Chemical evolution among artificial proto-cells

Yasuhiro Suzuki and Hiroshi Tanaka
Bio-Informatics,
Medical Research Institute,
Tokyo Medical and Dental University
Yushima 1-5-45, Bunkyo, Tokyo 113 JAPAN

Abstract

We develop an Artificial Cell System (*ACS*) based on an abstract chemical system. *ACS* consists of a multiset of symbols, a set of rewriting rules (*reaction rules*) and membranes. Throughout simulations, we find that chemical evolution like behavior emerges and cells evolve to a structure consisting of several cell-like membranes. We investigate the correlation among the type of reaction rules (rewriting rules), characteristic of a membrane and the evolution of cells, and then find the characteristics of a membrane effects on its evolution and obtains a parameter to describe the correlation. Furthermore, we introduce a genetic method to the system, and we attempt to apply it to Genetic Programming.

Introduction

A membrane is an important structure for living systems. It distinguishes "self" from its environment and hierarchical structures inside the system (like cells, organs and so on) are composed by membranes. Membranes change their structure dynamically and constitute a system. We are interested in their dynamical structure in terms of computation.

A membrane is composed of "chemical compounds (denoted by symbols)" which are generated through chemical reactions in the cell. In each cell there is some chemical compounds and these chemical compounds interact with each other according to the rewriting rule (*reaction rules*).

Based on the principles outlined above, we develop an "*Artificial Cell System*"(*ACS*). It consists of a multiset of symbols, a set of rewriting rules (reaction rules) and membranes. *ACS* consists of an abstract chemical system, "*Abstract Rewriting System on MultiSets (ARMS)*" that is a multiset transform system,(Suzuki 1996). It consists of a multiset of symbols and a set of rewriting rules. Although not many alife researches have tackled this topic previously (McMullin and Varela 1997), (Mizaro, Moreno, et. al. 1999), the focus of these researches is on the formation of a membrane. The aim of this study is to investigate the role of membrane in terms computation, thus we do not treat its formation.

Since *ARMS* is not a strings rewriting system but a (multi)-*set* rewriting system, it can deal with many degrees of freedom such as a *chemical solution* and the *concentration* of *chemical compounds*. We confirmed that it could simulate phenomena that we find in real bio-chemical reactions such as the "*Belouzov-Zhabotinsky reaction (BZ-reaction) (Suzuki 1998)*" the *BZ-reaction* is a spontaneous chemical oscillation, and is considered as the basic mechanism of bio-chemical systems.

ARMS

We will introduce the multiset rewriting system, "*Abstract Rewriting system on MultiSets*" in this section. Intuitively, *ARMS* is like a chemical solution in which *molecules* floating on it can interact with each other according to reaction rules. Technically, a chemical solution is a finite multi-set of elements denoted by $A^k = \{a, b, \ldots, \}$; these elements correspond to *molecules*. Reaction rules that act on the molecules are specified in *ARMS* by rewriting rules. As to the intuitive meaning of *ARMS*, we refer to the study of chemical abstract machines (Berry 1992). In fact, this system can be thought of as an underlying "*algorithmic chemistry* (Fontana 1994)."

Let A be an *alphabet* (a finite set of abstract symbols). The set of all strings over A is denoted by A^*; the empty string is denoted by λ. (Thus, A^* is the free monoid generated by A under the operation of concatenation, with identity λ.) The length of a string $w \in A^*$ is denoted by $|w|$.

A *rewriting rule* over A is a pair of strings (u, v), $u, v \in A^*$. We write such a rule in the form $u \rightarrow v$. Note that u and v can also be empty. A *rewriting system* is a pair (A, R), where A is an alphabet and R is a finite set of rewriting rules over A.

With respect to a rewriting system $\gamma = (A, R)$ we define over A^* a relation $\Longrightarrow$ as follows: $x \Longrightarrow y$ iff $x = x_1 u x_2$ and $y = x_1 v x_2$, for some $x_1, x_2 \in A^*$ and $u \rightarrow v \in R$. The reflexive and transitive closure of this relation is denoted by $\Longrightarrow^*$. A string $x \in A^*$ for which there is no string $y \in A^*$ such that $x \Longrightarrow y$ is said to be an *dead* one (in other words, from a dead string no string can be derived by means of the rewriting rules).

From now on, we work with an alphabet A whose elements are called *objects*; the alphabet itself is called a *set of objects*.

A *multiset* over a set of objects A is a mapping $M : A \longrightarrow \mathbf{N}$, where $\mathbf{N}$ is the set of natural numbers, 0, 1, 2,.... The number $M(a)$, for $a \in A$, is the *multiplicity* of object a in the multiset M. Note that we do not accept here an infinite multiplicity. The set $\{a \in A \mid M(a) > 0\}$ is denoted by $supp(M)$ and is called the *support* of M. The number $\sum_{a \in A} M(a)$ is denoted by $weight(M)$ and is called the *weight* of M.

We denote by $A^{\#}$ the set of all multisets over A, including the empty multiset, $\emptyset$, defined by $\emptyset(a) = 0$ for all $a \in A$.

A multiset $M : A \longrightarrow \mathbf{N}$, for $A = \{a_1, \ldots, a_n\}$, can be naturally represented by the string $a_1^{M(a_1)} a_2^{M(a_2)} \ldots a_n^{M(a_n)}$ and by any other permutation of this string. Conversely, with any string w over A we can associate a multiset: denote by $|w|_{a_i}$ the number of occurrences of object a_i in w, $1 \leq i \leq n$; then, the multiset associated with w, denoted by M_w, is defined by $M_w(a_i) = |w|_{a_i}, 1 \leq i \leq n$.

