Available online at www.sciencedirect.com
SCIENCE DIRECT®
BioSystems 73 (2004) 45–55
BioSystems
www.elsevier.com/locate/biosystems

Robust and emergent *Physarum* logical-computing

Soichiro Tsuda [a], Masashi Aono [a], Yukio-Pegio Gunji [a,b,*]

[a] *Graduate School of Science and Technology, Kobe University Nada, Kobe 657-8501, Japan*
[b] *Department of Earth & Planetary Sciences, Faculty of Science, Kobe University Nada, Kobe 657-8501, Japan*

Received 28 May 2003; received in revised form 4 August 2003; accepted 21 August 2003

Abstract

There have been many attempts for realization of emergent computing, but the notion of emergent computing is still ambiguous. In an open system, emergence and an error cannot be specified distinctly, because they are dependent on the dis-equilibration process between local and global behaviors. To manifest such an aspect, we implement a Boolean gate as a biological device made of slime mold *Physarum polycephalum*. A *Physarum* (slime mold) Boolean gate could be an internally instable machine, while it has the potential for emergent computing. First, we examined whether *Physarum* Boolean gate works properly, and then examined its behaviors when the gate is collapsed in terms of hardware. The behavior of *Physarum* changes and self-repairing computing is achieved as a result. The self-repairing against internal failure is one of attributes of emergent and robust computing.

Keywords: *Physarum polycephalum*; Emergent computing; Bio-computing; Logical gate; Concept lattice

1. Introduction

Since 1970s, the notion of "emergence" or "emergent computation" was discussed in various fields, such as Artificial Life (e.g. Crutchfield and Mitchell, 1995). Usually, the notion of emergent computation is expressed as unpredictable global behavior arisen from local non-linear dynamics (Banzhaf et al., 1996; Forrest, 1990). But, a fundamental problem how we can distinguish emergence from error, was not solved (Cariani, 1989, 1997). In a closed self-consistent system, it is easy because the criterion was given (i.e. it is defined by experimenters). On the other hand, in an open system such as living organisms in natural world, it is not clear to distinguish emergence from error because there is no criterion or experimenter there. In these systems, the relation between an object and its observer, or a machine and its user comes to be important.

According to the emergent property mentioned above, an observer is represented by the assumption that the global behaviors can be reduced into local behaviors. If the assumption is demolished, the emergent property appears. Discrepancy between an object and an observer corresponds with dis-equilibration process between local and global behaviors of an object. Therefore, the dynamical modification of the relationship between an object and an observer can be examined by the dynamical discrepancies between local and global behaviors of an object. The key notion is the unity as a whole maintained in an object, because estimating discrepancies needs both the notion of parts and wholeness. With a view to establish collaboration as the first step of realization of emergent computing, we studied about computing by actual living organism as we use true slime mold, *Physarum polycephalum* as a computing device. It has living wholeness.

* Corresponding author.
E-mail address: yukio@kobe-u.ac.jp (Y.-P. Gunji).

0303-2647/$ – see front matter
doi:10.1016/j.biosystems.2003.08.001

Bio-computing/natural computing has in common with our study (Lundh et al., 1997). It was proposed in 1970s, and increasingly developed last decade along with the development of biotechnology. For example, DNA computing (Adelman, 1994; Paun et al., 1998; Mao et al., 2000), protein-based computing (Zauner and Conrad, 2001), and amorphous computing (Abelson et al., 2000) were proposed. Although all of these studies use bio-molecule, these devices never utilize the living things that are characterized by the living wholeness. In fact, DNA computing focuses only on efficient computation (i.e. rapid search for a solution), in Hamilton path problem (Adelman, 1994) and SAT (Paun et al., 1998; Mao et al., 2000). In this scheme, local parallel processing ability is employed only as highly efficient computation. There is little possibility for realization of emergent computation.

By contrast, computation with the plasmodium has a living unity as a whole that cannot be predicted locally. In spite of this discrepancy between parts and whole, those who regard the plasmodium as a stable logical machine, have to neglect such discrepancies. If observers (i.e. users) face with the discrepancies between parts and whole, then it also implies the discrepancy between a real plasmodium and the model, or between an observer and an object. That is why a living system carrying the discrepancies between parts and whole involve the relationship between an object and an observer and has the potential to implement true emergent computation, and/or the computing as dis-equilibration process (Gunji, 1995).

From the reason mentioned above, we conducted some experiments that construct three types of logical gate (AND, OR, NOT), which are basic components of digital computer, with actually living organism, true slime mold. Biological computing operation is studied recently (Motoike and Yoshikawa, 1999; Nakagaki et al., 2000), but these studies do not utilize living wholeness. As device that can implement Boolean operation, billiard balls (Margolus, 1982), local patterns of cellular automata (Langton, 1991), and protein enzyme (Conrad, 1972) are used. But these studies also do not take discrepancy into account. After verified their operability and accuracy, as the next stage, we conducted the experiment that AND gate is collapsed in terms of hardware beforehand, to estimate whether re-organization of the relationship between parts and whole appears or not and such a re-organization leads to self-repairing computation or not. The experimental results are analyzed in terms of logical discrepancies between local and global behaviors expressed by concept lattice (Ganter and Wille, 1999).

