the biological membrane structure of cells. Since then, a considerable number of researches and studies have been and are being done. There are studies, such as in [9], [11], [15], exploring different variants of P systems and their characteristics. There are also diverse papers on solving NP-complete problems using various types of P systems (as in [13], [7], [14]).

Despite the numerous investigations on P systems, the issue of communication complexity of P systems is rarely considered. Thus, the authors in [1] focused their attention on it and introduced dynamical communication complexity measures and the Evolution Communication P system with energy (ECPe) which utilizes energy objects as a measure of communication cost. One study, as presented in [5], uses these ECPe systems in solving some NP-Complete problems, namely the Vertex Cover Problem(VCP), Independent Set Problem(ISP), and 3-Satisfiability Problem(3SAT). The proponents in [5] then compute for the communication complexity in solving the said problems with the use of the dynamical communication complexity measures suggested in [1].

There are three modes of ECP systems with energy (ECPe systems) considered in [1]: one has priority on communication, another has priority on evolution, and the other does not impose priority on either communication or evolution. The authors in [5] only explored one, which is the ECPe system without priority and this inspired this study. After reading [5], we became interested in finding out if using a different mode for a specific ECPe system will affect the utilization of communication resources of the system. As it turns out, the ECPe systems constructed in [5] can not be used directly if the priority is in evolution and can be further optimized if the priority is in communication. Modifying the ECPe system to suit the constraints led to the result showing that when the priority is in communication, the least amount of communication resources is used and when the priority is on evolution, the most amount is utilized.

The solutions presented in this paper and in [5], use the concept of non-confluent recognizer P systems. A P system is said to be confluent if all of its computations produce the same result, either acceptance(**yes**) or rejection(**no**), given an input. If this is not the case, then the P system is said to be non-confluent and the result is acceptance if and only if there exists an accepting computation of the P system. The notion of confluent and non-confluent P systems is further discussed in Section 3.2.

As an outline, in Section 2, we give the definition of a graph, VCP, conjunctive normal form(CNF) and 3SAT. On the third section, we discuss what ECPe systems are and the dynamic communication measures that we will use in analyzing the hardness of a solution. We will also give some other definitions that we will use in solving problems in ECPe systems. On the fourth and fifth sections of this paper, we will discuss the solutions to VCP and 3SAT, respectively, using ECPEe systems in CPE and EPC modes. On the conclusion section, we will compare the communication resources as well as the number of membranes utilized by each solution. Additionally, we will suggest some related future works that can be done.

2. DEFINITION OF SOME NP-COMPLETE PROBLEMS

We define a graph as an ordered pair (V, E) where V is the set of vertices and and the set of edges $E \subseteq V \times V$.
As in [5], in this study, we deal only with simple graphs without loops and parallel edges. Figure 1 is a graph where $V = \{1, 2, 3, 4\}$ and $E = \{(1, 2), (2, 3), (3, 4)\}$.

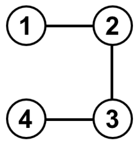

Figure 1. An Example of a Graph

A vertex cover VC_k where $1 \leq k \leq |V|$ is a set of vertices with size k where for all edges $(i, j) \in E$, $i \in VC$ or $j \in VC$.

Definition 1. ***Vertex Cover Problem (VCP)*** *Given a graph $G = (V, E)$ and a positive integer k where $1 \leq k \leq |V|$, is there a vertex cover VC_k?*

A boolean formula ϕ_X where X is a set of variables $x_1, x_2 \ldots x_p$ in conjunctive normal form (CNF) is a conjunction, denoted by $C_1 \wedge C_2 \wedge \ldots \wedge C_m$ where $m \in \mathbb{N}$, of propositional clauses C_i which are a disjunction of literals y_{i_j} defined as $C_i = (y_{i_1} \vee y_{i_2} \vee \ldots \vee y_{i_n})$ where $n \in \mathbb{N}$ and $y_{i_j} \in X \cup \{\overline{x} | x \in X\}, 1 \leq j \leq n$. The notation $\overline{x}$ implies a negation.
In a k-CNF boolean formula, each clause is a disjunction of exactly k variables. A boolean formula is said to be satisfiable if there exists a truth value (1 as true, 0 as false) assignment for all variables in which the formula evaluates to true.

Definition 2. ***3-SAT Problem (3SAT)*** *Given a 3-CNF boolean formula ϕ over a set of variables X, is ϕ satisfiable?*

An example of a satisfiable 3-CNF boolean formula is $\phi = (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3) \wedge (x_1 \vee \overline{x}_4 \vee x_5)$ where the configuration $x_1 = 0, x_2 = r, x_3 = r, x_4 = r, x_5 = 1, r \in \{0, 1\}$ makes the formula evaluate to true.

3. ECPE SYSTEMS

At this point, the readers are assumed to have core knowledge of membrane computing [10].

An Evolution-Communication P system (ECP system) is a type of P system introduced in [2] in which there is a separation between multiset rewriting rules, or simply evolution rules, and communication rules in the form of symport and antiport rules. It is said to be more realistic than other P systems for three reasons: evolution rules do not have target indications, simple symport/antiport rules are used for communication, and objects available in the environment are not needed in the beginning of the computation.
The authors in [1] used ECP system to analyze the communication complexity of P systems by introducing so-called *energy* objects to the system, hence the name Evolution-Communication P system with energy (ECPe system). These energy objects are assigned to each region and are used to enable communication such that every communication rule requires a certain number of energy. We can treat energy objects as catalysts: they can be produced by evolution rules but they themselves cannot evolve. We note here that while energy enables object transfer from one region to another, energy objects cannot pass through membranes for they are consumed during the communication.

Definition 3. *An ECPe system is formally defined as a construct of the form:*

$$\Pi = (O, e, \mu, w_1, \ldots, w_m, R_1, R_1', \ldots, R_m, R_m', i_{out})$$

i. m refers to the total number of membranes.

ii. O is the alphabet of the system.

iii. e is the energy object. $e \notin O$, should not be in the initial configuration, can not evolve and can not be transported from one region to another.

iv. μ represents the membrane structure and for this paper, we will be using square brackets with labels to denote a membrane. Membrane i is the the parent membrane of j, denoted by $parent(j)$, if j is located inside the square brackets that represents membrane i. Furthermore, if $parent(j)$ is i then $j \in children(i)$, i.e. j is a child membrane of i. For example, $[_i[_j]_j[_k]_k]_i$ is a membrane structure where membranes j, k are children of membrane i. For illustration purposes, we also used the representation of a membrane structure given in Figure 2.

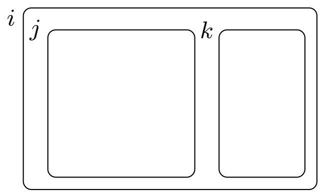

Figure 2. An example membrane structure where membranes j, k are children of membrane i

v. $w_1, \ldots, w_m \in O^*$ and each $w_i (1 \leq i \leq m)$ denotes the multiset of objects found in region i, i.e. the region bounded by membrane i.

vi. Each $R_i, 1 \leq i \leq m$, is a set of evolution rules associated with region i. An evolution rule takes the form $a \rightarrow v$ where $a \in O$ and $v \in (O \cup e)^*$.

vii. Each $R_i', 1 \leq i \leq m$, is a set of communication rules associated with membrane i. A communication rule can either be a symport or an antiport rule.

- A symport rule is or the form (ae^j, out) or (ae^j, in) where $a \in O$ and $j \geq 1$. If a symport rule is used, j copies of object e will be consumed, i.e. the j copies will not be transported, in order to transport a inside(denoted by *in*) or outside(denoted by *out*) of membrane i.

- An antiport rule is of the form $(ae^j, in; be^k, out)$ where $a, b \in O$ and $j, k \geq 1$. If an antiport rule is used, objects a and b will be switched and k amount of energy objects will be consumed by membrane i and j amount by $parent(i)$.

viii. $i_{out}, 1 \leq i \leq m$ is the output membrane. $i_{out} = 0$ means that the output is in the environment.