The union of two multisets $M_1, M_2 : A \longrightarrow \mathbf{N}$ is the multiset $(M_1 \cup M_2) : A \longrightarrow \mathbf{N}$ defined by $(M_1 \cup M_2)(a) = M_1(a) + M_2(a)$, for all $a \in A$. If $M_1(a) \leq M_2(a)$ for all $a \in A$, then we say that multiset M_1 is included in multiset M_2 and we write $M_1 \subseteq M_2$. In such a case, we define the multiset difference $M_1 - M_2$ by $(M_2 - M_1)(a) = M_2(a) - M_1(a)$, for all $a \in A$. (Note that when M_1 is not included in M_2, the difference is not defined).

A rewriting rule such as

$$a \rightarrow a \ldots b,$$

is called a *heating rule* and denoted as $r_{\Delta > 0}$; it is intended to contribute to the stirring solution. It breaks up a complex *molecule* into smaller ones: *ions*. On the other hand, a rule such as

$$a \ldots c \rightarrow b,$$

is called a *cooling rule* and denoted as $r_{\Delta < 0}$; it rebuilds *molecules* from smaller ones. In this paper, reversible reactions, i.e., $S \rightleftharpoons T$, are not considered. We shall not formally introduce the refinement of *ions* and *molecules* though we use refinement informally to help intuition (on both types of rules we refer to (Berry 1992)).

A *multiset rewriting rule* (we also use to say, *evolution rule*) over a set A of objects is a pair (M_1, M_2), of elements in $A^{\#}$ (which can be represented as a rewriting rule $w_1 \rightarrow w_2$, for two strings $w_1, w_2 \in A^*$ such that $M_{w_1} = M_1$ and $M_{w_2} = M_2$). We use to represent such a rule in the form $M_1 \rightarrow M_2$.

An *abstract rewriting system on multisets* (in short, an *ARMS*) is a pair

$$\Gamma = (A, R)$$

where:

(1) A is a set of objects;

(2) R is a finite set of multiset evolution rules over A;

With respect to an *ARMS* Γ, we can define over $A^{\#}$ a relation: $(\Longrightarrow)$: for $M, M' \in A^{\#}$ we write $M \Longrightarrow M'$ iff

$$M' = (M - (M_1 \cup \ldots \cup M_k)) \cup (M_1' \cup \ldots \cup M_k',)$$

for some $M_i \rightarrow M_i' \in R, 1 \leq i \leq k, k \geq 1$, and there is no rule $M_s \rightarrow M_s' \in R$ such that $M_s \subseteq (M - (M_1 \cup \ldots \cup M_k))$; at most one of the multisets $M_i, 1 \leq i \leq k$, may be empty.

With respect to an *ARMS* $\Gamma = (A, R)$ we can define various types of multisets:

- A multiset $M \in A^{\#}$ is *dead* if there is no $M' \in A^{\#}$ such that $M \Longrightarrow M'$ (this is equivalent to the fact that there is no rule $M_1 \rightarrow M_2 \in R$ such that $M_1 \subseteq M$).
- A multiset $M \in A^{\#}$ is *initial* if there is no $M' \in A^{\#}$ such that $M' \Longrightarrow M$.

How *ARMS* works

Example In this example, an *ARMS* is defined as follows;

$$\begin{array}{lcl} \Gamma & = & (A, R), \\ A & = & \{a, b, c, d, e, f\}, \\ R & = & \{a, a, a \rightarrow c : r_1, b \rightarrow d : r_2, c \rightarrow e : r_3, \\ & & d \rightarrow f, f : r_4, a \rightarrow a, b, b, a : r_5, f \rightarrow h : r_6, \}. \end{array}$$

The set of the rewriting rules, R is $\{r_1, r_2, r_3, r_4, r_5, r_6\}$. We assume the maximal multiset size is 4 and the initial state is given by $\{a, a, b, a\}$. In *ARMS*, rewriting rules are applied in parallel. When

$\{a,b,a,a\}$	$\subseteq a, a, a, a, b$
$\downarrow$	(the left hand side of r_1, r_2, r_5)
$\{c,d\}$	$\subseteq c, d$ (the left hand side of r_3, r_4)
$\downarrow$	
$\{e,f,f\}$	$\subseteq f$ (the left hand side of r_6)
$\downarrow$	
$\{e,h,h\}$	There are no rule to apply, it reaches the *death* state

Figure 1: Example of rewriting steps of $ARMS$

there are more than two applicable-rules, then one rule is selected randomly. Figure 1 illustrates an example of rewriting steps of the calculation from the initial state.

At the first step, the left hand side of rule of r_1, r_2 and r_5 are included in the initial state. In the next step, r_3 and r_4 are applied in parallel and $\{c,d\}$ is rewritten into $\{e,f,f\}$. In step 3, by using r_6, $\{e,f,f\}$ is transformed into $\{e,h,h\}$. There are no rules that can transform the multiset any further so, the multiset is in a *dead* state.

Artificial Cell System

In this section, we introduce the basic structural ingredients of ARMS, membrane structures and how ACS works.

The membrane structure (MS)

To describe the membrane and its structure in $ARMS$, we first define the language MS over the alphabet $\{[,]\}$ whose strings are recurrently defined as follows:

(1) $[,] \in MS$

(2) if $\mu_1, ..., \mu_n \in MS$, n ≥ 1, then $[\mu_1, ..., \mu_n] \in MS$;

(3) there is nothing else in MS.