2. Methods

2.1. Characteristics of the plasmodium of Physarum polycephalum

The plasmodium of *Physarum polycephalum* is a giant amoeboid organism that consists of a multi-nuclear single cell (Fig. 1A). Its figure is like a developed neural system (Fig. 1B). The behaviors of the plasmodium are studied in detail in terms of protoplasm flow (Miyake et al., 1996; Nakagaki et al., 1999, 2000), and with fast protoplasm flow (shuttle

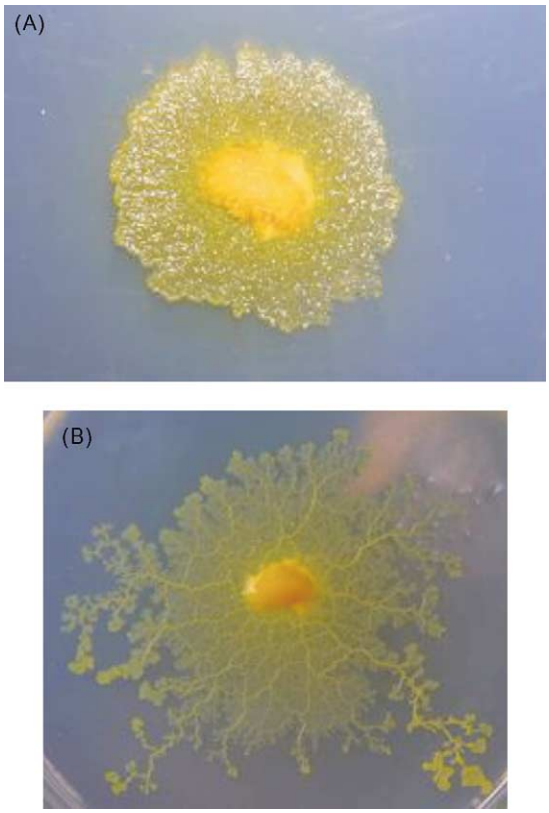


Fig. 1. (A) A photo of small size plasmodium of *Physarum polycephalum* (B) and its developed neuron-like figure.

streaming). It sees that the plasmodium can recognize external stimuli (attractant/repellent) and move to/from the stimuli without losing its wholeness (Ueda and Kobatake, 1982; Ueda et al., 1986). With this property, the plasmodium is called as proto-model of brain.

Another characteristic of the plasmodium is about its plasticity as an individual. If one sets two pieces of plasmodium at a short distance and let them alone, they fuse into one and behave as an individual after that. On the other hand, if one cuts a plasmodium into two or more, each segment can live as an individual.

In addition, we also focused on a property that they tend to avoid their fusion. As mentioned-before, the plasmodium has the ability to fuse with others, but it simultaneously tends to avoid fusing as possible as it can. In fact, in many cases, if there are some spaces to avoid, they migrate to other directions and do not fuse into one. So, fusion is the last selection to take (e.g. they are surrounded by others). It is known that slime secreted from the plasmodium's body, whose main component is polygalactose, works as repellent of other plasmodium and causes this behavior (Asworth and Dee, 1975).

These three characteristics of the plasmodium are assumed as rules to design logical gates (AND, OR, NOT). If the gates are implemented, any expressions in Boolean (i.e. classical) logic can be implemented.

2.2. Experimental setup

The plasmodium was cultured in the bucket that is paved with wet filter papers and was fed commercial oatmeal. All of the plasmodium used in the experiment was kept in food hunger at least 6 h.

In these gates, logical gates are constructed on polystyrene 90 mm × 15 mm petri dish and their sizes are about 50 mm × 50 mm. The plasmodium prefers wet region to dry one. The path of logical gate was created by plastic film that prohibits the move of the plasmodium on 1.5% agar gel that a gradient with respect to attractant (100 mM glucose). Accordingly, the plasmodium can only uncovered region (Fig. 2). The width of the path is about 5 mm. The plasmodium cut by approximately 10 mm × 20 mm was used for the experiment. All the experiment was recorded by digital video camera (Sony DCR-VX1000) once a 30 s.

2.3. Procedure

In those gates, the presence of the plasmodium represents logical value 1, and the absence represents 0. If a pair of inputs (x, y) is (1, 1), two fragments taken from an individual plasmodium are set at input sites. As shown in Fig. 2, the plasmodium always moves according to glucose gradient (downward). Although it has difference, expanding velocity of each plasmodial segment is same as a whole. With this property, we can use the plasmodium as a substitute for electric current to construct logical gates.

2.4. Experiment A

At the beginning, we verified whether these *Physarum* logical gates would work correctly.

As we mentioned above, the plasmodium acts on three rules in all of these experiments:

(1) The plasmodium moves toward highly concentrated glucose (shown as an arrow in Fig. 2).
(2) If two fragments encounter, they avert from one another.
(3) If a fragment encounters the impasse except for the fusion of slime fragments, they are fused.