In ECP systems, rules are applied in a nondeterministic, maximally parallel manner. This means that at any step, when more than one rule can be applied to an object, the system will nondeterministically choose which rule to apply and all the rules that can be applied will be applied. According to [1], in selecting which rule to be used, three modes are considered: (i) communication has priority over evolution (CPE) in which if communication rules can be applied by the system, then only communication rules are performed at that step and no evolution rule is used ; (ii) evolution has priority over communication (EPC) wherein if evolution rules are applicable, then only evolution rules are used at that step; (iii) the application of evolution and communication rules are mixed together (CEM)
The state of the system specifies the configuration of the system at any time i and is denoted by C_i. $C_i \Rightarrow C_{i+1}$ denotes a transition step from C_i to C_{i+1}. In this paper, we refer to a transition step from C_i to C_{i+1} simply as Step $i+1$. $C_i \Rightarrow^* C_j, i < j$, denotes a computation- a series of transitions. A halting configuration is a configuration wherein no more rules can be applied. A computation succeeds if the system reaches a halting configuration. A system's output can be defined as the objects sent to the environment, that is, outside the outermost membrane, or objects sent to a specified output membrane.

3.1 Dynamical Communication Complexity Measures

From [1], the dynamical communication complexity measures that can be used in analyzing ECPe systems are:

$$ComN(w_i \Rightarrow w_{i+1}) = \begin{cases} 1, \text{ if a communication rule} \\ \text{is used in this transition} \\ 0, \text{ otherwise} \end{cases}$$

$$ComR(w_i \Rightarrow w_{i+1}) = \text{the number of communication rules used in this transition,}$$

$$ComW(w_i \Rightarrow w_{i+1}) = \text{the total energy of the communication rules used in this transition.}$$

Another definition from [1] is:

Definition 4. *We let $N_{mode}(\Pi)$ be the set of numbers computed by the system where mode $\in \{CPE, CEM, EPC\}$. For $ComX \in \{ComN, ComR, ComW\}$, the following are defined:*

$$ComX(\delta) = \sum_{i=0}^{h-1} ComX(C_i \Rightarrow C_{i+1}),$$
$$\text{for } \delta : C_0 \Rightarrow C_1 \Rightarrow \ldots \Rightarrow C_h$$
$$\text{is a halting computation,}$$

3.2 Solving Problems in ECPe Systems

In [5], the authors used the notion of recognizer P systems to P systems from [3] and the definition of recognizer ECPe systems from [4] as follows:

Definition 5. *Let Π be an ECPe system whose alphabet contains two distinct objects **yes** and **no**, such that every computation of Π is halting and during each computation, exactly one of the objects **yes**, **no** is sent out from the skin to signal acceptance or rejection. If all the computations of Π agree on the result, then Π is said to be confluent; if this is not necessarily the case, then it is said to be non-confluent and the global result is acceptance if and only if there exists an accepting computation.*

[3] also states that a decision problem can be represented as a pair $Y = (I_Y, \theta_Y)$ where I_Y is a language over a finite alphabet and θ_Y is a total boolean function over I_Y. (cod, s), where cod is an encoding of the initial configuration and $s \in N$, denotes a representation of an instance of a decision problem in P systems.
From [5], the concept of the P systems that will be used in solving a problem is based on both [3] and [4]. A family $\Pi(n), n \in \mathbb{N}$, of P systems is a set of P systems that takes a parameter n to construct each system.

Definition 6. *A family $\Pi(n), n \in \mathbb{N}$, of ECPe systems, solves a problem (I_Y, θ_Y) if there exists a pair (cod, s) over I_Y such that for each instance $u \in I_Y$:*

(i) $n = s(u) \in \mathbb{N}$ and $cod(u)$ is an input multiset of the system $\Pi(n)$

(ii) there exists an accepting computation of $\Pi(n)$ with input $cod(u)$ if and only if $\theta_Y(u) = 1$.

Definition 7. *Let $Y = (I_Y, \theta_Y)$ be a decision problem, $\Pi(n), n \in \mathbb{N}$, be a family of recognizer ECPe systems solving Y with a pair (cod, s) over I_Y. For each instance $u \in I_Y$,*

$$ComX(u, \Pi(n)) = min\{ComX(\delta) \mid \delta : C_0 \Rightarrow C_1 \Rightarrow \ldots \Rightarrow C_h \text{ in } \Pi(n) \text{ with } n = s(u) \text{ and } cod(u) \text{ is an input multiset in } \Pi(n)\},$$

where $X \in \{N, R, W\}$. To analyze the communication resources used by $\Pi(n)$ in solving problem $Y, ComX(Y, \Pi(n))$ is defined as:

$$ComX(Y, \Pi(n)) = max\{ComX(u, \Pi(n)) \mid u \in I_Y\}.$$

Definition 8. *Let $F_{mode}ComX$ where $X \in \{N, R, W\}$ and mode $\in \{CPE, CEM, EPC\}$. A decision problem $Y = (I_Y, \theta_Y) \in FComX(k)$ if and only if:*

(i) There exists a family $\Pi(n), n \in \mathbb{N}$, of confluent recognizer ECPe systems that decides Y

(ii) $ComX(Y, \Pi(n)) = k$.

The analogous complexity classes for non-confluent recognizer ECPe systems are $NF_{mode}ComN, NF_{mode}ComR$ $NF_{mode}ComW$.
We say that $Y \in F_{mode}ComNRW(p, q, r)$ if and only if $Y \in F_{mode}ComN(p), Y \in FComR(q)$ and $Y \in FComW(r)$. We use $NF_{mode}ComNRW$ for non-confluent recognizer ECPe systems.

This definition differs from that in [5] since we take into consideration the modes of the ECPe systems.
The authors in [5] used ECPe system under CEM and presented the following theorems:

Theorem 1. *$VCP \in NF_{CEM}ComNRW(6, |V_G|+3k+6, 3|E_G|+|V_G|+k+5)$ where E_G is the edge set and V_G is the vertex set of the input graph G.*

Theorem 2. *$3SAT \in NF_{CEM}ComNRW(5, 2n+3, 4n+3)$ where n is the number of clauses for the input 3-CNF boolean formula ϕ_X.*

In the following sections, we shall prove that VCP and 3SAT are both solvable using ECPe systems over CPE and EPC by constructing recognizer ECPe systems that satisfy Definitions 6 and 8. For this, we use the same representation and encoding of the said problems as in [5].

4. SOLUTIONS TO VERTEX COVER PROBLEM(VCP)

Definition 9. *Let the Vertex Cover Problem (VCP) be presented by a pair $VCP = (I_{VCP}, \theta_{VCP})$ where $I_{VCP} = \{w_{(G,k)} \mid w_{(G,k)}$ is a string representing a graph G and a positive integer $k\}$. If the graph G contains a vertex cover of size at most k, $\theta_{VCP}(w_{(G,k)}) = 1$; otherwise, $\theta_{VCP}(w_{(G,k)}) = 0$.*

As example, we use the graph G presented in Section 2 with $VC_2 \in \{\{2,3\},\{1,3\},\{2,4\}\}$.

4.1 Solution in CPE mode

Theorem 3. *$VCP \in NF_{CPE}ComNRW(5, |V_G|+3k+3, 2|E_G|+|V_G|+k+4)$ where E_G is the edge set and V_G is the vertex set of the input graph G.*

Proof. We define a family of ECPe systems over CPE as $\Pi_{VCP}(n)$ where $n = s(w_{(G,k)}) = |V_G|$:

$$\Pi_{VCP}(n) = (O, e, [_0[_1]_1]_0, w_0, \emptyset, \emptyset, R_0, R_0', R_1, R_1')$$

where:

- $O = \{A_{ij}, B_{ij}, v_i, \hat{v}_i, i, \hat{i}, \underline{i}, \bar{i} \mid 1 \le i < j \le n\} \cup \{c, c', d, d', d''\} \cup \{\alpha_0, \alpha_1, \alpha_2, \gamma, \#_0, \#_1, \#_2, \#_3, \#_4, \mathbf{no}, \mathbf{yes}\}$
- $w_0 = v_1 v_2 \dots v_n cod(w_{(G,k)})$
- $R_0 = \{A_{ij} \to B_{ij}, B_{ij} \to i, B_{ij} \to j \mid 1 \le i < j \le n\} \cup \{v_i \to \hat{v}_i e, i \to \bar{i} \mid 1 \le i \le n\} \cup \{c \to c'e^2$ $d \to d'\alpha_0 e, d' \to d'', d'' \to e\} \cup \{\alpha_2 \to \mathbf{no}e, \#_0 \to \#_1, \#_1 \to \#_2, \#_2 \to \#_3, \#_3 \to \#_4, \#_4 \to \mathbf{yes}e\}$
- $R_0' = \{(\mathbf{no}e, out), (\mathbf{yes}e, out)\}$
- $R_1 = \{\hat{v}_i \to \hat{i}, \hat{i} \to \gamma, i \to \underline{i}^{n-2}e^{n-2} \mid 1 \le i \le n\} \cup \{c' \to e, \alpha_0 \to \alpha_1, \alpha_1 \to \alpha_2\}$
- $R_1' = \{(\hat{v}_i e, in), (ie, in; \hat{i}e, out), (\bar{i}e, in; \underline{i}e, out) \mid 1 \le i \le n\} \cup \{(c'e, in), (\alpha_0 e, in), (\#_4 e, in; \alpha_2 e, out)\}$