The most outer membrane M_0 corresponds to a container such as a test tube or reactor and it never dissolves.

Consider now the following relation on MS: for $x, y \in MS$ we write $x \sim y$ if and only if we can write the two strings in the form $x = [_1...[_2...]_2[_3...]_3...]_1, y = [_1...[_3...]_3[_2...]_2...]$, i.e., if and only if two pairs of parentheses which are not contained in one other can be interchanged, together with their contents. We also denote by $\sim$ the reflexive and transitive closures of the relation $\sim$. This is clearly an equivalence relation. We denote by $\overline{MS}$ the set of equivalence classes of MS with respect to this relation. The elements of $\overline{MS}$ are called *membrane structures.*

It is easy to see that the parentheses [,] appearing in a membrane structure are matching correctly in the usual sense. Conversely, any string of correctly matching pairs of parentheses [,], with a matching pair at the ends, corresponds to a membrane structure.

Each matching pair of parentheses [,] appearing in a membrane structure is called a *membrane*. The number of membranes in a membrane structure μ is called the *degree* of μ and is denoted by $\deg(\mu)$. The external membrane of a membrane structure μ is called the *vessel;* membrane of μ. When a membrane which appears in $\mu \in \overline{MS}$ has the form [ ] and no other membranes appear inside the two parentheses then it is called an *elementary* membrane.

ACS and ACSE

We will define two types of ACS;

(1) ACS and

(2) ACS with an Elementary membrane (ACSE).

ACSE is different only in the way of dissolving and dividing from ACS.

Descriptions of ACS

A transition ACS is a construct

$$\Gamma = (A, \mu, M_1, ..., M_n, R, MC, \delta, \sigma),$$

where:

(1) A is a set of objects;

(2) μ is a membrane structure (it can be changed throughout a computation);

(3) $M_1, ..., M_n$, are multisets associated with the regions 1,2, ... n of μ;

(4) R is a finite set of multiset evolution rules over A.

(5) MC is a set of membrane compounds;

(6) δ is the threshold value of dissolving a membrane;

(7) σ is the threshold value of dividing a membrane;

μ is a membrane structure of degree n, n ≥ 1, with the membranes labeled in a one-to-one manner, for instance, with the numbers from 1 to n. In this way, also the regions of μ are identified by the numbers from 1 to n.

Rewriting rules are applied in following manner:

(1) The same rules are applied to every membrane. There are no rules specific to a membrane.

(2) All the rules are applied in parallel. In every step, all the rules are applied to all objects in every membrane that can be applied. If there are more than two rules that can apply to an object then one rule is selected randomly.

(3) If a membrane dissolves, then all the objects in its region are left free in the region immediately above it.

(4) All objects and membranes not specified in a rule and which do not evolve are passed unchanged to the next step.

Rewriting rule R is a finite set of multiset rewriting rules over A. Both the left and the right side of a rule are obtained by sampling with replacement of symbols. A set of reaction rules is constructed as the overall permutation of both sides of the rules.

Input and Output Chemical compounds are supported from outside of the system to M_0 and some compounds are exhausted from M_0. All chemical compounds are transformed among cells, a randomly selected chemical compound is transformed into the membrane just above or below it. Although a membrane does not allow specificity of transport across the membrane, a cell can control its chemical environment by chemical reaction.

Dissolving and dividing a membrane of ACS A membrane is composed of a "membrane compound" which is in fact a symbol. To maintain a membrane, it needs to have a certain minimal volume. A membrane disappears if the volume of membrane compounds decreases below the needed volume to maintain the membrane. Dissolving the membrane is defined as follows:

$$[_h a, ...[_i b, ...]_i]_h \rightarrow [_h a, b, ...]_h,$$

where the ellipsis {...} illustrates chemical compounds inside the membrane. Dissolving takes place when

$$\frac{|w_i|_{MC}}{|M_i|} < \delta$$

where δ is a threshold value for dissolving the membrane. All chemical compounds in its region are then set free and they are merged into the region immediately above it.

On the other hand, when the volume of membrane compounds increases to a certain extent, then a membrane is divided. Dividing a membrane is realized by dividing it in multisets random sizes. The frequency at which a membrane is divided is decided in proportion to its size. As the size of a multiset becomes larger, the cell is divided more frequently. Technically, this is defined as follows;

$$[_h a, b, ...]_h \rightarrow [_h a, ...[_i b, ...]_i]_h$$

Dividing takes place when

$$\frac{|w_h|_{MC}}{|M_h|} > \sigma$$

where σ is a threshold for dividing the membrane. All chemical compounds in its region are then set free and they are separated randomly by new membranes.

Description of ACSE

ACSE is different only in the way of dividing and dissolving cells from ACS. Dissolving the membrane is defined as follows:

$$[_h a, b, ...]_h \rightarrow [_0 a, b, ...,]_0$$

Dissolving takes place when

$$\frac{|w_h|_{MC}}{|M_h} < \delta$$

where δ is a threshold value for dissolving the membrane. All chemical compounds in its region are then set free and they are merged into the region of M_0.

Dividing is defined as follows;

$$[_h a, b, ...]_h \rightarrow [_h a, ...]_h [_i b, ...]_i.$$

Dividing takes place when

$$\frac{|w_h|_{MC}}{|M_h|} > \sigma,$$

where σ is a threshold for dividing the membrane. All chemical compounds in its region are then set free and they are separated randomly in the old and new membranes. Hence, in ACSE, a structured cell such as $[a, b[c, [d, e]]]$ does not appear.

