Progressive Evolution Model Using A Hardware Evolution System

Tomofumi Hikage†, Hitoshi Hemmi††, and Katsunori Shimohara†
†NTT Human Interface Laboratories
1-1 Hikarinooka Yokosuka-Shi
Kanagawa 239, JAPAN
††Evolutionary Systems Department,
ATR Human Information Processing Research Laboratories,
2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619-02, JAPAN

Abstract

A progressive evolution model is proposed in which evolution takes place stepwise to match environmental changes. A verification system of the model is constructed on a hardware evolution system called AdAM (Adaptive Architecture Methodology) in which each individual circuit takes parallel input sequences and operates accordingly. A measurement is designed to express "environmental complexity" which is suitable for such a parallel simultaneous operations scheme.

Simulations using an artificial ant problem, a modified John Muir Trail, show that in progressive evolution model circuits can evolve such that complex behaviors can be obtained easily.

1 Introduction

With the increased use of evolutionary methodology in the creation of adaptive systems increases, the application problem areas have become more and more complex and large scale. Techniques for coping with such scalability issues include the utilization of high speed mechanizms by which thousands of generations each with hundreds of individuals can be evaluated in a very short time. Hardware Evolution technologies are in this category, and systems using them are expected to be much more powerful than conventional evolutionary systems. The increase of speed with existing technologies, however, is linear, while problem complexity often increases polynomially or exponentially with their size. This means that more sophisticated speed-up techniques should be used.

The progressive evolution model discussed in this paper is a good candidate for such a technique. In this paper, we will show that in the model, evolution occurs more smothly compared to that in the conventional evolution model.

2 Hardware Evolutionary System: AdAM

The effectiveness of the progressive evolution model was verified on a hardware evolutionary system named AdAM[1]. We will briefly review this system. Figure 1 shows a system block diagram of the simulator version of AdAM.

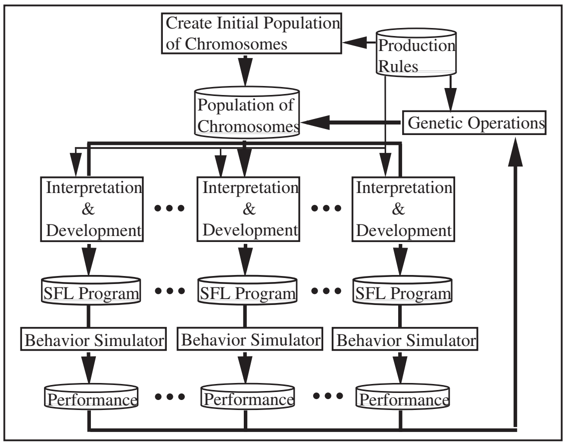

Figure 1: System Block Diagram of AdAM

The operation of the system is as follows. In Fig. 1, first, an initial chromosome population is randomly created according to a production rules set. These production rules are the Backus-Naur form of HDL (Hardware Description Language) named SFL, and chromosomes correspond to the parse tree of this language.

Each chromosome generates an individual SFL program through interpretation and development. Individual SFL programs, i.e., individual hardware behaviors, are simulated and evaluated by a certain fitness criterion. Selection and a set of genetic operations are performed on the chromosome population and produce next the generation of population. Such evolutionary processes eventually evolve individuals that encode for desirable hardware behaviors.

3 A Progressive Evolution Model

3.1 Basic idea

The progressive evolution model is a model in which evolution occurs in environments that are changing stepwise to the final target environment. The purpose of this model is to divide a problem "hurdle" into small steps, so that the evolutionary process can easily tackle each one.

In the conventional evolution mode with difficult problems, the evolutionary process must tackle plural steps all at one time. Sometimes evolution never occurs in this mode. This is because individuals who obtain some of the required steps are often evaluated no better than those who obtain nothing. The progressive evolution model is expected to resolve such difficulties. To make things easy and concrete, we will focus on one particular problem, which is briefly discribed in the next subsection.

3.2 The artificial ant problem

The artificial ant problem is a problem in which artificial ants adapt their behavior in a certain environment so that they gather food faster and more effectively; the best artificial ant is an ant that collects the most food in the fewest steps. Usually the environment is an arangement of food in a toroidal space of 32×32 cells (see Fig. 3). In our case, each artificial ant takes five input pins showing food existance and performs one of four actions.(Fig. 2)

3.3 A Measurement to express environmental complexity

As shown in Fig. 2, each ant has five inputs and two outputs. In the ant's inner mechanism, not only each input needs to be taken into account, but also arbitration among them is indispensable; an ant may encounter two different foods.