As in [5], (cod, s) over I_{VCP} is defined as such that for a given instance $w_{(G,k)} \in I_{VCP}$ we have $n = s(w_{(G,k)}) = |V_G|$ and the encoding $cod(w_{(G,k)})$ is a multiset containing A_{ij} for every $(i,j) \in E_G$, k copies of object c and $|E_G|-k$ copies of object d. To satisfy the condition (i) of Definition 6, this encoding is placed in region 0 ensuring that $s(u)$is a natural number and $cod(u)$ is an input multiset for $\Pi_{VCP}(n)$.
To show that we can satisfy the condition (ii) of Definition 8, we shall also break down the solution into phases (namely the setup, finding the candidate solution, validation, and output phases) patterned by the process presented in [5].

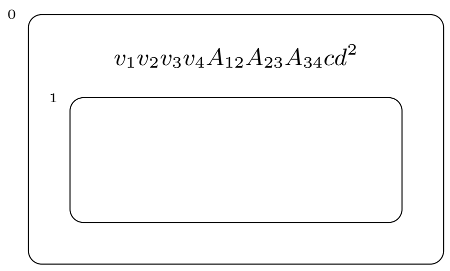

Figure 3. C_0: Initial configuration given G and $k = 1$

Setup phase. This is the phase where we represent each vertex of the input graph in region 1, and delegate a vertex for each edge present in region 0.

Step 1: All objects v_i, A_{ij}, c, d, and $\#_0$ evolve to $\hat{v}_i, B_{ij}, c'$, d', and $\#_1$ respectively. The evolution rules $v_i \to \hat{v}_i e$, $c \to c'e^2$, and $d \to d'\alpha_0 e$ produce energy objects and α_0.

Step 2: This step involves communicating the objects $\hat{v}_i, c'$, and α_0 through membrane 1 using the energy produced in the first step; leaving k energy in region 0 (produced by the rule $c \to c'e^2$) where $k =$ the maximum size of the vertex cover.

Step 3: Representative vertices for the edges are non-deterministically chosen when objects B_{ij} evolve to either i or j by rules $B_{ij} \to i$ or $B_{ij} \to j$. Simultaneously, in region 1, $\hat{v}_i$ evolves to $\hat{i}$ (through $\hat{v}_i \to \hat{i}$), and objects c' and α_0 transform to e and α_1, respectively. At the same time, $\#_1$ evolves to $\#_2$.

Finding a candidate solution. In this phase, a candidate vertex cover is chosen, and vertices composing the candidate are transferred from region 1 to region 0.

Step 4: At this point, a representation of all the vertices in the form of the objects $\hat{i}$ is present in region 1. Rules $\hat{i} \rightarrow \gamma$ and $(ie, in; \hat{i}e, out)$ are both available for the objects. However, since communication has priority over evolution, only the latter rule is used. The same principle applies to objects i in region 0 on which the evolution rule $i \rightarrow \bar{i}$ cannot be applied yet. These antiport rules let the system to non-deterministically choose a possible solution of size k and transfer them from region 1 to region 0. Hence, the communicated objects $\hat{i}$ in region 0 is the candidate solution.

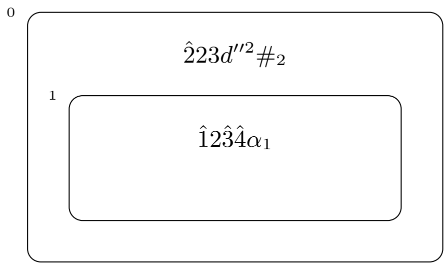

Figure 4. C_4: Candidate solution $\{\hat{2}\}$ is chosen

Validating the solution. In region 1, the candidate solution is represented by the objects i. These are used to validate that the selected vertex cover includes all edges.

Step 5: The next thing to do is to evolve objects i to $\bar{i}$ (through rule $i \rightarrow \bar{i}$)in region 0, and objects $\hat{i}$ to γ (through rule $\hat{i} \rightarrow \gamma$) in region 1. This step is necessary to prevent the communication rule $(ie, in; \hat{i}e, out)$ from repeating when energy objects are produced for validation. Simultaneously, the evolution rule $i \rightarrow \underline{i}^{n-2}e^{n-2}$ is applied to each objects i in region 1 producing $n-2$, which is the maximum number of edges a vertex can have minus one edge already verified when the candidate solution was selected, $\underline{i}$ energy objects. At the same instant $\#_2$ evolves to $\#_3$ and α_1 to α_2.

Step 6: Communication is prioritized and so the antiport rule $(\bar{i}e, in;$
$\underline{i}e, out)$ is used to confirm that the candidate solution covers all the edges. If no object $\underline{i}$ remains in region 1, it means that the chosen vertex cover is verified.

Step 7: $\#_3$ evolves to $\#_4$.

Output phase. In the output phase, the object **yes** is released to the environment if the chosen vertex cover is valid; otherwise, **no** is released.

Step 8: If all the edges where verified, there will be no remaining energy present in region 0, hence no communication can take place. The evolution rule $\#_4 \rightarrow$ **yes**e is applied. If, however, not all the edges were covered, α_2 is sent to region 0 and $\#_4$ to region 1 by the rule $(\#_4 e, in; \alpha_2 e, out)$.

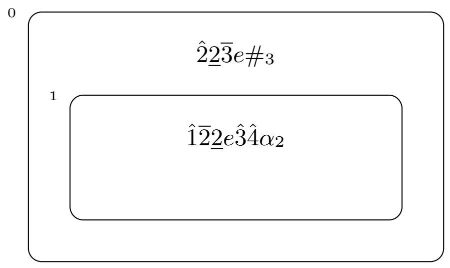
0
$\hat{2}\underline{2}\bar{3}e\#_3$
1
$\hat{1}\bar{2}\underline{2}e\hat{3}\hat{4}\alpha_2$

Figure 5. C_6: Since VC_1 for G does not exist, not all edges were verified leaving $\underline{2}$ in region 1, and an energy object in regions 0 and 1

Step 9: If the chosen candidate solution is valid, **yes** will be sent to the environment and the computation will halt. Otherwise, α_2 evolves to **no**e.

Step 10: **no** is sent to the environment and the computation halts.

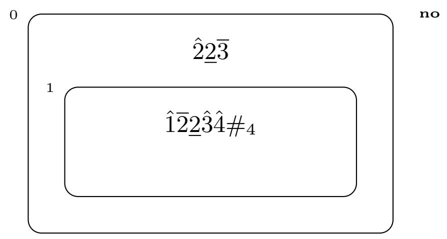
0
no
$\hat{2}\underline{2}\bar{3}$
1
$\hat{1}\bar{2}\underline{2}\hat{3}\hat{4}\#_4$

Figure 6. C_{10} : **no** is sent to the environment and computation halts

We then compute for the communication complexity at each phase of the computation to show that Definition 8 is satisfied:

a. In the setup phase,
   - number of communication step is one (involving symport rules for objects v_i, c', and α_0)
   - number of communication rules used is $|V_G|+k+1$
   - number of energy objects used is $|V_G|+k+1$

b. In finding a candidate solution,
   - number of communication step is one (involving antiport rules for of objects $\hat{v}_i$ and i)
   - number of communication rules used is k
   - number of energy objects used is $2k$

c. In validating the candidate solution,
   - number of communication step is one (involving antiport rules for objects $\bar{i}$ and $\underline{i}$ to validate remaining edges)
   - number of communication rules used is k
   - number of energy objects used is $2(|E_G|-k)$

d. In the output phase,
   - maximum number of communication step is 2 when **no** is released to the environment (through rules $(\#_4 e, in;\ \alpha_2 e, out)$ and $(\mathbf{no}e, out)$)

- maximum number of communication rules used is 2
- maximum number of energy objects used is 3

Summing all the resources we computed above, we have proven that $VCP \in NF_{CPE}ComNRW(5, |V_G|+3k+3, 2|E_G|+|V_G|+k+4)$.