2.4.1. AND gate

We use averting property of the plasmodium (rule 2) to construct this gate. To let an averting behavior happen, a plasmodium segment needs to close up to the parts except expanding front line because plasmodium secretes slime where it crawled around once. Accordingly, the time lag between plasmodial segments is needed. We use the difference of the length of the path to generate this time lag and control the encounter of two segments. But, from the constraint of the size of petri dish, to generate plenty time lag for proper state of encounter, time lag is generated by the difference of inputting time of plasmodium sengment (about 3 h) in these experiments. Fundamentally, considering that expanding velocity of each individual piece is approximately same, one can control the encounter timing with the length of the path. To make it work more precisely, this gate has buffer region and first expanding plasmodium (right input plasmodium in Fig. 2) moves to this region along the attractant gradient. If a plasmodium extends to this region, it will get out of the

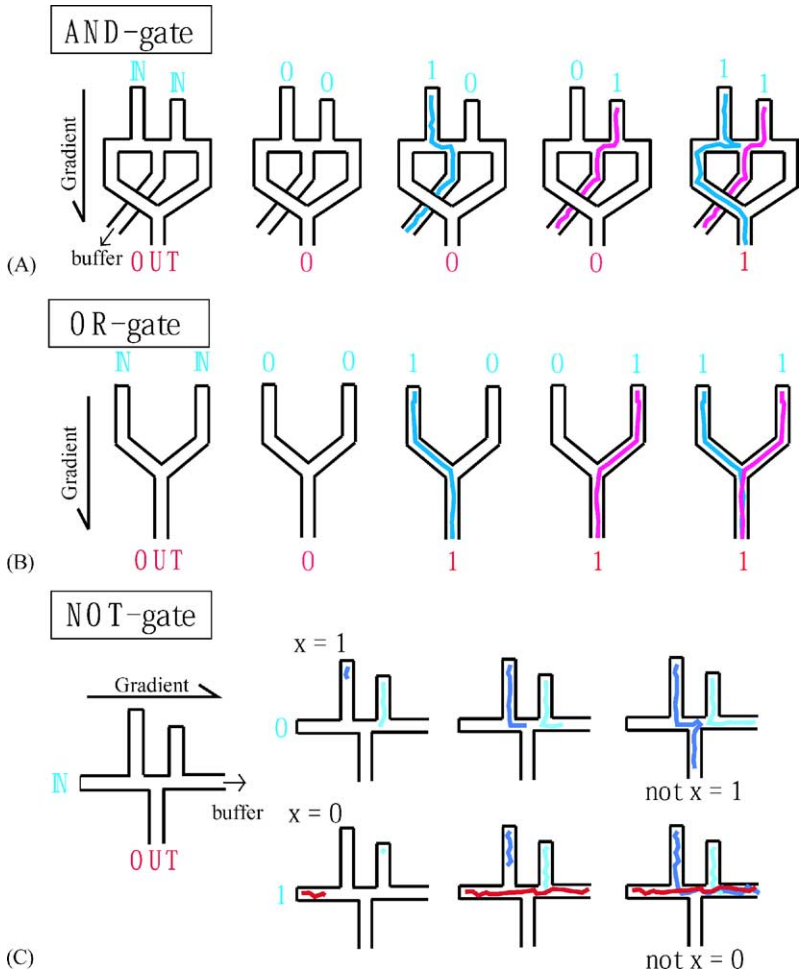


Fig. 2. Schematic diagrams of a *Physarum* Boolean gate—AND, OR, and NOT gate. The accompanied histogram shows the distribution of the time taken to compute outputs, where the horizontal line represents the time (minutes) and the vertical one represents the corresponding frequency. In each diagram, solid curves inside of the gate represent the moves of the plasmodium. The symbols IN and OUT represents the input and output site of a gate, respectively.

gate and no longer commit to the behavior of the gate. Given a pair of inputs as (1, 1), the first plasmodial segment (right) goes to the buffer along the attractant gradient (rule 1), and the other averts from the first one (rule 2). Finally, it takes the left path to the output and reaches the output (Fig. 2A). As a result, one can find 1AND1 = 1. Granted that the left input comes first, it goes to the buffer and the right one averts. Hence, its result remains the same.

In these experiments, we assume that first plasmodial segment is right input, but the contrary case also works out. If left input plasmodium extends to the buffer region first, right one takes the right path to the output and reaches the output. So, the result remains the same.

In the other cases, all of their output is 0 because no plasmodial segment goes to output. From these results, AND operation can be implemented in this gate.

2.4.2. OR gate

This gate is very simple. If there is at least one plasmodial segment, the output comes 1. Even if input is (1, 1), two segments fuse into one because there is no space to avert (rule 3). Output is always 1 except

for input (0, 0) (Fig. 2B). Consequently, OR operation is implemented.

2.4.3. NOT gate

Differently from above two gates, this gate has two plasmodial segments as reference (two downward arrows in Fig. 2). They are inherited independent of the input. And they are always set in the gate with time lag. Right segment (R1) is set at first, and about 3 h later, left segment (R2) is set.

If input is 0, only two reference segments are involved in this gate. The first plasmodium comes to the point of intersection, and goes to the buffer along with the attractant gradient (rule 1) as well as AND gate. Then the other averts from the first one (rule 2), and it takes the path to output because it is the better way to higher gradient region. If input is 1, input plasmodial segment moves in the gate at first. When the input segment reaches the center of the intersection, R1 is set in the gate, and after 3-h interval, R2 is set. But their paths are occupied by input segment and there is no space to escape, the two segments fuse with input segment. Eventually, output 0 is computed. In this manner, NOT operation is implemented.

2.5. Experiment B

Experiment A is the verification whether these *Physarum* logical gates work as primitive computing device with some degrees of precision. Next, we examined how the *Physarum* AND gate works when the gate has a crash in its hardware (Fig. 3A). If this gate works as same as normal AND gate, it can be said as a robust computer.

