Artificial Ecosystem on the Resource Conservative Tierra Structure

Shuichi Matsuzaki*, Hideaki Suzuki**, Minetada Osano*

*Graduate School of Computer Science and Engineering, Aizu University
Aizu-Wakamatsu City, Fukushima 965-8580 Japan

**ATR Network Informatics Laboratories
2-2-2 Hikaridai Seika-cho, Soraku-gun, Kyoto 619-0288 Japan

Abstract

In this paper, we present an Artificial Life system with a remodeling of Tierra. This system provides a rule for a system environment, which is induced by the natural resource property: "matter resources" will be conserved and recycled through biochemical reactions. Because of that point, fundamental components (as matter) of digital creatures being conserved (never being added/deleted) are introduced into this system. Consequently, the system forces the creatures to recycle their program ingredients in order to self-replicate. We have developed two distinct ancestor programs called "Plants" and "Animals," which they can recycle resources by executing photosynthesis/preying. Our experimental results demonstrate that particular interactions, which are similar to a *food web*, appeared as a result of the recycling actions, and such interactions stabilized the population equilibrium among groups of Plants and Animals. In addition, co-adaptive evolutionary behaviors caused by predator-prey interactions are also observed.

Keywords: Tierra, self-replication, resource conservative system

1. Introduction

The Cambrian explosion, an example of Adaptive radiation with the sudden appearance of many animal body plans, produced most types of higher taxonomic categories of actual animals [1]. It is assumed that the rich environmental resources of "space" and "matter" mainly caused this outstanding evolutionary event.

Tierra is an artificial life (Alife) system designed to realize such biological behaviors in computer architectures [2,3]. A Tierran digital creature that consists of a self-replicating program and a CPU create its replicant (copy of itself), and also evolve it through mutations. For example, one class of mutants called "parasites" appears in Tierra experiments, and these parasites will self-replicate by partly depending on other programs. In addition, Network Tierra, a following study of Tierra, modeled digital creatures that are organized by several distinct programs, and showed that the programs can differentiate into more specialized units [3,4]. Having shown such evolutionary behaviors in artificial systems, a kind of extraordinary evolution of the natural ecosystem can be reproduced by the Alife systems, while another type of evolutionary phenomenon in which it will continuously affect creatures with far tinier changes is hardly shown in such systems.

In principle, natural creatures belong to a particular trophic level, and their evolutions are affected by mutual interactions (such as predator-prey interactions) in order to stabilize an entire food web (the resource-recycling system). In terms of this, an aspect of the preservation of space/matter resources, which is a type of abiotic rule, causes creature interaction. In contrast to such a system, resources become freely available (with looser limits) within the above-mentioned systems that Tierra shows. From these points, we consider that a strong restriction of the resources may produce one kind of evolutionary behavior, while a weak one causes another kind.

In this paper, we present an Alife system with a remodeling of the Tierran structure. The remodeling focuses on the appearance of a spontaneous interaction of digital creatures and the global systems resulting from the interaction. A theory proposed by Suzuki *et al*. [5] formed the underlying concept of this study: the theory

of “Symbol Resource Conservation”. This theory suggests that the component symbols (or symbol ingredients) in which matter is equivalent be conserved in each elementary reaction. A set of Tierran instruction words, represented by bits, composes a self-replicating program. Those bits are regarded as a fundamental matter of the system. The instruction words, however, can be copied/overwritten by elementary reactions (computations). Hence, the Tierra Operating System (OS; namely, a higher-level manager) prohibits overwriting on “active instruction words” (components of active programs). Nevertheless, the total number of active instruction words explodes in the RAM unless a *reaper* perpetually eliminates creatures. If no elimination occurs, the system will not be able to function due to memory overflow. Without conserving matter, systems with a conserved (limited) space will experience this problem in principle. Consequently, we have assumed that the instruction words of Tierra should be given more rational characteristics as matter. Our model introduces an environmental rule whereby each instruction word is conserved, and because of this rule, creatures recycle the words to self-replicate. We show here two types of creature designed with distinct manners of recycling the words.

In the next section, we introduce the digital creatures and some important rules of the system. Following that, we show several of our experimental results, and then provide concluding remarks.

2. The Model

2.1. Self-Replicating Programs

Our model characterizes a “word” with the following two aspects: on the one hand it is represented by six bits (each order corresponds to a type); on the other hand, each of its appropriate sets represents a self-replicating program. The system produced by this model conserves every bit as the ingredient of the words. All–possible combinations of six-bit sets are classified into two categories: *instruction words* and *no-instruction words* (the CPU will decode instruction words into particular computations, but will not decode no-instruction words). These words are necessary for digital creatures to use/reuse in their self-replication, and such actions are included within a copy procedure of our ancestor programs (thus, this part of the Tierra program has been almost fully renewed). We have developed the following two types of ancestor program with distinct copy procedures.