4.2 Solution in EPC mode

Theorem 4. *$VCP \in NF_{EPC}ComNRW(9, |V_G|+3k+16, 3|E_G| + |V_G|+k+15)$ where E_G and V_G are the set of edges and vertices, respectively.*

Proof. A family of ECPe systems over EPC that solves VCP is defined as $\Pi_{VCP}(n)$:

$$\Pi_{VCP}(n) = (O, e, [_0[_1]_1[_2]_2[_3]_3]_0, w_0, \emptyset, \emptyset, R_0, R_0', R_1, R_1', R_2, R_2', R_3, R_3')$$

where

- $O = \{A_{ij}, v_i, \hat{v_i}, i, \hat{i}, \underline{i} | 1 \le i < j \le n\} \cup \{c, c', d, d'\} \cup \{\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5\} \cup \{\beta_0, \beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \beta_6, \beta_7, \beta_8, \Omega\}$
- $w_0 = v_1 v_2 \ldots v_n cod(w_{(G,k)})\alpha_0\beta_0$
- $R_0 = \{A_{ij} \to ie, A_{ij} \to je | 1 \le i < j \le n\} \cup \{v_i \to \hat{v_i}e | 1 \le i \le n\} \cup \{c \to c'e^2, d \to d'e\} \cup \{\beta_0 \to \beta_1 e, \beta_2 \to \beta_3 e, \beta_4 \to \beta_5 \Omega e^2, \beta_6 \to \beta_7 e, \beta_8 \to \mathbf{yes}e, \alpha_0 \to \alpha_1 e, \alpha_2 \to \alpha_3 e, \alpha_4 \to \alpha_5 e\}$
- $R_0' = \{(\mathbf{no}e, out), (\mathbf{yes}e, out)\}$
- $R_1 = \{\hat{v_i} \to \hat{i} | 1 \le i \le n\} \cup \{c' \to e\}$
- $R_1' = \{(\hat{v_i}e, in), (ie, in; \hat{i}e, out) | 1 \le i \le n\} \cup \{(c', in)\}$
- $R_2 = \{d' \to e\} \cup \{\hat{i} \to \underline{i}^{n-2} | 1 \le i < j \le n\} \cup \{\alpha_1 \to \alpha_2 e, \alpha_3 \to \alpha_4 e, \alpha_5 \to \mathbf{no}\}$
- $R_2' = \{(\hat{i}e, in), (ie, in; \underline{i}e, out) | 1 \le i \le n\} \cup \{(d'e, in), (\alpha_1 e, in), (\alpha_3 e, in), (\alpha_5 e, in)\} \cup \{(\alpha_2 e, out), (\alpha_4 e, out)\} \cup \{(\Omega e, in; \mathbf{no}e, out)\}$
- $R_3 = \{\beta_1 \to \beta_2 e, \beta_3 \to \beta_4 e, \beta_5 \to \beta_6 e, \beta_7 \to \beta_8 e\}$
- $R_3' = \{(\beta_1 e, in), (\beta_3 e, in), (\beta_5 e, in), (\beta_7 e, in), \} \cup \{(\beta_2 e, out), (\beta_4 e, out), (\beta_6 e, out)\} \cup \{(\Omega e, in; \beta_8 e, out)\}$

As with the ECPe system in CPE mode, this ECPe system uses the same encoding and follows the same pattern of solution as that in [5].

Setup phase. Similar to the setup phase in CPE mode, in this phase, a vertex is nondeterministically chosen to cover each edge. Also, a representation of each vertex communicated to region 1.

Step 1: In the first step, each v_i, A_{ij}, c, d, α_0 and β_0 will evolve into $\hat{v_i}e$, ie or je, $c'e^2$, $d'e$, $\alpha_1 e$ and $\beta_1 e$, respectively. The value of i and j represents which vertex will represent the edge A_{ij}.

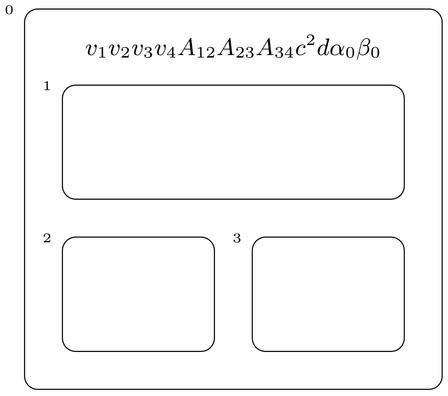

Figure 7. C_0: Initial configuration given G and $k = 2$

Step 2: All the objects $\hat{v_i}$ and c' are transported to region 1 using the rules $(\hat{v_i}e, in)$ and (c', in), respectively. At the same step, all the d' together with α_1 are transported to region 2 and β_1 to region 3.

Step 3: All the communicated objects will evolve using the rules $\hat{v_i} \to \hat{i}$, $c' \to e$, $d' \to e$, $\alpha_1 \to \alpha_2 e$ and $\beta_1 \to \beta_2 e$.

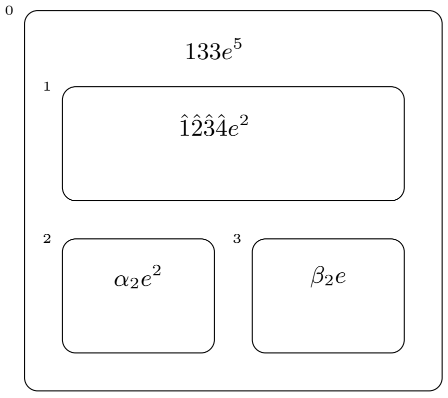

Figure 8. C_3: Representative vertices $1, 3, 3$ were chosen in C_1 for edges $(1,2), (2,3), (3,4)$ respectively and other objects were set up to their respective regions and evolved

Finding a candidate solution. From the representation of vertices in region 1, a candidate vertex cover is chosen and is communicated to region 0.

Step 4: This step involves swapping k number of objects $\hat{i}$ in region 1 with their counterparts in region 0 using the antiport rule $(ie, in; \hat{i}e, out)$. Also done in this step, α_2 and β_2 are transported to region 0.

Step 5: α_2 and β_2 evolves into α_3 and β_3, respectively.

Validating the solution. A representation of the candidate vertex cover is transported to region 2 from region 1 and this is used to validate if all the edges are covered by the selected vertex cover. A vertex cover is valid if no representation of the edges remain in region 0.

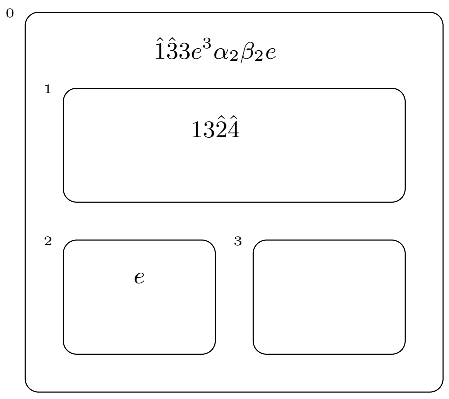

Figure 9. C_4: Candidate solution $\{\hat{1},\hat{3}\}$ were chosen

Step 6: The representative vertex cover as well as α_3 will be communicated to region 2 and β_3 will be transported to region 3.

Step 7: Evolution rules $\hat{i} \to \underline{i}^{n-2}$, $\alpha_3 \to \alpha_4 e$ and $\beta_3 \to \beta_4 e$ will be used. The objects $\underline{i}$ will be used to verify the remaining edges in region 0.

Step 8: The antiport rule $(ie, in; \underline{i}e, out)$ will be applied to verify if the chosen vertex cover covers all the edges. If no object i remains in region 0, then the vertex cover is valid. In this same step, α_4 and β_4 are transported to region 0.

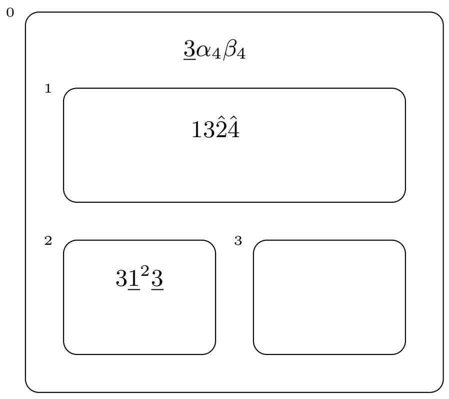
0
$\underline{3}\alpha_4\beta_4$
1
$13\hat{2}\hat{4}$
2
$3\underline{1}^2\underline{3}$
3

Figure 10. C_8: Remaining edge represented by 3 was validated

Step 9: α_4 evolves to α_5 and β_4 will evolve to $\beta_5\Omega e^2$.