Robustness of a logical gate is defined by the stable behavior in terms of the input–output correspondence on the truth table, if the internal structure of a Boolean gate is broken. Fig. 3A shows the behavior of robust AND gate. The cross in Fig. 3A represents a broken location (when the plasmodium reaches this point, a blade falls and cuts the forefront of the plasmodium and an impasse appears). If one sticks to the three rules mentioned before, the behavior of slime can be predicted as the following. Given a pair of inputs (1, 1), the one moves toward the buffer because of the gradient (rule 1). The other averts from the one (rule 2), encounters the broken area (i.e. impasse), and finally is fused with the first one because there is no more space to avert (rule 3). It results in 1AND1 = 0. It entails that the AND gate does not work if the internal structure of the gate is broken (Fig. 3A, center).

But, the actual results are different from the prediction dependent on the fundamental three rules. Although some really showed predicted behavior, the other examples showed unpredicted results. After the first fragment moved toward the buffer, the second one avoided the first one (rule 2) and encountered the impasse. Then the second one fuses together with the first one and become one plasmodium (rule 3). Although the prediction till that stage was still correct, the second fragment did not follow the first one and fused plasmodium bifurcated to the new route that leads to right path to output. As a result, the second fragment moved toward the output, and that leaded 1AND1 = 1 (Fig. 3A and C, right). Given all possible pairs of inputs except for (1, 1), the gates work as well as the normal gate. One concludes that the AND gate works to satisfy the truth table of AND even if the gate is structurally broken.

3. Experimental results and analysis

Table 1 shows the result of Experiments A and B. With regard to only Experiment A, one can see that these three gates work with about 85% accuracy. On the other hand, it was verified that broken-AND gate works with 60% accuracy. It is observed that results are robust under slight changes of temperature and moisture to some extent. Of course, in broken-AND gate, “success” means that the fused plasmodium bifurcates and takes the right path to output. There was no case except for unpredicted bifurcating moves and fusing moves (predicted cases). Histograms in Fig. 2 (right) and Fig. 3B shows the relation between the total times of completion of gate working and the num-

Table 1
The rate of successful result of a *Physarum* Boolean gate

	Success/trials	Percentage
AND-normal	11/16	69
OR	19/19	100
NOT	19/23	83
AND-broken	18/30	60
Total	67/88	76