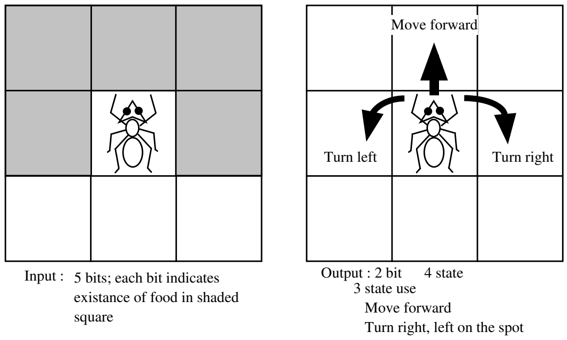

Figure 2: Artificial Ant

There are 96 combinations of inputs and actions. We call each such combination an action primitive.

Action primitives can be used to express the difficulty in adapting to an environment,i.e. the more action primitives the best ant must have, the harder the environment is to adapto to. We call this number the complexity of the environment. [1]
For instance,
{input from front : "move forward"}
{input from front, input from right : "turn right"}
are two examples of action primitives.

4 Progressivelly changing environments

To test the progressive evolution model, we designed a series of environments for artificial ants. Figure 7 shows the final target environment, and Figs. 3–6 show intermediate environments leading to the target environment. To take all the food in each environment, ants must obtain several action primitives. The target and the intermediate environments are designed so that the obtainment occurs incrementally.

The first intermediate environment in Fig. 3 requires four action primitives:
{input from front:"move forward"}
{input from right front:"move forward"}
{nothing:"move forward"}
{input from right:"turn right"}
So the complexity of this environment is 4.

The second intermediate environment in Fig. 4 requires three action primitives in addition to those of the first one:

[1] Of course, in sequential circuit phenomena, action primitive should be context sensitive; that is, action to the same inputs may vary according to situation. But the general idea of environment complexity should be useful for sequential circuits.

{input from left front:"move forward"}
{input from front and left front:"move forward"}
{input from left:"turn left"}
So the complexity of the environment is 7.

In the same way, the third intermediate environment in Fig. 5 requires two more action primitives in addition to those of the previous ones:
{input from front and right front:"move forward"}
{input from right and left front:"turn right"}
So the complexity of the environment is 9.

The fourth environment in Fig. 6 adds three action primitives:
{input from front and left front and left:"move forward"}
{input from front and right front and right:"turn right"}
{input from left and left front:"turn left"}
Thus, the complexity is 12.