A Cell like Chemoton Because the components of a membrane diminish with the lapse of a certain time, a cell has to generate the components to maintain the membrane through chemical reactions in the cell in ACS and ACSE. Hence, all survived cells in ACS and ACSE become cells like a *chemoton*(Gánti 1975). We confirmed this through simulations.

Evolution of Cells When a cell grows and the cell exceeds the threshold value for dividing, it divides into parts of random sizes. This can be seen as a kind of *mutation*. If a divided cell does not have any membrane compounds, it must disappear soon.

Furthermore, to maintain the membrane through chemical reactions inside the cell can be seen as *natural selection*. If cells can not maintain the membrane compounds, it must disappear soon.

Thus, both dividing membranes and dissolving membranes produce evolutionary dynamics. These correspondences are summarized into;

Natural Selection	Dissolving a membrane,
Mutation	Dividing a cell into parts of random size.

Behavior of *ACSE and ACS*

We will show some experimental results of ACSE and ACS in this section.

ACSE The evolution of elementary cells can be regarded as an approximate model of the chemical evolution in the origin of life.

The following *ACSE* was simulated;

$\Gamma = (A = \{a, b, c\}, \mu = \{ [,]_0, ...[,]_{100}\} M_0 = \{[a^{10}, b^{10}, c^{10}]^{100}\}, R, MC = \{b\}, \delta = 0.4, \sigma = 0.2)$,

where:

(1) R, the length of the left- or right-hand-side of a rule is between one and three. Both sides of the rules are obtained by sampling with replacement of the three symbols a, b and c;

(2) Membrane structures are assumed to be $(\mu = \{[_1]_1 ... [_{100}]_{100}\})$.

Through the simulation we discovered that the strength of a membrane affects the behavior of cells. The strength of a membrane is defined as the frequency of decreasing membrane compounds.

When a membrane is strong When a membrane is strong, the most stable cell consists of only one membrane, cells of this type become " mother" cells and they produce "daughter" cells.

In order to display a state of a cell we transform the state of a cell to a number by using the transformation function; $f(M(a), M(b), M(c)) = 10^2 \times M(a) + 10^1 \times M(b) + 10^0 \times M(c)$. For example, the state $\{a, a, b, c, c\}$ is transformed into $10^2 \times 2 + 10^1 \times 1 + 10^0 \times 2 = 212$.

The figure 2 illustrates the evolution of cells when a membrane is strong. The cells that are close to the horizontal axis are mother cells. Some daughter cells depart from the group and evolve different types of cells, even though almost all cells are in the group. In this case, dissolving a membrane compound takes place per 100 steps.

The figure 3 is focused to the mother cells. At first there are about ten groups, and some of them become extinct: after 200 steps there remain about four groups.

When a membrane is weak Figure 4 illustrates the case when a membrane is weak, a membrane dissolves every 3 steps. In this case, the system can not form a group of mother cells such as in the previous case. Since the group of cells drifts to more stable cells, the cells grow larger. Even if a large cell divides into parts of random sizes, the probability of including

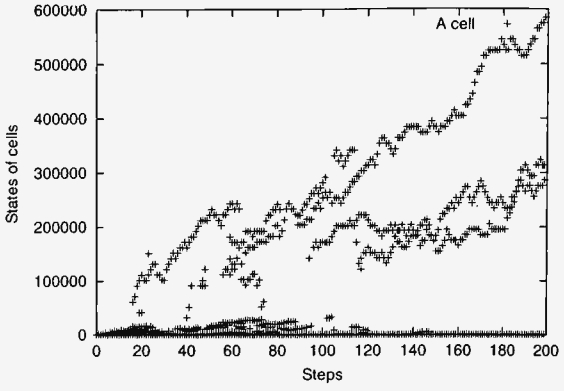

Figure 2: When a membrane is strong. The lines illustrate the regions where cells exist and points correspond to the state of cells.

enough membrane compounds to maintain its membrane are larger than a small cell.

We believe that this behaviors of evolution is similar to the evolution of viruses (Tanaka 1999). The settings of this simulation are so rough, however, that the possibility remains open that chemical evolution in origin of life is similar to virus evolution. This will be addressed in future research.

The correlation between the behavior of an ACS and the characteristics of rewriting rules

Description of a simulation Next, the following *ACS* was simulated;

$\Gamma = (A = \{a, b, c\}, \mu = \{/\ 0\}, M_0 = \{a^{10}, b^{10}, c^{10}\}, R, MC = \{b\}, \delta = 0.4, \sigma = 0.2)$,

where:

(1) R, the length of the left- or right-hand-side of a rule is between one and three. Both sides of the rules are obtained by sampling with replacement of the three symbols a, b and c;

(2) Membrane structures are not assumed.

λ_e parameter In order to investigate the correlation between the rewriting rule and the behavior of the model, we will introduce the λ_e parameter (Suzuki 1998). This parameter is introduced as an order parameter of *ARMS*.