(1) Plant Ancestor

In principle, creatures in this model can replicate one instruction word within one replicating cycle (thus, self-replication will be accomplished by finishing all the cycles).

An ancestor program called a “Plant” synthesizes no-instruction words during each replication cycle, and then produces instruction words. Figure 1 shows an actual example of a replication cycle, and each of its actions (1-7) is described as follows:

1. Identify an instruction word that a creature will replicate during this cycle.
2. Obtain an arbitrary no-instruction word (Element A) by searching entire memory space.
3. Obtain another arbitrary no-instruction word (Element B) by searching entire memory space.
4. Produce an instruction word (Product. It is equivalent to what is identified in **1**) by using and arranging 12 bits within Elements A and B.
5. Produce an arbitrary instruction word (By-product) by using and arranging six bits that are not used in **4**.
6. Move the Product to an address allocated for a replicant.
7. Move the By-product to the address where Element B was.

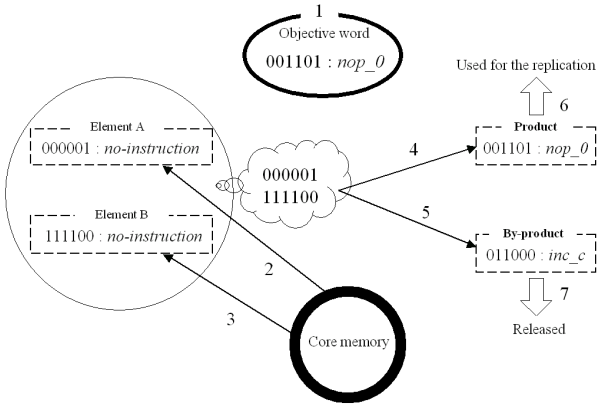

Figure 1. An example of the replication cycle for Plants.

In brief, in order to replicate each word, ingredients of two no-instruction words (obtained in **2** and **3**) will be

recycled to ingredients of two instruction words (**4** and **5**). Since this action will increase the number of instruction words in a system, Plants are considered as a producer of instruction words in a system.

(2) Animal ancestor

In each replicating cycle, Animals first identify an instruction word that they will replicate during this cycle (the same manner as Plant's), after which they will begin searching this instruction word by checking each address outward in both directions (within the range of 350). Then, Animals will obtain the instruction word and finish the cycle if they can satisfy the following conditions in parallel:

(a) If the creature finds the instruction word;
(b) If its "CPU state" equals *True*.

The condition of (a) will be satisfied when the search reveals a corresponding instruction word. In the next condition, the CPU state represents a kind of Boolean register (*True*/*False*) equipped in the CPU, and the state existing at the every beginning of replication cycles is fixed to *False*. It can only be changed with "template matching," which will be executed in parallel with the search. The search template, which has been installed on the Animal programs, will look for a corresponding template matching itself (such a manner is similar to that of Tierra). On matching the search template with another template, the CPU state will switch to another state: *True* →*False* or *False*→*True*.

To take the case of Animal ancestor, a matching of its search template with its own "membrane template" (templates which exist at beginning and end of each program) will occur, and the Animal changes its CPU state (*False*→*True*). Having changed its CPU state to *True*, the Animal satisfies the condition of (b), and it will obtain a word to replicate when it satisfies the condition of (a) after this. However, by matching its search template with another membrane template afterward, it will change its CPU state again (*True*→*False*), and it will not obtain words.

Animals can obtain not only non-active instruction words (not a component of programs), but also active instruction words, enabling them to take component instruction words away from the other self-replication programs. Since such behavior of Animals can eliminate other creatures, it is regarded as predation in our model.

2.2 System environments

(1) Death

A function of OS for eliminating a creature removes a CPU from the system and changes the state of component words into the non-active one. This function will eliminate a creature in which at least one of the following conditions is fulfilled:

- By losing an instruction word necessary to self-replicate, the OS will eliminate the creature. Predations and some mutations (wrong cases) can remove/modify instruction words, and they might make a self-replicating program an invalid one.
- By failing to execute any of its own instruction words within 3,000 steps, the OS will eliminate it. An *instruction pointer* might jump incorrectly to other programs by errors/mutations, but will rarely return to its own program.
- By failing to replicate an instruction word in 100 cycles, the OS will eliminate this creature at 10%.

(2) Resource Management

The OS sets the following rules relating to an existence of words.