Output phase. In the output phase, the object **yes** is released to the environment if the chosen vertex cover is valid and **no** is released otherwise.

Step 10: α_5 is communicated to region 2 and β_5 to region 3.

Step 11: α_5 evolves to **no** and β_5 to $\beta_6 e$.

Step 12: β_6 is transported to region 0 (through rule $(\beta_6 e, out)$) and **no** may also be switched with Ω (though rule $(\Omega e, in; \beta_8 e,$
$out)$) in region 0. Note that **no** will only be communicated to region 0 if there is at least one energy object left in region 2 and this will only happen if not all the edges were verified, meaning, the vertex cover is not valid.

Step 13: β_6 evolves to $\beta_7 e$.

Step 14: β_7 is communicated to region 3 and **no** may be released to the environment depending on whether the vertex cover is valid or not.

Step 15: If *no* was released to the the environment in the previous step, β_7 evolves to $\beta_8 e$ and the computation halts. In the case that the vertex cover is valid, however, β_7 still evolves but the computation will not yet halt.

Step 16: The antiport rule $(\Omega e, in; \beta_8 e, out)$ is applied.

Step 17: β_8 evolves to **yes**e.

Step 18: Lastly, **yes** is released to the environment using the communication rule $(\mathbf{yes}e, out)$.

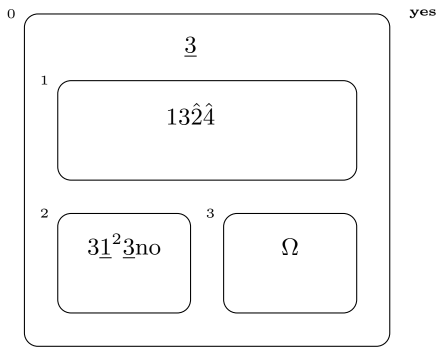

Figure 11. C_{18}: **yes** was sent to the environment and the computation halts

As a conclusion to our proof, we show that Definition 8 is satisfied by analyzing the communication resources at each phase of the computation.

a. The setup phase involves one communication step. In this communication step, the objects $v_i (1 \le i \le |V_G|)$, c', d', α_1 and β_1 will be communicated and for each of these objects, one energy object is consumed.
   - number of communication step is one
   - number of communication rules used is $|V_G|$+4
   - number of energy objects consumed is $|E_G|$+$|V_G|$ +2

b. Finding a candidate solution also involves one communication step. In this phase $2k$(this size of the vertex cover) number of objects are swapped in order to choose a candidate vertex cover. Also done in this phase is the communication of α_2 and β_2 to region 0.
   - number of communication step is one
   - the number of communication rules applied is $k+2$
   - the number of energy objects consumed is $2k+2$

c. For the validation and output phase:

- The first communication step is used to transport the candidate vertex cover to region 2. Another communication step may be used in checking if the vertex cover covers all the remaining edges. This step is optional because it may not happen depending on the chosen vertex cover. Note, however, that if this step does not happen, then the vertex cover is not valid and if it does occur, it does not necessarily mean that the vertex cover is valid. Another communication step is used in transporting α_5 and β_5. Another four communication steps will occur if the answer is **yes** and this is the maximum. The total number of communication steps in this phase is 7.
- Since the size of the vertex cover is k, in the first communication step, k rules will be used to communicate the candidate vertex cover to region 2 and 2 additional rules will be used to communicated α_3 and β_3. In the second communication step, a maximum of k rules of the form $(ie, in; \underline{i}e, out)$ will be used to verify the remaining edges and 2 rules will be used to transport α_4 and β_4. In the next communication step, 2 rules will be used to transport α_5 and β_5. Since we are computing for the maximum amount of communication rules used, from this point, we will only consider the case if the vertex cover selected is valid since it uses the most communication rules. In the proceeding communication step, β_6 will be communicated to region 0 and this uses one communication rule. Another communication step will be used to transport β_7. In the next communication step, an antiport rule of the form $(\Omega e, in; \beta_8 e, out)$ will be used in preparation for the release of **yes** in the environment. Lastly, **yes** will be released into the environment. The total number of communication rules in the validation and output phases is $10 + 2k$.
- Following the explanation above, the number of consumed energy objects is $11 + 2|E_G| - k$.

From the analysis above, we can observe that the maximum communication cost will be incurred if the chosen vertex cover is valid. As a summary, for this ECPe system used in solving the VCP, $ComNRW(9, |V_G|+3k+16, 3|E_G|+|V_G|+k+15)$.

5. SOLUTIONS TO 3-SATISFIABLE PROBLEM (3SAT)

We now move on to solutions to 3SAT using ECPe systems in CPE and EPC modes.
Again, we used the representation of 3SAT as described in [5].

Definition 10. *Let the 3-SAT problem be represented by a pair $(I_{3SAT}, \theta_{3SAT})$ where $I_{3SAT} = \{w_{(\phi_X)} | w_{(\phi_X)}$ is a string representing a 3-CNF boolean formula $\phi_X\}$. Boolean function $\theta_{3SAT}($ $w_{\phi_X}))$ evaluates to 1 if ϕ is satisfiable, otherwise, $\theta_{3SAT}(w_{\phi_X})$ $= 0$.*

As an example, we use the boolean formula $\phi_X = (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3) \wedge (x_1 \vee \overline{x}_4 \vee x_5)$ where $n = 2$.

5.1 Solution in CPE mode

Theorem 5. $3SAT \in NF_{CPE}ComNRW = (4, 2n+1, 3n+1)$ *where n is the number of clauses for the input 3-CNF boolean formula.*

Proof. We define a family of ECPe systems over CPE as:

$$\Pi_{3SAT}(n) = (O, e, [_0[_1]_1]_0 \ldots [_n]_n, w_0, \emptyset, \ldots, \emptyset, R_0, R'_0, R_1, R'_1, \ldots, R_n, R'_n)$$

where:

- $O = \{x_d, d, \hat{d} | 1 \leq d \leq 3n\} \cup \{0_{dq}, 1_{dq}, A_{i_1 i_2 i_3, q}, B_{i_1 i_2 i_3, q} | 1 \leq d \leq 3n, 1 \leq q \leq n$ and $i_r \in \bigcup_{d=1}^{3n} \{d, \hat{d}\}, \forall r \in \{1,2,3\}\} \cup \{c, \beta_0, \beta_1, \#_0, \#_1, \#_2, \#_3, \textbf{no}, \textbf{yes}\}$
- $w_0 = x_1 x_2 \ldots x_{3n} \#_0 cod(w_{\phi x})$
- $R_0 = \{x_d \to 0_{d1} \ldots 0_{dn}, x_d \to 1_{d1} \ldots 1_{dn} | 1 \leq d \leq 3n\} \cup \{A_{i_1 i_2 i_3, q} \to B_{i_1 i_2 i_3, q} c e^2 | 1 \leq q \leq n$ and $i_r \in \bigcup_{d=1}^{3n} \{d, \hat{d}\}, \forall r \in \{1,2,3\}\} \cup \{\#_0 \to \#_1, \#_1 \to \#_2, \#_2 \to \#_3, \#_3 \to \textbf{yes}e, \beta_1 \to \textbf{no}e\}$
- $R'_0 = \{(\textbf{yes}e, out), (\textbf{no}e, out)\}$
- For $1 \leq q \leq n$:
  - $R_q = \{B_{i_1 i_2 i_3, q} e \to i_1 i_2 i_3 \beta_0 e | i_r \in \bigcup_{d=1}^{3n} \{d, \hat{d}\}, \forall r \in \{1,2,3\}\} \cup \{\beta_0 \to \beta_1\}$
  - $R'_q = \{(B_{i_1 i_2 i_3, q} e, in) | i_r \in \bigcup_{d=1}^{3n} \{d, \hat{d}\}, \forall r \in \{1,2,3\}\}$ $\cup \{(0_{dq} e, in; \hat{d}e, out), (1_{dq} e, in; de, out) | 1 \leq d \leq 3n\} \cup \{(\#_3 e, in; \beta_1, out)\}$

Again, as in [5], the pair (cod, s) over I_{ϕ_X} is defined as such where for each instance $w_{\phi_X} \in I_{\phi_X}$, $s(w_{\phi_x}) = n$ where n is the number of clauses for the boolean formula ϕ_X. The encoding $cod(w_{\phi_X})$ is defined as a multiset containing $A_{i_1 i_2 i_3, q}$ for $1 \leq q \leq n$ where if $C_q = y_{i_1,q} \vee y_{i_2,q} \vee y_{i_3,q}$, then

$$i_l = \begin{cases} d \text{ if } y_{i_l,q} = x_d \\ \hat{d} \text{ if } y_{i_l,q} = \overline{x}_d \end{cases}$$

for $l = \{1,2,3\}$, where $x_d \in \{x_1, x_2, \ldots, x_n\}$. Note that $s(w_{\phi_x})$ is the number of clauses, hence a natural number, and $cod(w_{\phi_X})$ showing that condition (i) of Definition 6 holds. We show condition (ii) is satisfied through the systems computation.