Let us define the λ_e parameter as follows:

$$\lambda_e = \frac{\Sigma r_{\Delta S > 0}}{1 + (\Sigma r_{\Delta S < 0} - 1)} \quad (1)$$

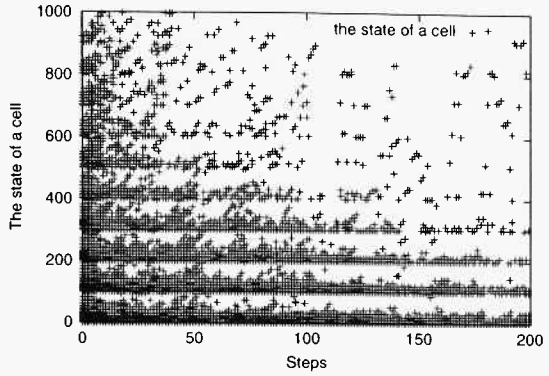

Figure 3: Evolution of mother cells. The points correspond to cells.

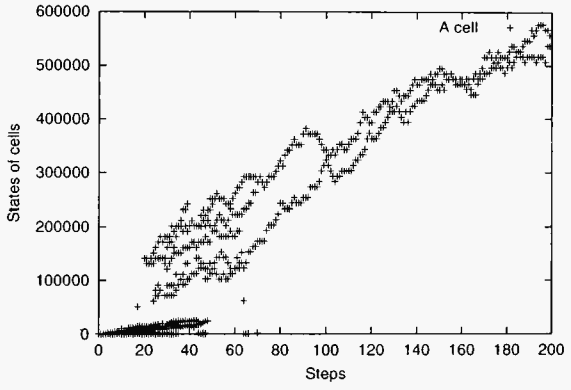

Figure 4: When a membrane is weak

where $\Sigma r_{\Delta S>0}$ corresponds to the number of *heating rules*, and $\Sigma r_{\Delta S<0}$ to the number of *cooling rules*. This parameter is well-defined when the number of rules is greater than 1.

When the *ARMS* only uses rules of the type $r_{\Delta S<0}$, λ_e is equal to 0.0. On the contrary, if the *ARMS* uses rules of the type $r_{\Delta S>0}$ and $r_{\Delta S<0}$ with the same frequency, λ_e is equal to 1.0. Finally, when the *ARMS* only uses rules of the type $r_{\Delta S>0}$, λ_e is greater than 1.0.

λ_e indicates the degree of reproduction in a cell. When λ_e close to 0.0, the degree is quite low and as λ_e is getting larger than 1.0, the degree becomes higher.

λ_e parameter and system's behavior The behavior of *ACS* is classified into four classes by this parameter as follows;

- Type I: A cell does not evolve and disappears;
- Type II: The period of dividing membranes and dissolving membranes appears cyclically.
- Type III: A cell evolves to a complex, hierarchically structured cell;
- Type IV: All chemical compounds inside a cell increase rapidly. However, cells hardly divide;

where each λ_e value is not important very much. Although they change in the different environments, these four classes are unchanged.

Type I (λ_e close to 0.0) When λ_e is close to 0.0 the *cooling rule* is mainly used, and cells will hardly grow up. Thus membranes will hardly be divided (figure 5).

Type II (λ_e in between 0.5 and $\sim$ 1.0) When λ_e is in between $0.5 \sim 1.0$, membranes are more likely to be divided than when λ_e is close to 0.0. But, since *cooling rules* are likely to be used, the membrane compounds do not increase very much. Thus, when a cell evolves to a certain size the membrane compounds of each cell decrease and they are dissolved (figure 6). Thus there emerges a *cell cycle* like behavior.

step	state
0.	$[a^{10}, b^{10}, c^{10}]$
1.	$[a^{9}, b^{5}, c^{10}]$
2.	$[a^{10}, b^{2}, c^{7}]$
3.	$[a^{11}, b^{3}, c^{7}]$
4.	$[a^{6}, b^{4}, c^{5}]$
	.....
10.	$[a^{1}, b^{4}, c^{2}]$
	.....
16.	$[a^{1}, b^{4}]$

Figure 5: An example of state transition of *ACS*: Type I (λ_e close to 0.0)

Type III (λ_e in between 1.05 $\sim$ 2.33) When λ_e exceeds 1.0 and the *heating rule* is likely to be used, a cell grows up easier and the frequency of cell division becomes high. Thus a cell evolves into a large complex cell (figure 7).

Type IV (λ_e more than 2.5) When the λ_e parameter becomes much larger than 1.0, membranes will hardly be divided, because the number of compounds of all kinds in the cell increase and it is difficult to specifically increase the number of only membrane compounds. Consequently, a cell will hardly divide (figure 7).

step	state
0.	$[a^{10}, b^{10}, c^{10}]$
1.	$[a^{6}, b^{9}, c^{9}]$
	.....
91.	$[a^{4}, b^{4}, c^{1}]$
92.	$[[b^{2}]\ [b^{3}, c^{1}]]$
93.	$[[b^{2}]\ [b^{1}, c^{1}]]$
94.	$[a^{6}, b^{1}, c^{1}]$
	.....
114.	$[a^{2}, b^{5}]$
115.	$[[a^{1}, b^{2}][a^{1}, b^{3}]]$
116.	$[[\ a^{1}, b^{2}][b^{2}][a^{1}, b^{1}]]$
	.....
140.	$[[[\ a^{1}, b^{1}\ ][b^{1}, c^{1}]][[b^{2}\ ][a^{1}, b^{1}]][b^{1}, c^{1}]]$
141.	$[[a^{3}, c^{3}][a^{3}, c^{2}][a^{1}, c^{2}]$
142.	$[\ a^{6}, c^{5}\ ]$