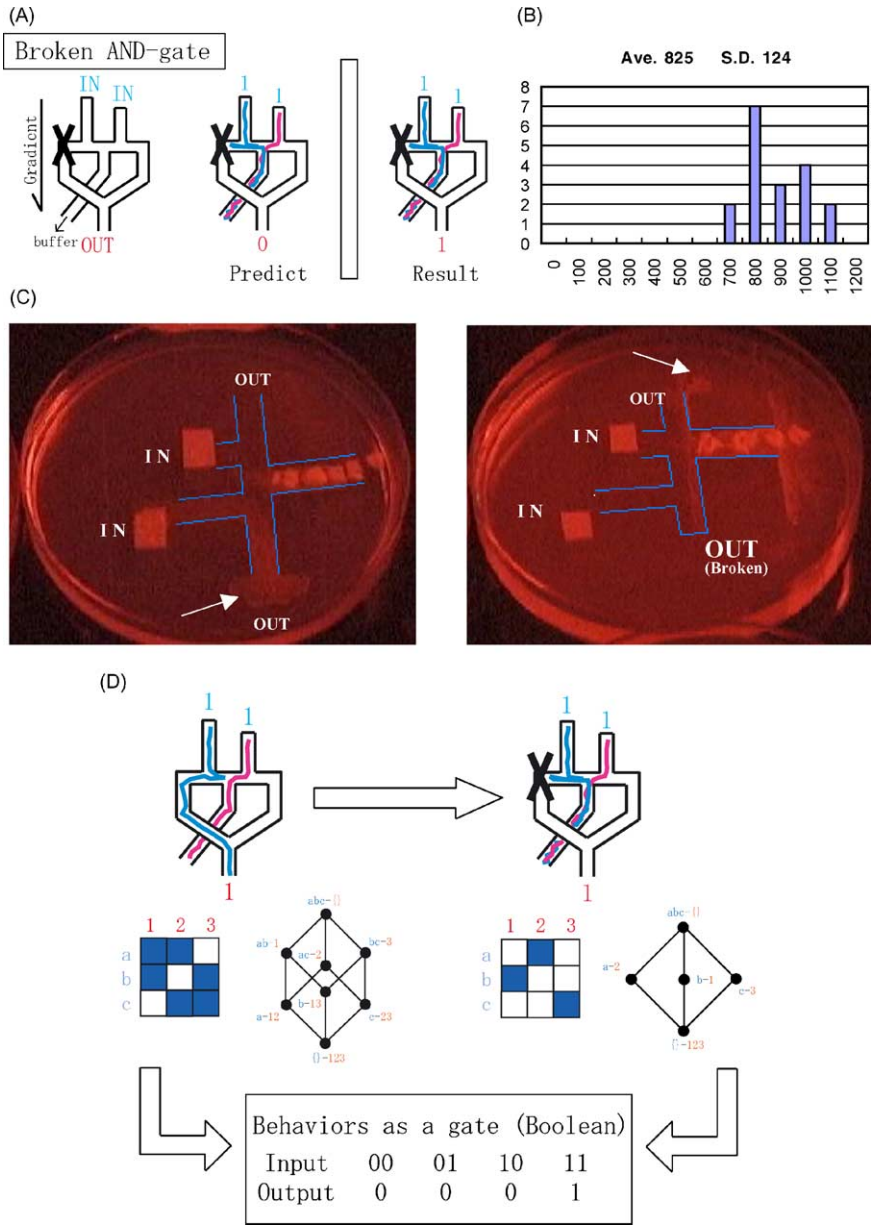


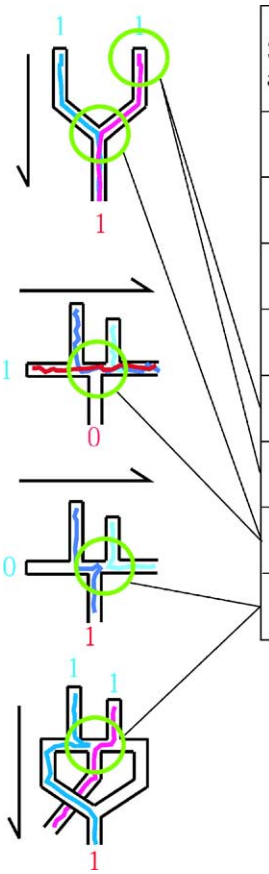
Fig. 3. (A) The schematic diagram of the broken-AND gate, predicted behavior of *Physarum*, and the result of the experiment. A thin arrow represents the gradient of glucose. Solid curves represent the moves of the plasmodium. (B) The histogram shows the distribution of the time taken to compute outputs, where the horizontal line represents the time (minutes) and the vertical one represents the corresponding frequency. (C) Photos of the experiment of AND gate. Normal condition (left) and Broken condition (right) Lines in the photos represent the boundary of the paths. A *Physarum* fragments with attached paper are located at the input site, and move along the glucose gradient following the three rules mentioned in the text. The experiment was conducted under the infrared lamp, and especially in the right path, the most attracted materials (oatmeal) are located. Although the gate is different from the schematic diagram, the essential structure is identical. (D) The algebraic expression of the change of local behaviors of *Physarum* fragments derived from the hardware breakdown, where it is expressed as a concept lattice. Through the change of the experimental conditions, a lattice representing local behaviors of the plasmodium is changed from a Boolean to a non-distributive modular lattice. In the latter lattice, distributive law does not hold for all elements.

ber of examples. With these results, it seems possible that setting up the system can process some kind of task from these three gates.

In the actual AND gate experiment (includes broken one), because of the experimental architecture, we did not unite two outputs of both sides into one path, and assumed as output 1 if the plasmodium reaches the some line of either output. We assumed that first input plasmodium was right one like AND gate in Experiment A, but it is reasonable that the experiment of the contrary case shows the same result.

The logical gates with plasmodium are designed, based on the relation between the three control parameters: (1) the gradients of the attractant, (2) the existence of other individuals, and (3) free space; and the corresponding behaviors. The relation is called the context-relation expressed as Fig. 4, and it shows that all possible combinations of three parameters, presence or absence can be discriminated in terms of the corresponding behaviors. It also means that plasmodium can recognize each combination of three parameters with respect to presence or absence. From this relation, one can estimate the logic of plasmodium in terms of lattice theory.

The logic of plasmodium is consistent with the notion of topological space that is a kind of "filter" to observe all possible combinations. What we take three possible parameters is chosen as a set of observable elements. A topological space is defined as a filter by which some combinations of observable elements can be observed, and explicitly corresponds to a lattice or logic. In other words, empirical data are interpreted into logic by identifiable combinations. Logic is defined not just by three control parameters but by dis-



Spece to avoid	Others	Gradient	Behavior
0	0	0	Immobilized
0	0	1	Free expansion
0	1	0	Simple avoidance
0	1	1	Simple fusion
1	0	0	+taxis/backward move
1	0	1	+taxis/foreward move
1	1	0	Gradient-oriented fusion
1	1	1	Gradient-oriented avoidance

Fig. 4. Classifications of three parameters as local behavioral patterns, and their behaviors (right). Left schematic figures represent correspondences of some cases to actual behaviors in the gates.

tinction among combinations of control parameters. With respect to the relationship between observed elements and an identifiable phenomenon resulting from combinations of observed elements, one is always employed to a particular logic that is not necessarily Boolean logic based on a set theory.

Given a set $G = \{a$ (gradient), b (existence of other plasmodium), c (presence of escape-route)$\}$, one can obtains a power set of G, $P(G)$, that consists of all subsets of G. If observed subsets are chosen from $P(G)$ as S (i.e. $S \subseteq P(G)$), one can obtain a topology of recognized space in the form of a lattice. A lattice is defined as a partially ordered set closed with respect to intersection and union. For example, if $S = \{\phi, G\}$, $\phi \cap \phi = \phi \in S$, $\phi \cap G = \phi \in S$, $G \cap G \in S$, and then it is closed with respect to intersection. In a similar manner, it is also verified that S is closed with respect to union (i.e. for all x, y in S, $x \cup y \in S$). By contrast, if a subset $\{a\}$ is also empirically observed and one obtains $S = \{\phi, G, \{a\}\}$, then a lattice becomes a non-complemented Heyting algebra. It does not contain the law of excluded middle.