- If the number of active instruction words amounts to 80% of the total, the OS will prohibit creatures from obtaining any more words. Without this rule, creatures use most of the words for their own self-replication, giving rise to result that none of them will be able to obtain enough further words to self-replicate.
- If a non-active instruction word has not been used by any creatures during those first 100 steps, the OS will change it into an arbitrary no-instruction word. As we have already seen, since Plants will synthesize instruction words from no-instruction words, the number of instruction words will unilaterally increase. In this respect, this rule will make up a balance of the two word types. In comparison with the natural world, this rule is regarded as a kind of dissimilation: the process of gradually changing materials of dead bodies into other materials by chemical reactions.

3. Experiments

3.1. Basic Experiment

Here we present experimental results of a system in which Plant and Animal ancestors are initially inoculated. First, Fig. 1 shows variables for populations of Animals and Plants.

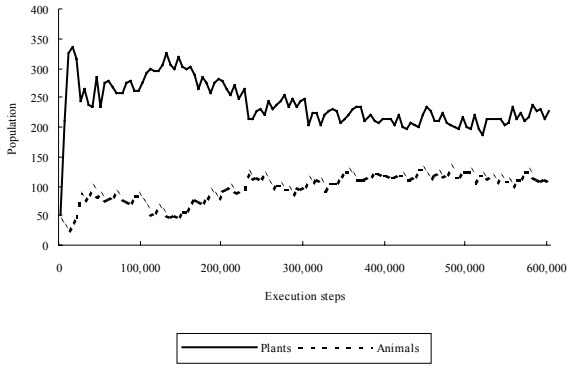

Figure 2. Populations of Animals and Plants.

In the early steps the system explodes its population of Plants and brings the population almost to its maximum possible size. The system then starts increasing its population of Animals, decreasing the Plant population inversely. Although the population equilibrium between them fluctuates during the several-hundred-thousand steps of the experiment, it does gradually stabilize after a while. With respect to these results, we have considered the following:

1. The words are randomly given initially (the total number is fixed). Plants can obtain sufficient no-instruction words, and they increase. On the other hand, Animals can rarely gather enough instruction words for self-replication because there are few available instruction words around them.
2. By increasing the population of Plants, an increase in the total number of instruction words will follow, after which the Animal population will start to increase. However, having decreased the number of no-instruction words, further replications of Plants becomes increasingly difficult.
3. A decrease in the total production of instruction words will follow a decrease in the Plant population (and an increase in the Animal population). Through the function of the OS (see 2.2(2)), the number of instruction words will also decrease after that. Thus, further replications of Animals become ever more difficult. However, having boosted the number of no-instruction words, the population of Plants will start to increase again. As a result of these interactions, the population equilibrium between Animals and Plants will stabilize.

Next, we describe another experimental result that illustrates predator-prey interactions. In the system, the interaction by which predators attempt to gain the instruction words of prey is regarded as a predation. Before every predation occurs, the predator checks that its search template can match a membrane template of the prey. If these templates match, the prey will be protected from the predator. Therefore, It is considered that a diversity of matching patterns of the membrane template is a measure of a creature's ability to protect itself against enemies (by making a template longer, the membrane template will come to match more types of short templates representing the search templates in most cases). For this reason, we use the Matched Template Number (MTN) as the measure that indicates a prey's ability to protect its entire program against predation. The MTN of each creature represents the total number of template matchings in which its membrane templates match templates of "all possible four-letter words" (the search template basically contains four-letter words). Figure 3 shows variables of the MTN (maximum: 32) of each group.

Figure 3. Matched Template Number.

The MTN of each group keeps increasing almost through the experiment. In addition, the MTN of Plants

is higher than that of Animals in most cases. An increasing MTN actually indicates a lengthening of templates, and places the heavier load on self-replication; however, it has been observed perpetually in the experiment, that is to say, this result shows mutations in creatures for adapting to predations by other Animals.

As we have seen above, the experimental results in Figs. 2 and 3 indicate that this system produces two varieties of creature interaction that influence creature behaviors. We finally summarize the consideration given in the experiment.

- In this resource-conservative system, a particular relationship between Animals and Plants in which each group indirectly cooperates with each other has appeared through their reactions of the resource-recycling process.
- In local reactions, competition among individual creatures has been observed, causing creatures to mutate in order to adapt to each other.

3.2. Experiment with a Simple Replicator

It is considered as one of the advantages of resource-conservative systems that they may limit an explosion of trivial replicators with a simple structure. These systems will preserve the digital creatures from being expelled by such trivial replicators. From this point, we have planned an experiment in which simple replicators are deliberately inoculated. We have conducted the experiments not only in our model, but also in Tierra, to compare the results. We prepared a simple replicating program with 20 instruction words that were all composed by Tierran instruction words. Therefore, this program can perform within both of those models.