Figure 6: An example of state transition of ACS: Type II (λ_e in between 0.5 and $\sim$ 1.0)

step	state
0.	$[\ a^{10}, b^{10}, c^{10}]$
1.	$[\ a^{10}, b^{10}, c^{9}]$
	.....
41.	$[a^{2}, b^{7}, c^{5}]$
42.	$[[b^{2}]\ [b^{6}, c^{1}]]$
43.	$[[b^{4}]\ [b^{3}, c^{1}]]$
44.	$[[b^{4}]\ [b^{2}\ ][b^{2}, c^{1}]]$
45.	$[[b^{4}, c^{2}][b^{2}, c^{2}][b^{1}, c^{3}]]$
46.	$[[[b^{3}, c^{1}][b^{1}, c^{1}]][b^{2}, c^{2}][b^{1}, c^{3}]]$
47.	$[[[[b^{2}][a^{1}, b^{1}]][b^{1}, c^{1}]][a^{1}, b^{3}, c^{2}]]$ $[a^{1}, b^{2}, c^{3}\ ]]$
140.	$[[[a^{1}, b^{1}][b^{1}, c^{1}]][[b^{2}][a^{1}, b^{1}]][b^{1}, c^{1}]]$
214.	$[[[[a^{16}, b^{3}, c^{2}][a^{5}, b^{5}, c^{1}][a^{3}, b^{2}]]$ $[a^{4}, b^{3}]][[[[b^{2}][b^{2}][a^{1}, b^{1}]][a^{3}, b^{1}]]$ $[a^{3}, b^{1}][b^{4}][a^{3}, b^{1}]]]$

Figure 7: An example of state transition of ACS: Type III (λ_e in between 1.05 $\sim$ 2.33)

We believe that the division in four classes describes the basic behavior of the system. However, when δ or σ are changed the λ_e value that corresponds to each type is also changed. Thus a deeper investigation is needed with respect to the correlation between δ, σ and λ_e more precisely.

Genetic ACS (GACS)

Since a rewriting rule promotes a reaction, it can be regarded as an enzyme. Here we extend ACS with evolutionary mechanism We called the system Genetic ACS (GACS).

Descriptions of $GACS$

A transition in $GACS$ is a construct

step	state
0.	$[a^{10}, b^{10}, c^{10}]$
1.	$[a^{12}, b^{5}, c^{9}]$
2.	$[a^{14}, b^{4}, c^{10}]$
3.	$[a^{13}, b^{6}, c^{11}]$
4.	$[a^{14}, b^{8}, c^{12}]$
5.	$[a^{9}, b^{9}, c^{10}]$
6.	$[a^{7}, b^{10}, c^{1}0]$
	.....
87.	$[a^{17}, b^{12}, c^{21}]$
	.....
147.	$[a^{14}, b^{20}, c^{29}]$
	.....
242	$[a^{17}, b^{50}, c^{44}]$
	.....
300.	$[a^{3}, b^{56}, c^{58}]$

Figure 8: An example of state transition of ACS: Type IV (λ_e more than 2.5)

$$\Gamma = (A, \mu, M_1, ..., M_n, R, \delta, \sigma),$$

where:

(1) A is a set of objects;

(2) μ is a membrane structure (it can be changed throughout a computation);

(3) $M_1, ..., M_n$, are multisets associated with the regions 1,2, ... n of μ;

(4) R is a finite set of multiset evolution rules over A.

(5) δ is the threshold of dissolving membrane;

(6) σ is the threshold of dividing membrane.

The way of applying rewriting rules, the way of dissolving and dividing and input and output are the same as ACS and ACSE.

An enzyme We denote a set of reaction rules as follows;

	a	b	c
a	x_{aa}	x_{ab}	x_{ac}
b	x_{ba}	x_{bb}	x_{bc}
c	x_{ca}	x_{cb}	x_{cc},

where x_{ij} means the number of compounds i which are transformed from j. For example, 2_{ab} means a rewriting rule, $b \rightarrow a, a$. We call the table a transformation table.

Transmission of an enzyme When a membrane is divided, the enzyme which is inside the membrane is copied and passed down to a new divided cell. At that time, a point mutation occurs only in the copied enzyme and it is passed down to the new cell. The enzyme remain in the old membrane as well as the new one. Point mutations occur every time a membrane divides. When a membrane is dissolved the enzyme which is inside the membrane loses its activity. A point mutation is a rewriting of the number of x_{ij}. Thus, it changes the number of transforming compounds to i. We assumed $x_{ij} \in \{0, 1, 2\}$. In ACS and ACSE the system have only one rewriting rule, however, in GACS, each membrane has a set of rules.

An experimental result of a GACS

We will show experimental results of GACS. At first, the following *GACS* was simulated;

$\Gamma = (A = \{a, b, c\}, \mu = \{/\ 0\} M_0 = \{a^{10}, b^{10}, c^{10}\}, R, MC = \{c\}, \delta = 0.4, \sigma = 0.2)$,

where the transformation table (R) is set as follows in the initial states;

	a	b	c
a	0_{aa}	0_{ab}	1_{ac}
b	1_{ba}	0_{bb}	0_{bc}
c	0_{ca}	1_{cb}	0_{cc}.

The productivity of membrane compounds p is defined as the ratio of the total number of non-membrane compounds to be produced to the total number of membrane compounds to be produced;

$$p = \frac{\Sigma_{j=a}^{j=b} x_{cj}}{\Sigma_{j=a}^{i=b} \Sigma_{j=a}^{j=c} x_{i,j}},$$

When $p = 0$ the enzyme does not produce any membrane compounds, when $p = 1$, it produces the same number of membrane compounds to the non-membrane compounds, and when $p > 1$, it produces more membrane compounds than non-membrane compounds. Figure 9 illustrates the time series of productivity, where the vertical axis illustrates the productivity, the horizontal axis illustrates the steps and each dot is an enzyme. It shows that at first almost enzymes evolve to $p > 1$. However, after 100 steps, the productivity of the enzymes decrease.