In the case of plasmodium, we obtain $S = P(G)$ because all combinations of three control parameters can be distinguished with each other in a term of plasmodium moves. Fig. 4 shows plasmodium's behavior corresponding to a subset of three parameters. For example, compare the moves of plasmodium under $\{a, b\}$ with ones of $\{a\}$. The experimental condition with $\{a\}$ means that there is a gradient of glucose, and $\{a, b\}$ means that there is a gradient and other plasmodium individual. In both conditions there is no space to avoid other plasmodium. The motion under $\{a, b\}$ is observed as simple fusion, and the motion under $\{a\}$ is observed as free expansion along gradient. As a result, one can distinguish $\{a\}$ from $\{a, b\}$ in a term of plasmodium behavior. In analogous manners, we can distinguish all combinations of three control parameters in a term of plasmodium moves, and then we can obtain $S = P(G)$ that is a set lattice. Especially, it also satisfies the distribute law (i.e. for all x, y, z in S, $x \cap (y \cup z) = (x \cap y) \cup (x \cap z)$) and the complemented law (i.e. for all x in S, there exists an x^c such that $x \cap x^c = G$, $x \cup x^c = \phi$). Distributed complemented lattice is called Boolean lattice, and it exactly corresponds to the prepositional classical logic. As a result, the relation between parameters and the corresponding behaviors can be expressed as a Boolean lattice.

A lattice obtained from the context-relation is constructed also as a concept lattice (Ganter and Wille, 1999; Gunji et al., 2002). If the context-relation is given as a binary relation, R, between two sets, G and M, a formal concept is defined as a pair (A, B) with $A \subseteq G$, $B \subseteq M$, such that $A' = B$ and $B' = A$, where

$$A' = \{m \in M | gRm, \forall g \in A\},$$
$$B' = \{g \in G | gRm, \forall m \in B\}.$$

For all concepts, a partial order, $\geq$, is defined by inclusion relation (i.e. $(A_1, B_1) \geq (A_2, B_2) :\Leftrightarrow A_1 \supseteq A_2$). If a partial order is drawn as a line, the structure among concepts is expressed as a Hasse diagram representing a finite lattice. In our experiments, G is expressed as a set of stimulus or environmental factors $G = \{a$ (gradient), b (presence of other plasmodium), c (presence of escape-route)$\}$ and M is defined as a set of behaviors $M = \{1$ (fusion), 2 (chemicotaxis), 3 (avoidance)$\}$. A binary relation between G and M in the condition of Experiment A, is shown in the left matrix in Fig. 3D, where an environmental factor g relating a behavior m is represented by gRm that is shown as a filled square. Because three components of environmental factors are not independent, one has to remark the relation. With respect to the gradient column (a), the relation between the gradient and the behavior is defined by the presence of the behavior induced from gradient. There is little observation of which avoidance from other individual against the gradient. As a result, we obtain that $aR1$, $aR2$, and $aR-3$. With respect to the presence of other plasmodium column (b), the relation between b and the behavior is defined by the presence of the behavior induced from the other individual. Both fusion and avoidance to the other plasmodium were observed, although chemicotaxis is not derived from the presence of other plasmodium. It shows that $bR1$, $bR2$ and $bR3$. With respect to the escape-route column (c), the relation between c and the behavior is defined by the behavior under the presence of the escape-route. There is little observation of the fusion with other plasmodium under the presence of the escape-route. As a result, we obtain $cR1$, $cR2$ and $cR3$.

In Experiment A, a concept lattice obtained from the binary relation is a Boolean lattice as Fig. 3D (center). An element of a lattice is represented by a formal concept. For example, given $A = \{a, b\}$, $A' =$

$\{m \in M | gRm, \forall g \in A\} = \{m \in M | aRm \wedge aRm\} = \{1\}$. On the other hand, given $B = \{1\}$, $B' = \{g \in G | gRm, \forall m \in B\} = \{g \in G | gR1\} = \{a, b\}$. A pair $(A, B) = (\{a, b\}, \{1\})$ satisfies $A' = B$ and $B' = A$, and then it is a formal concept. Finally, all formal concepts represent a power set of G, and then the concept lattice coincides with a set lattice. That is why the result of concept lattice analysis is consistent with the analysis in terms of the recognized sets of all possible combinations.

With respect to concept lattice analysis, the relation between the stimulus and the corresponding behaviors under the Experiment B is different from the one under the Experiment A (Fig. 3D, right matrix). In the Experiment A, whenever the plasmodium encounters the other individual under the presence of glucose gradient, it is fused with the other moving along the gradient. It shows that fusion is induced from the gradient. It is expressed as the presence of the relation between gradient and fusion (i.e. $aR1$). Compared with Experiment A, the second plasmodium moves against the gradient in encountering the first plasmodium in the Experiment B. It shows that there is no fusion induced from the gradient. It, therefore, leads that $a\not{R}1$. In the similar manner, it is found that $bR1$ and $bR3$. After encountering the impasse (i.e. the failure of the hardware), there is fusion induced from the other plasmodium (i.e. $bR1$) but no avoidance induced from the other plasmodium (i.e. $b\not{R}3$). Actually, the second plasmodium is fused with the first one and then it moves toward the free space. It never avoids the other plasmodium. With respect to the presence of the escape-route, we observed little chemicotaxis with the escape-route in the Experiment B. If there is the escape-route, the plasmodium moves against the glucose gradient, and it moves toward the free space (i.e. $c\not{R}2$). By contrast, in the first contact between the first and second plasmodium, the second plasmodium avoids the second one and moves to the escape-route (i.e. $cR3$). As a result, the binary relation between G and M is changed to the matrix as shown in Fig. 3D (right).