Setup and finding a candidate solution phase. In this phase, each variable is assigned a truth value and each clause i where $1 \leq i \leq n$ is represented in each region q, $1 \leq q \leq n$.

Step 1: The first thing to do is to nondeterministically assign a truth value to each variable x_i by applying the rule

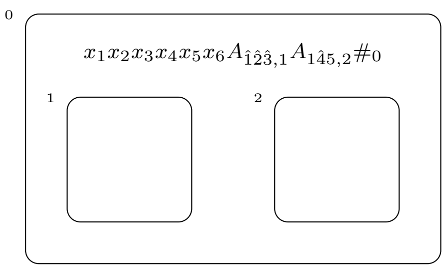
0
$x_1x_2x_3x_4x_5x_6A_{\hat{1}\hat{2}\hat{3},1}A_{1\hat{4}5,2}\#_0$
1
2

Figure 12. C_0: Initial configuration given boolean formula ϕ_X

$x_d \to 0_{d1}\dots 0_{dn}$ or $x_d \to 1_{d1}\dots 1_{dn}$. At the same instant, objects $A_{i_1i_2i_3,q}$ evolve to $B_{i_1i_2i_3,q}$ and produce $2n$ energy necessary for the setting up the clauses and validating them. Also, $\#_0$ becomes $\#_1$.

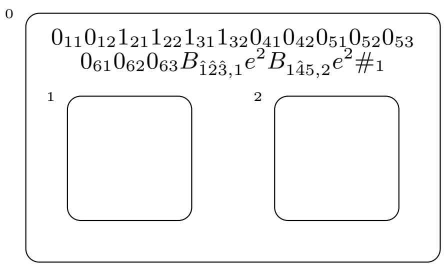
0
$0_{11}0_{12}1_{21}1_{22}1_{31}1_{32}0_{41}0_{42}0_{51}0_{52}0_{53}$
$0_{61}0_{62}0_{63}B_{\hat{1}\hat{2}\hat{3},1}e^2B_{1\hat{4}5,2}e^2\#_1$
1
2

Figure 13. C_1: Each variable $x_i, 1 \le i \le 3n$ was assigned a value

Step 2: In this step, communication is prioritized and $B_{i_1i_2i_3,q}$ objects are transferred to their corresponding region q.

Step 3: Evolution rule $B_{i_1i_2i_3,q} \to i_1i_2i_3\beta_0$ is used in each region q producing a representation of each clause q where $1 \le q \le n$.

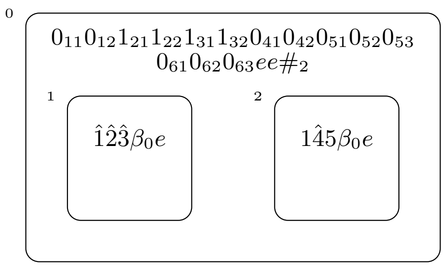

Figure 14. C_3: Each $B_{i_1i_2i_3,q}$ was transferred to corresponding q membranes in Step 2 and evolved to $i_1i_2i_3, q\beta_0e$ in Step 3

Validating candidate solution and output phase. In this phase, it is checked whether each clause evaluates to true. If all of the clauses are verified, then the object **yes** is sent to the environment and **no** otherwise.

Step 4: The clauses are verified by applying the antiport rules $(0_{dq}e,\ in; \hat{d}e, out)$ and $(1_{dq}e, in; de, out)$.

Step 5: Objects $\#_2$ and β_0 evolve to $\#_3$ and β_1 respectively.

Step 6: If all the clauses evaluates to true, there would be no energy left in the system and communication cannot ensue and $\#_3$ proceeds to evolve to **yes**e. If not all of the clauses evaluates to true, there is enough energy for a clause which evaluates to false to use the antiport rule $(\#_3e, in; \beta_1e, out)$.

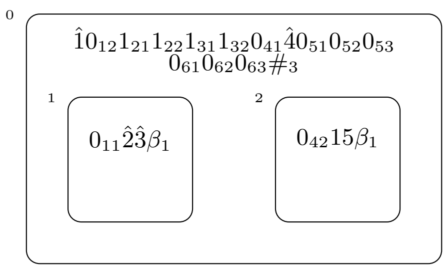
0
$\hat{1}0_{12}1_{21}1_{22}1_{31}1_{32}0_{41}\hat{4}0_{51}0_{52}0_{53}$
$0_{61}0_{62}0_{63}\#_3$
1
2
$0_{11}\hat{2}\hat{3}\beta_1$
$0_{42}15\beta_1$

Figure 15. C_5: Each clause were verified in Step 4 using rules $(0_{11}e,$ $in; \hat{1}e, out)$ and $(0_{42}e,\ in; \hat{4}e, out)$. $\#_2$ and β_0 evolved to $\#_3$ and β_1 respectively in Step 5

Step 7: If all clauses are satisfied, **yes** is communicated out to the environment and the computation stops. Otherwise, β_1 in region 0 evolves to **no**e.

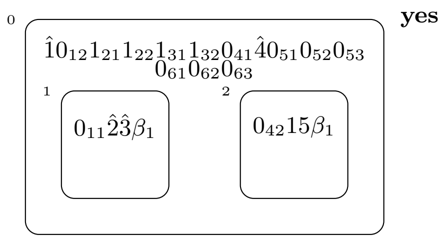
0
yes
$\hat{1}0_{12}1_{21}1_{22}1_{31}1_{32}0_{41}\hat{4}0_{51}0_{52}0_{53}$
$0_{61}0_{62}0_{63}$
1
2
$0_{11}\hat{2}\hat{3}\beta_1$
$0_{42}15\beta_1$

Figure 16. C_7: All clauses were satisfied, hence **yes** is sent to the environment and the computation halts

Step 8: Lastly, if not all clauses are satisfied, **no** is sent to the environment and the computation halts.

We now compute the communication resources used at each phase.

a. In the setup phase,
   - The number of communication steps is one (involving symport rules for each clause representation $A_{i_1i_2i_3,q}$).
   - The number of communication rules used is n.
   - The number of energy consumed is n.

b. In the validation and output phase,
   - The maximum number of communication steps is 3 which occurs when not all clauses are satisfied, and **no** is sent to the environment. These 3 steps involves antiport rules for the verified clauses, the rule $(\#_3e, in; \beta_1e, out)$, and the symport rule for communicating **no** to the environment.
   - The maximum number of communication rules $n+1$ for sending a **no** to the environment occurs when only $n-1$ clauses are verified. This is equal to the number of communication rules used when the 3-CNF boolean formula is satisfiable.

- The maximum number of energy consumed is $2n+1$ which is equal for sending a **no** or a **yes** to the environment.

From the calculations above, we have shown that $3SAT$ is indeed in $NF_{CPE}ComNRW(4, 2n+1, 3n+1)$.