After 100 steps, both the number of cells and the sizes of cells increase exponentially. Furthermore, the structure of cells becomes complicated. Figure 10 illustrates the correlation between the number of cells, the size of cells and the number of steps, where each dot corresponds to a cell.

Figure 11 illustrates the internal nodes of the whole system. If we regard M_0 as the root and other cells

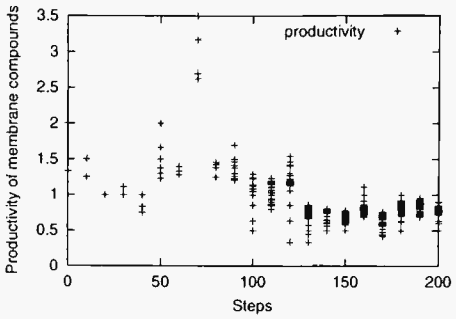

Figure 9: Productivity of enzymes.

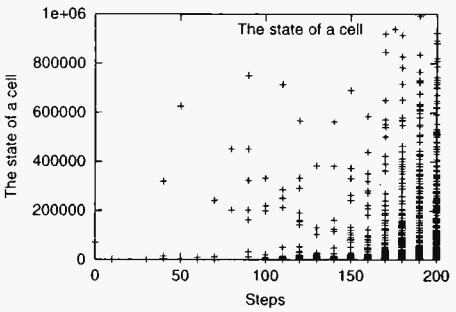

Figure 10: State transition of the system

as internal nodes and leaves, we can regard the whole system as a tree. In order to indicate the complexity of the tree, we use the number of internal nodes in the tree. Figure 11 illustrates that the number of internal nodes increases exponentially, after 150 steps.

It is interesting that when a cell grows into a hierarchical cell, the enzyme evolves to a low productivity one.

The reason of this behavior can be considered as follows; the enzyme whose productivity is high always suffers from mutations, because it promotes membrane division thus it generates more mutations than low productivity ones. If every cell is an elementary cell, an enzyme have to keep producing membrane compounds at a high rate. However, when the cell forms structure, it is not necessary one with high productivity, because, in a structured cell, if an inside cell dissolves, the cell that includes the dissolved cell obtains its membrane compounds.

Therefore, during evolution of a cell into a structured cell, the cell needs a high productivity enzyme. However, once it evolves a structured cell, high pro-

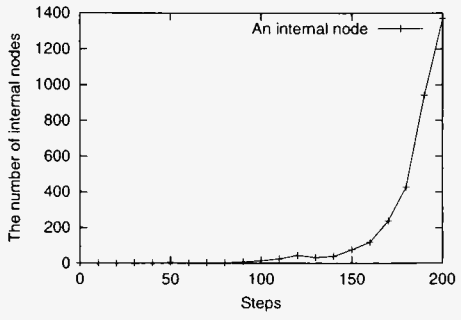

Figure 11: Internal nodes of the cell

ductivity enzymes are weeded out. This is the role of membranes in terms of computation.

Genetic Programming by using GACS

We attempted to generate a program by using a GACS. In the GACS, a program corresponds to an enzyme, thus we breed an enzyme which can solve a particular problem. We apply it to a simple problem *doubling*; calculate the double value of the number of a and b then show the result as the number of c ($c = 2(a+b)$).

Description of GACS A *GACS* is defined as follows;

A transition *GACS* is a construct

$\Gamma = (A = \{a, b, c\}, \mu = \{[_1]_1 \cdots [_{100}]_{100}\}, M_0 = \{[a^2, b^2, c^0]^{100}\}, R)$. In the initial state, all transformation tables R are

	a	b	c
a	0_{aa}	0_{ab}	1_{ac}
b	1_{ba}	0_{bb}	0_{bc}
c	0_{ca}	1_{cb}	0_{cc},

and one hundred of elementary cells are assumed inside M_0. No compounds are transformed among cells and no input and output are assumed. Although we performed simulation by using different type of cells in the initial state, the experimental results are same, thus we will address only when each cell is $[a^2, b^2, c^0]$ in the initial state.

Dissolving and dividing a membrane The way of dissolving and dividing are the same as in ACSE. After n rewriting steps, if the number of c is smaller than 7, the membrane is dissolved and the enzyme inside it loses its activity, for example,

$$[_h a^3, b^5, c^8]_h [_i a^4, b^5, c^1]_i \rightarrow [_h a^3, b^5, c^8]_h.$$

In the above example, the number of c inside the membrane i is smaller than 7, so the membrane is dissolved.

After n rewriting steps, if the number of c is larger than 9, the membrane is divided and a point mutation takes place in its enzyme. Furthermore, a new enzyme is passed down to a new cell. When a cell divided, the inside multiset of the divided cell and its parent cell are set to $\{a^2, b^2, c^0\}$ again, so they try to solve the problem again. The results in:

$$[_h a^3, b^5, c^1 0]_h \rightarrow [_h a^2, b^2, c^0]_h [_i a^2, b^2, c^0]_i$$

In the example, because the number of c inside the membrane h is larger than 9, the membrane h is divided and a new membrane i emerges. Then compounds which are inside both membranes i and h are set to $\{a^2, b^2, c^0\}$. In this GACS, cells continue to solve the problem.