The Hasse diagram of this result is a modular lattice in which distributive law does not hold for all elements, as shown in Fig. 3D (right center). Recall the significance of distributive law, $x \cap (y \cup z) = (x \cap y) \cup (x \cap z)$. In a lattice, two binary operations the least upper bound, $\cup$, and the greatest lower bound, $\cap$ are defined. Elements of a lattice are employed for these operations. If a distributive law holds in a lattice, any elements of a lattice can be copied and manipulated. Although a variable x appears once at the left hand, it appears twice due to the copy in the equation of the distributive law. On the contrary, it means that parallel operations $x \cap y$ and $x \cap z$ can be summed up by one operation. Compared with a distributive lattice, elements of a modular lattice cannot be copied without satisfying a specific condition, and then $x \cap (y \cup z)$ does not coincide with $(x \cap y) \cup (x \cap z)$. Therefore, if the model of a computation in a modular lattice is expressed as a particular equation involving $x \cap (y \cup z)$, the model is destined to be inconsistent with a real behavior following the equation involving $(x \cap y) \cup (x \cap z)$. Because we define a concept lattice to describe the relationship between the stimulus and the corresponding behaviors, such a lattice is a model of local behaviors of plasmodium. The change of a lattice from a Boolean (Experiment A) to a modular lattice (Experiment B), therefore, implies the change of the local behaviors of plasmodium. In Experiment A, all formal concepts derived from the relationship between stimulus and behaviors can be perfectly manipulated. On the other hand, in Experiment B, the description consisting of formal concepts (i.e. computing results of the binary operations in a concept lattice) is destined to be imperfect. An observer overlook all concepts simultaneously and then his model for the local behaviors of plasmodium must be imperfect.

Compared the results of Experiments A with B, logic of local behaviors of plasmodium changes, although in both experiments implemented plasmodium gates work properly as Boolean gates. In Experiment A, local behavior expressed as a Boolean concept lattice is consistent with the behavior as a Boolean gate. In Experiment B, there is discrepancy between the local behaviors and the behaviors as a Boolean gate. The germ of emergent computing can be found in this aspect.

4. Discussion and conclusion

Consider these experimental results in the context of emergence/emergent computation. Recall our point for the emergent computation. In order to implement the emergent computing, the relationship between a machine and its user has to be embedded in a ma-

chine. For this purpose, the plasmodium is adopted as a computing agent, because the relationship between a machine and its user is embedded in the form of the relationship between parts (local behaviors) and whole (global behaviors). In our framework, the logic of global behaviors is described as the relation table between inputs and outputs. If implemented three Boolean gates, AND, OR and NOT gates work properly, global logic is defined as Boolean algebra. The logic of local behaviors is defined as a concept lattice. By this analysis, we can estimate discrepancies between local and global behaviors.

First, in Experiment A, there is no discrepancy between local (local behaviors of the plasmodium) and global (input–output correspondence) behavior because both behaviors are expressed as Boolean lattices. As a result, it looks as if global behaviors of the plasmodium gates could be reduced into the local behaviors because they are consistent with each other. On the other hand, in Experiment B, although local behavior is expressed as a non-distributive modular lattice, global behavior is still expressed as a Boolean lattice. It is clear to see that the discrepancies between local and global behaviors are inherited in computing and it gives rise to self-repairing computing. Although global behaviors as a Boolean gate cannot be controlled by local behaviors of plasmodium, one can use plasmodium gate as a Boolean gate. Computing (global behavior) is generally based on perfect control of local behavior. As far as a machine works properly in this sense, we call it a stable machine. In Experiment A, it looks as if global behaviors could be controlled by local behaviors. By contrast, in Experiment B, it is uncontrollable local behaviors that give rise to Boolean gate robustly. In this sense, such a machine is here called a robust machine. In other words, a robust computing is defined by a computing inheriting the discrepancies between local and global behaviors. Not a stable but a robust machine has potential to emergent computing. From Experiment A, one cannot see that a machine is robust, but through Experiment B, one can see that plasmodium gates are robust machines in principle.

Material computing yields the problem regarding the negotiation between parts and whole. Imagine a ballistic computation (Margolus, 1982). In assuming that a trajectory of a billiard ball is a truth value, Boolean gates are implemented by controlling a trajectory by a specific walls and collisions of balls. Input is defined by a configuration of balls and output is observed as a particular trajectory at a particular site. Therefore, a user of a machine (i.e. observer) needs the precise knowledge on the topography of trajectories, or the relationship between a whole billiard table and a point (a part) in a table. The knowledge is abandoned only to a user. In other words, if the knowledge attributes only to a user, discrepancies between a machine and a user are hidden perfectly. By contrast, if the knowledge on the relationship between parts and whole attributes not only to a user but a machine, the process of computation must be dis-equilibration process (Gunji, 1995). The observer's perspective on parts–whole is betrayed by the actual relationship between parts and whole in a machine, and that triggers to make an observer change his perspective and how to control a machine. This process keeps on going, and then the process of computation can be maintained robustly and simultaneously opened to emergent computing.

Although a machine is instable in principle, the interaction between a machine and a user can be maintained as "computation." Recall the results of Experiment B. In spite of the hardware failure, a plasmodium gate can work properly. As a result, a user can use such a gate as a Boolean gate without the knowledge on the relationship between local and global behaviors. Against a hardware failure, dis-equilibration process between local and global behavior results in material-based new AND gate. Discrimination of emergence from error depends on dis-equilibration process.