5.2 Solution in EPC mode

Theorem 6. $3SAT \in NF_{EPC}ComNRW = (6, 4n+7, 7n+7)$ *where n is the number of clauses for the input 3-CNF boolean formula.*

Proof. We define a family of ECPe systems that solve 3SAT as a construct Π_{3SAT}:

$$\Pi_{3SAT}(n) = (O, e, [_0[_1]_1 \dots [_n]_n[_{n+1}]_n+1]_0, w_0, \emptyset, \dots, \emptyset,$$
$$R_0, R'_0, R_1, R'_1, \dots, R_n, R'_n, R_{n+1}, R'_{n+1})$$

where

- $O = \{x_d, d, \hat{d} | 1 \leq d \leq 3n\} \cup \{0_{dq}, 1_{dq}, A_{i_1i_2i_3,q}, B_{i_1i_2i_3,q} | 1 \leq d \leq 3n, 1 \leq q \leq n$ and $i_r \in \bigcup_{d=1}^{3n}\{d, \hat{d}\}\} \cup \{c, \alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4, \Omega, \beta_0, \beta_1, \beta_2, \beta_3, \mathbf{no}, \mathbf{yes}\}$
- $w_0 = x_1x_2 \dots x_{3n}\alpha_0 cod(w_{\phi x})$
- $R_0 = \{x_d \rightarrow 0_{d1} \dots 0_{dn}, x_d \rightarrow 1_{d1} \dots 1_{dn} | 1 \leq d \leq 3n\} \cup \{A_{i_1i_2i_3,q} \rightarrow B_{i_1i_2i_3,q}ce^2 | 1 \leq q \leq n$ and $i_r \in \bigcup_{d=1}^{3n}\{d, \hat{d}\}, \forall r \in \{1,2,3\}\} \cup \{\alpha_0 \rightarrow \alpha_1\Omega e^2, \alpha_2 \rightarrow \alpha_3 e, \alpha_4 \rightarrow \mathbf{yes}e^2, \beta_0 \rightarrow \beta_1 e\} \cup \{d \rightarrow e, \hat{d} \rightarrow e | 1 \leq d \leq 3n\}$
- $R'_0 = \{(\mathbf{no}e, out), (\mathbf{yes}e^{n+2}, out)\}$
- For $1 \leq q \leq n$:
  - $R_q = \{B_{i_1i_2i_3,q} \rightarrow i_1i_2i_3\beta_0 e^3 | i_r \in \bigcup_{d=1}^{3n}\{d, \hat{d}\}, \forall r \in \{1,2,3\}\} \cup \{\beta_1 \rightarrow \beta_2\}$
  - $R'_q = \{(B_{i_1i_2i_3,q}e, in) | i_r \in \bigcup_{d=1}^{3n}\{d, \hat{d}\}, \forall r \in \{1,2,3\}\} \cup \{(\beta_0 e, out), (\beta_1 e, in), (ce, in; \beta_2 e, out)\} \cup \{(0_{dq}e, in; \hat{d}e^2,\ out), (1_{dq}e, in; de^2,\ out) | 1 \leq d \leq 3n\}$
- $R_{n+1} = \{\alpha_1 \rightarrow \alpha_2 e, \alpha_3 \rightarrow \alpha_4 e, \Omega \rightarrow \mathbf{no}e\}$
- $R'_{n+1} = \{(\Omega e, in), (\alpha_1 e, in), (\alpha_2 e, out), (\alpha_3 e, in), (\alpha_4, out), (\beta_2 e, in; \mathbf{no}e, out)\}$

This ECPe system will use the same encoding as [5].

Setup and finding a candidate solution phase. In this phase, a truth value is assigned to each variable and each region $i(1 \leq i \leq n)$ is given a representation of the *ith* clause.

Step 1: In the first step, each x_i from the initial configuration is assigned a truth value using the rule $x_d \rightarrow 0_{d1} \dots 0_{dn}$ or $x_d \rightarrow 1_{d1} \dots 1_{dn}$. Also, the evolution rules $A_{i_1i_2i_3,q} \rightarrow B_{i_1i_2i_3,q}ce^2$ and $\alpha_0 \rightarrow \alpha_1\Omega e^2$ are used. The objects 0_{dq} and 1_{dq} represent the candidate assignment for each variable.

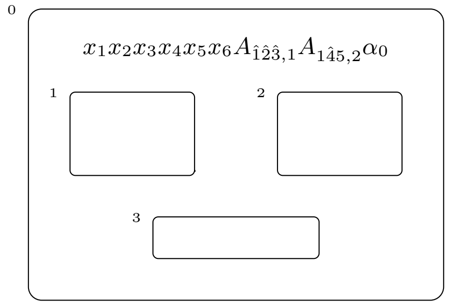
0
$x_1x_2x_3x_4x_5x_6A_{\hat{1}\hat{2}\hat{3},1}A_{1\hat{4}5,2}\alpha_0$
1
2
3

Figure 17. C_0: Initial configuration given boolean formula ϕ_X

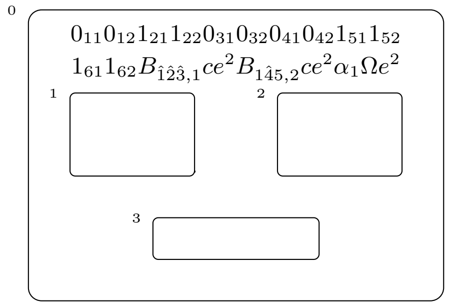
0
$0_{11}0_{12}1_{21}1_{22}0_{31}0_{32}0_{41}0_{42}1_{51}1_{52}$
$1_{61}1_{62}B_{\hat{1}\hat{2}\hat{3},1}ce^2B_{1\hat{4}5,2}ce^2\alpha_1\Omega e^2$
1
2
3

Figure 18. C_1: Each variable $x_i, 1 \leq i \leq 3n$ was assigned a value

Step 2: In the second step, each $B_{i_1i_2i_3,q}$ is communicated to region $q(1 \leq q \leq n)$ and α_1 and Ω are transported to region $n+1$.

Validating candidate solution and output phase. In this phase, it is checked whether each clause evaluates to true. If yes, then the object **yes** is sent to the environment and **no** otherwise.

Step 3: α_1 evolves into $\alpha_2 e$, Ω evolves into $\mathbf{no}e$ and each $B_{i_1i_2i_3,q}$ uses the evolution rule $B_{i_1i_2i_3,q} \rightarrow i_1i_2i_3\beta_0 e^3$.

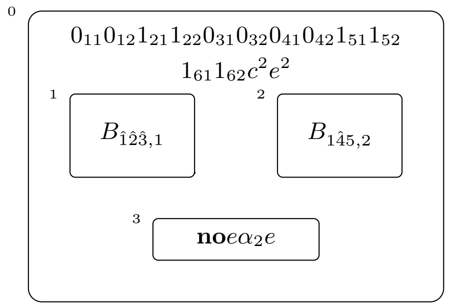

Figure 19. C_3: Each $B_{i_1i_2i_3,q}$ was transferred to corresponding q membranes in Step 2 and evolved to $i_1i_2i_3, q\beta_0 e^3$ in Step 3

Step 4: In this step, it is checked if each clause can be evaluated to true. This is done using the antiport rules $(0_{dq}e, in; \hat{d}e^2, out)$ and $(1_{dq}e, in; de^2, out)$. In the same step, the objects β_0 and α_2 are transported to region 0. If each region q successfully performs the either of the antiport rules, then the answer to 3SAT for the given boolean formula is **yes**.

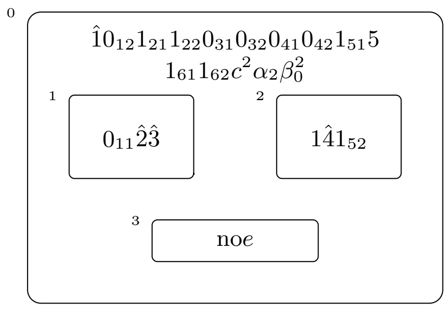
0
$\hat{1}0_{12}1_{21}1_{22}0_{31}0_{32}0_{41}0_{42}1_{51}5$
$1_{61}1_{62}c^2\alpha_2\beta_0^2$
1
$0_{11}\hat{2}\hat{3}$
2
$1\hat{4}1_{52}$
3
no$e$

Figure 20. $C_5 4$: Each clause were verified using rules $(0_{11}e, in; \hat{1}e, out)$ and $(0_{42}e,\ in; \hat{4}e, out)$

Step 5: All the objects d and $\hat{d}$ in region 0 evolve into energy objects, α_2 evolves into $\alpha_3 e$ and all objects β_0 evolve into $\beta_1 e$.

Step 6: All of the objects β_1 are transported to their respective regions and α_3 is communicated to region $n+1$

Step 7: α_4 is transported to region 0. In case the given boolean formula is not satisfiable, the antiport rule $(ce, in;$ $\beta_2 e, out)$ will be used by all membranes that have remaining energy objects in them, i.e. membranes that have not found a truth value that can make their corresponding propositional clauses true.

Step 8: α_4 evolves into **yes**e^2.