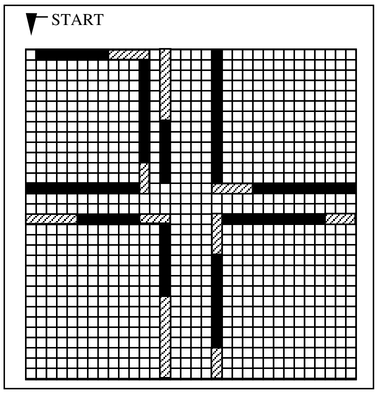

Figure 3: Progressive Environment 1

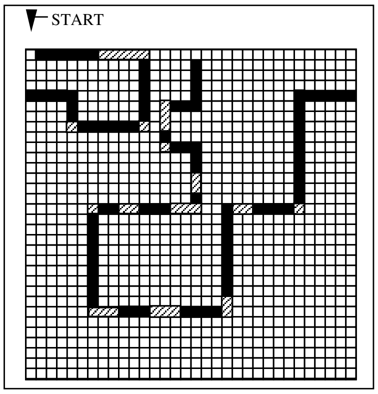

Figure 4: Progressive Environment 2

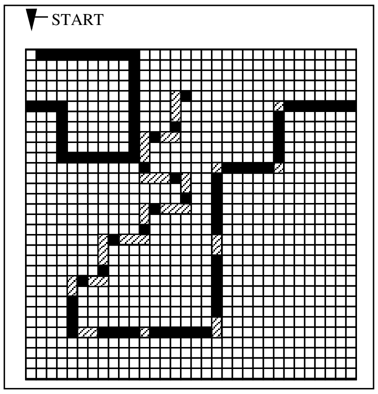

Figure 5: Progressive Environment 3

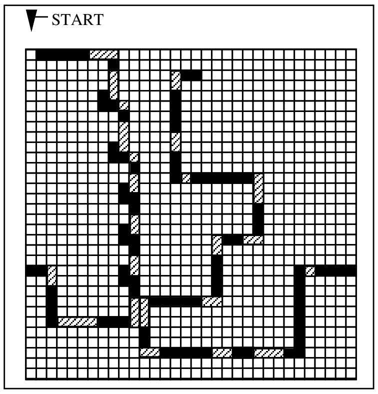

Figure 6: Progressive Environment 4

Finally, the target einvironment requires two more action primitives in addition to those of the previous ones:
{input from front and right front and left front : "move forward"}
{input from right and left : "turn right"}
And thus the complexity measure is 14.

The timings to exchange the environments are set five generations after the best individual appears in each environment. In addition, the experiments use an elite strategy so that the best individuals spread through the population and most of the circuits adapt to the environment during the five generations.

5 Result

The experimental result obtained in early generations with the progressive evolution model is shown in Fig. 8. In this figure, the best individuals in the first, second, and third intermediate environments appear in the lst, 11th, and 18th generations respectively. For

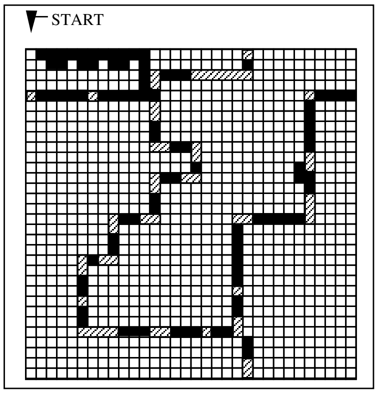

Figure 7: Target Environment

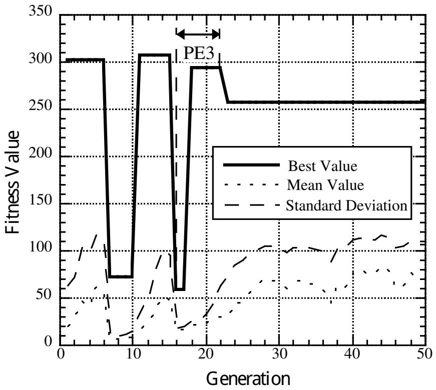

Figure 8: Result of Progressive Evolution Model (early generation)

comparison, Fig. 9 shows the result obtained for evolution in the third environment with an ordinary evolution model. In this figure, the best individual appears in the 77th generation, indicating that a progressive model is more efficient than an ordinary model.

Figure 10 shows the fitness transition for the generations as a whole. It takes about 7,900 generations to produce the best individual in the fourth environment. The reason for this result is considered to be that the action primitives which the environment requires are much more difficult to obtain than the previous ones. But once the primitives are obtained, only one generation is needed to obtain another two (equally difficult) action primitives and adapt to the final environment.

6 Conclusion

This paper has proposed a progressive evolution model, and a complexity measurement suitable for hardware evolution. Experiments with artificial ant problems have demonstrated the effectiveness of the progressive model.

References

[1] H.Hemmi, M.Jun'ichi, K.Shimohara, "Development and evolution of hardware behaviors.", In *Artificial Life IV*,1994

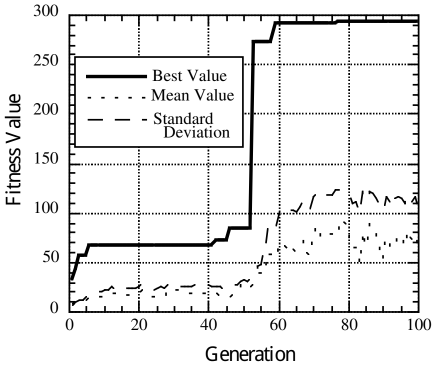

Figure 9: Result of evolution in only PE 3

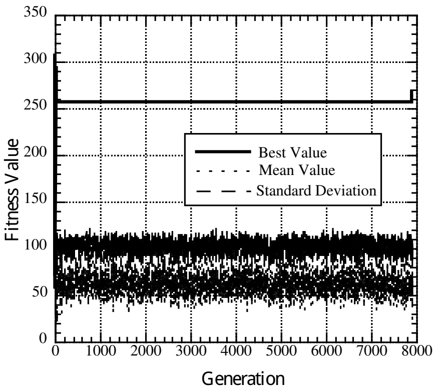

Figure 10: Result of Progressive Evolution Model