The dis-equilibration process is generated from robust computation in which the discrepancies between parts and whole change the relationship between local and global behaviors of computing. In Experiment B, discrepancy between non-distributive modular lattice (local behavior) and Boolean lattice (global behavior) changes the notion of computation, from the computation with consistency between local and global behaviors to the computation inhering the discrepancy. There is discrepancy, but global notion of computing still remained. That is why a user can use plasmodium gates as logical gates. In assuming that global behaviors of computing is unchanged, it makes sense that re-organization of parts can drive the new relationship between non-distributive modular lattice and Boolean lattice. Re-organization of parts plays an essential

role in material or biological computing in a parallel fashion.

Protein, DNA and/or membrane computing has an advantage in parallel computing. For example, DNA computing can utilize huge numbers (with the scale of Avogadro constant) of computing resources. Although searching for a solution in a solution space proceeds in a parallelism of such a huge numbers of computations, there is no intelligent device of distribution of computing resources. Re-organization of parts is a hopeful candidate of such an intelligent device. Imagine that two computing agents simultaneously access the unique computing resources. If the resource cannot be divided into well-defined computing resources in advance, parallel access entails to dead locked. In this sense, parallel computing is impossible in principle. But such a perspective is based on the ideal formal computation, and material process as computation proceeds with such parallel accesses to the unique computing resource. Clearly, the parts–whole relationship keeps on changing. Before an agent's touching a resource, a resource is not only the outside of the computing resource but yields the environment of execution of computation for the agent. After touching, the computing resource is embedded into the computing agent. Re-organization of computing resources and agents proceeds without dead lock, and that can lead to new idea of parallel processing. In the previous sense of parallel processing, computing output needs the synthesis of each processing in parallel fashion. The synthesis needs satisfactory long time. By contrast, parallel processing with re-organization of computing agents and resources has an advantage with respect to computing time.

A *Physarum*-computer user can acknowledge as computer that has self-repairing system. It could lead to the dis-equilibration in which a user can regard the unpredicted output of a computer not as an error but as the emergent state. It could lead to the emergent computing.

References

Abelson, H., Allen, D., Coore, D., Hanson, C., Homsy, G., Knight, T.F., Nagpal Jr., R., Rauch, E., Sussman, G.J., Weiss, R., 2000. Amorphous computing. Comm. ACM 43 (5), 74–82.

Adelman, L.M., 1994. Molecular computation of solutions to combinatorial problems. Science 226, 1021–1024.

Asworth, J.M., Dee, J., 1975. The Biology on Slime Moulds. Edward Arnold Ltd., London, Tokyo.

Banzhaf, W., Dittrich, P., Rauhe, H., 1996. Emergent computation by catalytic reactions. Nanotechnology 7, 1–8.

Cariani, P., 1989. On the design of devices with emergent semantic functions. Ph.D. Thesis, State University of New York at Binghamton, Binghamton, New York, 234 pp.

Cariani, P., 1997. Emergence of new signal-primitives in neural networks. Intellectica 2, 95–143.

Conrad, M., 1972. Information processing in molecular systems. Curr. Modern Biol. 5, 1–14.

Crutchfield, J.-P., Mitchell, M., 1995. The evolution of emergent computation. Proc. Natl. Acad. Sci. U.S.A. 92 (23), 10742–10746.

Forrest, S., 1990. Emergent Computation. MIT Press.

Ganter, B., Wille, R., 1999. Formal Concept Analysis: Mathematical Foundations. Springer-Verlag, Berlin.

Gunji, Y.-P., 1995. Global logic resulting from disequilibrium process. Biosystems 38, 127–133.

Gunji, Y.-P., Kusunoki, Y., Aono, M., 2002. Interface of global and local semantics in a self-navigating system based on the concept lattice. Chaos, Solitons Fractals 13 (2), 261–284.

Langton, C.G., 1991. Life at the edge of chaos. In: Langton, C.G., Taylor, C., Farmer, D., Rasmussen, S. (Eds.), Artificial Life II, SFI Studies in the Sciences of Complexity, vol. X. Addison-Wesley, pp. 41–91.

Lundh, D., Narayanan, A., Olsson, B., 1997. Biocomputing and Emergent Computation. World Scientific.

Mao, C., Labean, T.H., Reif, J.H., Seeman, N.C., 2000. Logical computation using algorithmic self-assembly of DNA triple-crossover molecules. Nature 407, 493–496.

Margolus, N., 1982. Physics-like models of computation. Physica D10, 81–95.

Miyake, Y., Tabata, S., Murakami, H., Yano, M., Shimizu, H., 1996. Environmental-dependent self-organization of positional information field in chemotaxis of *Physarum plasmodium*. J. Theor. Biol. 178, 341–353.

Motoike, N.I., Yoshikawa, K., 1999. Information operations with an excitable media. Phys. Rev. E 59 (5), 5354–5360.

Nakagaki, T., Yamada, H., Ueda, T., 1999. Modulation of cellular rhythm and photoavoidance by oscillatory irradiation in the *Physarum plasmodium*. Biophys. Chem. 82, 23–28.

Nakagaki, T., Yamada, H., Toth, A., 2000. Intelligence: maze-solving by an amoeboid organism. Nature 407, 470.

Paun, G., Rosenberg, G., Salomaa, A., Brauer, W., 1998. DNA Computing: New Computing Paradigm. Springer-Verlag, Heidelberg.

Ueda, T., Kobatake, Y., 1982. Cell biology of Physarum and Didymium. Academic Press, New York.

Ueda, T., Matsumoto, K., Kobatake, Y., 1986. Exp. Cell Res. 162, 486–494.

Zauner, K.-P., Conrad, M., 2001. Molecular approach to informal computing. Soft Comput. 5 (1), 39–44.