Step 9: If the given boolean formula is satisfiable, there will be enough energy to send the object **yes** to the environment and the computation will halt. Otherwise, the antiport rule $(\beta_2 e, in; \mathbf{no}e, out)$ will be used once. Note that even if there are several copies of β_2 in region 0, the previously mentioned antiport rule can only be applied once because there is only one copy of object **no** in region $n+1$.

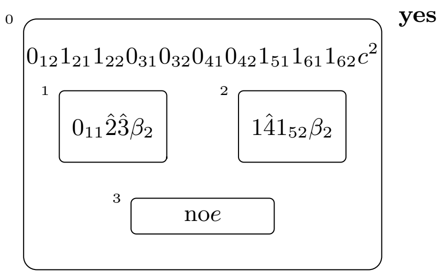
0
yes
$0_{12}1_{21}1_{22}0_{31}0_{32}0_{41}0_{42}1_{51}1_{61}1_{62}c^2$
1
$0_{11}\hat{2}\hat{3}\beta_2$
2
$1\hat{4}1_{52}\beta_2$
3
no$e$

Figure 21. C_9: All clauses were satisfied, hence **yes** is sent to the environment and the computation halts

Step 10: Lastly, if the not all the clauses are satisfied, **no** will be released to the environment and the computation will halt.

We will now move on to analyzing the communication resources used at each phase.

a. The setup phase uses one communication step. In this communication step, each object B is communicated to a region $q (1 \leq q \leq n)$. Since the objects B represent the clauses, we know that there are n number of objects B hence, n number of communication rules are used and n number of energy objects are consumed.

b. For the validation and output phase:

- The maximum number of communication steps (5) will occur if the boolean formula is not satisfiable because releasing **no** to the environment takes one more step than releasing **yes**. The rules $(\beta_2 e, in; \mathbf{no}e, out)$ and $(\mathbf{no}e, out)$ will be used in succeeding steps if the answer is no, in contrast to $(\mathbf{yes}e^{n+2}, out)$, which will take just one step.
- The maximum number of communication rules used is $4n+7$ and this happens if the answer is **no**. Again this is because releasing **no** to the environment uses one more rule than releasing **yes** to the environment.
- In contrast to the first two communication cost measures, the amount of energy objects used is maximum when the answer is **yes**. A total of $7n+7$ energy objects are consumed throughout a computation.

Summing up the computations, we have proven that $3SAT \in NF_{EPC}ComNRW = (6, 4n+7, 7n+7)$.

6. CONCLUSION

Based on the study we conducted, we present here the comparisons of the communication resources consumed in solving VCP and 3SAT using Evolution-Communications P systems with energy in CEM, CPE, and EPC mode.

	CPE	**CEM**	**EPC**
ComN	5	6	9
ComR	$\lvert V_G\rvert+3k+3$	$\lvert V_G\rvert+3k+6$	$\lvert V_G\rvert+3k+16$
ComW	$2\lvert E_G\rvert+\lvert V_G\rvert+k+4$	$3\lvert E_G\rvert+\lvert V_G\rvert+k+5$	$3\lvert E_G\rvert+\lvert V_G\rvert+k+15$
membranes	2	4	4

Table 1. Comparison of 3 modes of ECPe on Solving VCP

	CPE	**CEM**	**EPC**
ComN	4	5	6
ComR	$2n+1$	$2n+3$	$4n+7$
ComW	$4n+3$	$3n+1$	$7n+7$
membranes	n + 1	n+2	n+2

Table 2. Comparison of 3 modes of ECPe on Solving 3SAT

In Table 1 and Table 2, we compared the $ComN, ComR, ComW$ and the number of membranes used in the three modes of ECPe systems namely CPE, CEM, and EPC. We can see that ECPe in CPE performs the least number of communication steps while ECPe in EPC performs the most. This is more obviously seen when solving VCP which requires a constant number of membranes than when solving 3SAT of which the number of membranes required is dependent on the number of clauses.

When talking about the number of communication rules, CEM and CPE mode does not differ greatly when solving both VCP and 3SAT, whereas we see a relatively greater number of communication rules used in EPC mode. In the matter of energy consumption, the ECPe systems in CPE mode use linearly less energy than CEM and EPC mode in both problems. ECPe systems with priority on evolution use a constant number of energy less than CEM when solving VCP, while uses linearly less energy when solving 3SAT.

Aside from the communication complexity, we also look into the number of membranes needed to solve the Vertex Cover and 3-Satisfiability problem. It can be seen that VCP and 3SAT can be solved using the same number of membranes for CEM and EPC mode. However, ECPe with priority on communication uses less membranes than the two other modes.

We now ask: can we solve VCP and 3SAT using constructs of ECPe systems under different modes requiring less amount of communication steps, rules, and energy? Are there certain characteristics of an ECPe system that may improve the utilization of the priority of evolution in ECP mode? of communication in CPE? Can we create an ECPe that utilizes a constant number of membranes that solves 3SAT, perhaps with a different encoding? Under what conditions, if such exist, will it be better to use an ECPe system in EPC mode over the two other modes in terms of communication complexity? in CPE? in CEM? Additionally, there has been recent researches, such in [13], [12], and [6], which concurs to the concept of time-free variants of P systems wherein rules are not necessarily completed in a single step. Can we construct time-free ECP systems with and without energy, and how will those systems behave?

For future works, we are particularly interested in how the communication complexity measures given in [1] can be applied to analyze confluent solutions of NP-Complete problems using active membranes (as for example, solutions presented in [7], [11] and [8]). Additionally, we are interested in determining if we can define a deterministic confluent solution to hard problems.

7. REFERENCES

[1] Henry Adorna, Gheorghe Păun, and Mario Jesus de Pérez-Jiménez. On Communication Complexity in Evolution-Communication P systems. *Romanian Journal of Information Science and Technology*, 13(2):113–130, 2010.

[2] Matteo Cavaliere. Evolution-Communication P systems. In *Membrane Computing*, pages 134–145. Springer, 2003.

[3] Mario Jesus de Pérez-Jiménez. A Computational Complexity Theory in Membrane Computing. *Workshop on Membrane Computing*, pages 125–148, 2009.

[4] Antonio E.Porreca, Giancarlo Mauri, and Claudio Zandron. Non-confluence in Divisionless P systems with Active Membranes. *Theoretical Computer Science*, 411(6):878–887, 2010.

[5] Nestine Hope S. Hernandez, Richelle Ann B. Juayong, and Henry N. Adorna. On Communication Complexity of Some Hard Problems in ECPe systems. In *Membrane Computing*. Springer Berlin Heidelberg, 2014.

[6] Xiangrong Liu, Ziming Li, Juan Suo, Ying Ju, Juan Liu, and Xiangxiang Zeng. Solving Multidimensional 0-1 Knapsack Problem with Time-free Tissue P systems. *Journal of Applied Mathematics*, 2014, 2014.

[7] Chun Lu and Xingyi Zhang. Solving Vertex Cover Problem by Means of Tissue P systems with Cell Separation. *International Journal of Computers Communications & Control*, 5(4):540–550, 2010.

[8] Linqiang Pan and Artiom Alhazov. Solving HPP and SAT by P systems with Active Membranes and Separation Rules. *Acta Informatica*, 43(2):131–145, 2006.

[9] Andrei Păun. On p systems with active membranes. In *Unconventional Models of Computation, UMC'2K*, pages 187–201. Springer London, 2001.

[10] Gheorghe Păun. Introduction to Membrane Computing. In *Applications of Membrane Computing*, pages 1–42. Springer, 2006.

[11] Gheorghe Păun, Mario Jesus de Pérez-Jiménez, and Agustın Riscos-Núnez. Tissue P systems with Cell Division. *International Journal of Computers, Communications & Control*, 3(3):295–303, 2008.

[12] Tao Song, Luis F. Macías-Ramos, Linqiang Pan, and Mario Jesus de Pérez-Jiménez. Time-free Solution to SAT Problem using P systems with Active Membranes. *Theoretical Computer Science*, 529:61–68, 2014.

[13] Tao Song, Xun Wang, and Hongjiang Zheng. Time-free Solution to Hamilton Path Problems using P systems with D-division. *Journal of Applied Mathematics*, 2013, 2013.

[14] Tao Song, Hongjiang Zheng, and Juanjuan He. Solving Vertex Cover Problem by Tissue P Systems with Cell Division. *Applied Mathematics and Information Science*, 8(1):333–337, 2014.

[15] Xinyi Zhang, Linqiang Pan, and Andrei Păun. On the Universality of Axon P Systems. *IEEE Transactions on Neural Networks and Learning Systems*, in press.