The fitness of the GACS The fitness of an enzyme is defined as the number of steps to reach the solution. By using this fitness, good enzymes are there that can solve the problem within a smaller number of steps than the others are selected.

Experimental result At first, all enzymes are set to,

$$\begin{aligned} a &\rightarrow b, \\ b &\rightarrow c, \\ c &\rightarrow a. \end{aligned}$$

After 5000 rewriting steps, the enzymes that reach the solution within 8 steps are selected;

$$\begin{array}{lll} a \rightarrow a, c, & a \rightarrow a, b, c, c, \\ b \rightarrow a, b, c, c, & b \rightarrow a, b, c, c, \\ c \rightarrow c, c; & c \rightarrow c, c; \\ a \rightarrow a, c, c, \\ b \rightarrow b, \\ c \rightarrow c, c. \end{array}$$

For each rule, one hundred of elementary cells that are $[a^2, b^2, c^0]$ are set again and calculations performed again. Next, the enzymes that can solve the problem within 5 steps are selected. Then, there remains only one enzyme;

$$\begin{aligned} a &\rightarrow a, c, c, \\ b &\rightarrow c, c, \\ c &\rightarrow c, c. \end{aligned}$$

it is a solution of this simulation. In fact a solution of this problem is

$$\begin{aligned} a &\rightarrow c, c, \\ b &\rightarrow c, c, \\ c &\rightarrow c; \end{aligned}$$

Thus the survived enzyme evolved to a similar enzyme in the solution.

By using this method, we have attempted to treat the system as an artificial living system of computation. In the future we plan to create a GACS as an artificial living system of computation that can solve more complicated problems. In such a system, in order to obtain the result, we observe their output (behaviors) and change the condition, do not stop their computations. In other words, we steer them in the right direction by using the selection and mutation, and lead them to our settled goal. Although GACS may not fit to make optimizer, we believe this method can apply to design artificial living things such as robots.

Discussion

P systems

The above ACSes correspond to a class of P systems, a parallel molecular computing model proposed by G. Păun(Păun 1998) that is based on the processing of multisets of objects in cell-like membrane structures. A P system is a multiset transformation system. However, it is different from (Berry 1992), since it includes a "*membrane*" in its computing mechanism. In P systems *cells* are structured like living cells. The system itself is composed of several cells that are delimited from the neighboring cells.

The computing power of P systems is equal to that of a Turing machine, as proved in (Păun 1998), and an algorithm to compute the SAT problem in linear time (Păun 1999) was proposed. Various P systems have also been proposed and their mathematical properties have been investigated. On the computing power of ACSes that are treated in this paper is open problems.

Conclusion

It is obvious that living systems never do calculation only by using *pen* and *paper*. Thus, if we could abstract *computation* in terms of living systems then we might obtain a new computational world.

We have not found the new world yet, however, we have already obtained some *hints* throughout the observations on living systems, such as "parallelism" and "membranes." In a living system, every cell and every organ is acting in parallel and they are consisted of membranes.

To implement our system on a parallel computer and investigating its mathematical properties are our future work. We expect that these studies lead us to a new computing paradigm which goes beyond the Turing Machine based computing paradigm.

Acknowledgment

We thank for fruitful discussions with Dr. Steen Rasmussen, Prof. Dr. W. Banzaf gave us important comments and encouraged us, long discussions with P. Dittrich were useful. F. Pepper and SOMA research group gave us important suggestions. And the authors would like to express many thanks to Dr. Gheorghe Păun for his useful comments, discussions and mathematical refinements. This research is supported by Grants-in Aid for Scientific Research No.11837005 from the Ministry of Education, Science and Culture in Japan.

References

Fontana, W. and L.W. Buss, The arrival of the fittest: Toward a theory of biological organization. *Bulletin of Mathematical Biology* 56: 1–64. 1994.

Berry, G. and G. Boudol, The chemical abstract machine. *Theoretical Computer Science* 96: 217–248. 1992.

Gánti, T., Organization of chemical reactions into dividing and metabolizing units: the chemotons. *Biosystems* 7: 189–195. 1975.

McMullin, B. and F. Varela, Rediscovering Computational Autopoiesis, *ECAL'97*. 1997.

Mirazo, K., A. Moreno, F. Moran, et. al., Designing a Simulation Model of a Self-Maintaining Cellular System *ECAL'97*. 1997.

Nicolis, G. and I. Prigogine. *Exploring Complexity, An Introduction*. San Francisco: Freeman and Company. 1989.

Păun, G., Computing with Membranes, *Turku Center for Computer Science TUCS Technical Report No. 208* (submitted, also on http://www.tucs.fi). 1998.

Păun, G., P Systems with Active Membranes: Attacking NP Complete Problems, Center for Discrete Mathematics and Theoretical Computer Science CDMTCS-102 (also on http://www.cs.auckland.ac.nz/CDMTCS). 1999.

Suzuki, Y. and H. Tanaka. Order parameter for a Symbolic Chemical System, *Artificial Life VI*:130-139, MIT press. 1998.

Suzuki, Y. S., Tsumoto and H. Tanaka. Analysis of Cycles in Symbolic Chemical System based on Abstract Rewriting System on Multisets, *Artificial Life V*: 522-528. MIT press. 1996.

Tanaka, H., F. Ren, S. Ogishima, Evolutionary Analysis of Virus Based on Inhomogeneous Markov Model, ISMB'99, p 148, 1999.