Figure 4 shows the results of the experiments where the simple replicator is added in 3,000 steps.

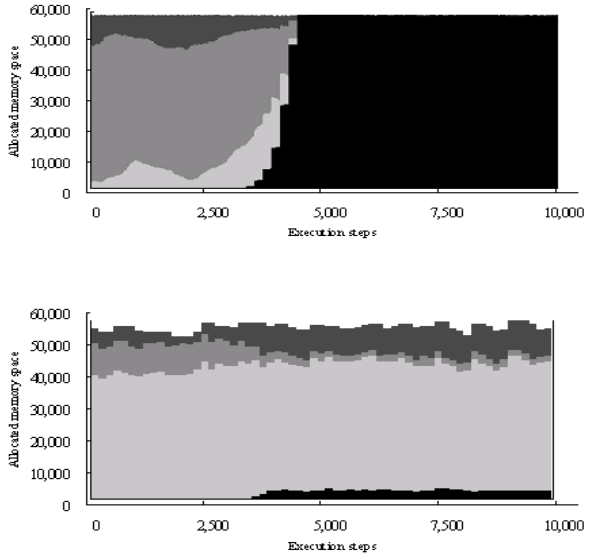

Figure 4. Results of an experiment with a simple replicator. Both of these graphs, Tierra (upper) and our model (lower), show the balance of power among creature groups. The Tierra experiment evaluates the balance between simple replicators (black) and normal creatures (other colors). The experiment conducted with our model evaluates the balance among three groups: simple replicators (black), Plants (light gray), and Animals (gray/dark gray).

In the Tierra experiment, the group of simple replicators drastically expands, reducing groups of normal creatures. Then, it occupies an entire space, and the others disappear. In our model, the group of simple replicators expands slightly at the beginning, but it hardly changes at all afterward.

Since the simple replicators are actually capable of self-replicating almost six times faster than the ancestor creatures, the Tierra result is considered natural at this point. In contrast, the results from our model will contradict that, with the following principles:

- Reducing the number of Plants will make the obtaining of instruction words difficult.
- Only a few kinds of word are necessary for a simple program to self-replicate, and they will be consumed within a short period. Then, only these words will become insufficient, and this will restrain the simple program from self-replication.
- A simple program has few templates, and seldom avoids predation (by Animals).

These results demonstrate the robustness of our mechanism of our system compared with that of Tierra against the simple replicating program, and owing to it this system keeps such replicators under control. In general, any creature that cannot be placed in a cycling rule of this system will be weak, even if it can self-replicate faster than the others.

4. Conclusion

We have proposed an artificial system that had been designed based on the Tierra structure. The system ruled that any program component regarded as matter is conserved. We also introduced two self-replicating programs (Animals and Plants) in which processes to recycle the components are preprogrammed. We then, demonstrated the evolutionary performances of such digital creatures. First, particular creature interactions caused by the recycling processes were observed that stabilized the population equilibrium existing among creature groups. Next, we showed predator-prey relationships through local inter-creature competition. As a result of such competition, creatures mutated their programs in order to adapt to each other. Finally, the robustness of this system against a trivial replicator was proved by a comparison with Tierra.

Acknowledgements

The second author's research work was supported in part by the National Institute of Information and Communications Technology and Doshisha University's Research Promotion Funds.

References

[1] Sole R, V. Salazar-Ciudad I., & Garcia-Fernandez J. Landscapes, (2000). *Gene Networks and Pattern Formation: on the Cambrian Explosion*. Adv. Complex Systems 2, 313-337.

[2] Ray T, S. (1994). *An Evolutionary Approach to Synthetic Biology: Zen and the Art of Creating Life*, Artificial Life Vol.1, 179--209.

[3] Ray, T. S. (2001). *Overview of Tierra at ATR*. Technical Information, No.15, Technologies for Software Evolutionary Systems. ATR-HIP. Kyoto, Japan.

[4] Ray, T. S., & Hart, J. (1998). *Evolution of differentiated multi-threaded digital organisms*. In: Adami, C., Belew, R.K., Kitano, H., and Taylor, C.E. (eds.): Artificial Life VI: Proceedings of the Sixth International Conference on Artificial Life, MIT Press, Cambridge, MA 295--304

[5] Suzuki H., Ono N., & Yuta K. (2003). *Several Necessary Conditions for the Evolution of Complex Forms of Life in an Artificial Environment*, Artificial Life Vol.9, No.2, 537-558