Schema Analysis on Co-Evolutionary Algorithm

공진화에 있어서 스키마 해석

Hyo-Byung Jun* and Kwee-Bo Sim

Robotics and Intelligent Information System Laboratory
Dept. of Control and Instrumentation Engineering, Chung-Ang University
221, Huksuk-Dong, Dongjak-Ku, Seoul 156-756, Korea
Tel:+82-2-820-5319, Fax:+82-2-817-0553
E-mail:kbsim@cau.ac.kr, URL:http://rics.cie.cau.ac.kr

ABSTRACT

The theoretical foundations of simple genetic algorithm(SGA) are the Schema Theorem and the Building Block Hypothesis. Although SGA does well in many applications as an optimization method, still it does not guarantee the convergence of a global optimum in GA-hard problems and deceptive problems. Therefore as an alternative scheme, there is a growing interest in a co-evolutionary system, where two populations constantly interact and cooperate each other. In this paper we show why the co-evolutionary algorithm works better than SGA in terms of an extended schema theorem. Also the experimental results show a co-evolutionary algorithm works well in optimization problems.

I. Introduction

Recently artificial life concept was proposed by C. Langton and has become one of the most popular research area as a solution of intelligent information processing system under uncertain, complex and dynamic environment. The evolutionary computation based on the natural selection theory plays an important role in artificial life. The concept of natural selection has influenced our view of biological systems tremendously. Genetic Algorithms (GAs) are computational models of living system's evolution process and population-based optimization methods. GAs can provide many opportunities for obtaining a global optimal solution, but the performance of a system is deterministic depending on the fitness function given by a system designer. Thus GAs generally work on static fitness landscapes. But natural evolution works on dynamic fitness landscapes that change over evolutionary time as a result of co-evolution. And co-evolution between different species or different organs results in the current state of complex natural systems. In this point, there is a growing interest in co-evolutionary systems, where two populations constantly interact and co-evolve in contrast with traditional single population evolutionary algorithms.

In this paper, we derive an extended schema theorem associated with a host-parasite co-evolutionary algorithm, where the fitness of a population changes according to the evolutionary process of the other population. Host-parasite co-evolutionary algorithm has two different, still cooperatively working, populations called as a host-population and a parasite-population, respectively. The first one is made up of the candidates of solution and works the same with conventional genetic algorithm. The other one, a parasite-population, is a set of schemata, which is to find useful schemata called "Building Block"[1][2]. Using the conventional genetic algorithm the host-population is evolved in the given environment, and the individual of the host-population is parasitized by a schema in the parasite-population evolving to find useful schemata for the host population. We show why a co-evolutionary algorithm works better than SGA and demonstrate the comparative results in solving a deceptive function. Also we construct the schema return map to compare the schema dynamics.

II. Co-Evolutionary Algorithm and Extended Schema Theorem

The theoretical foundations of genetic algorithms rely on a binary string representation of solutions, and a notion of a schema. Generally the combined effect of selection, n points crossover, and mutation on the expected number of a schema is formulated by[1][2]:

$$m(H,k+1) \geq m(H,k) \cdot \frac{f(H,k)}{\bar{f}(k)} \cdot \left(1 - p_c \frac{{}_{l}C_n - {}_{l-1-\delta(H)}C_n}{{}_{l-1}C_n} - p_m o(H)\right) \quad (1)$$

where $f(H, k)$ is the average fitness of all strings in the population matched by a schema H, $\bar{f}(k)$ is the average fitness of all individuals in the population at generation k, $m(H,k)$ is the number of instances of a schema H, $o(H)$ is the order of a schema, $\delta(H)$ is the defining length of a schema, and p_c, p_m are crossover rate and mutation rate, respectively.

This is known as the Schema Theorem and means that the short, low-order, above-average schema, called as the Building Blocks, would receive an exponentially increasing number of strings in the next generations. However if there does not exist a solution in the Building Blocks, simple genetic algorithm might fail to find that solution. The deceptive function is most well known as a problem violating above theorem. T. Kuo and S.Y. Hwang[3] showed that disruptive selection works better than directional selection on the deceptive functions.

In the other hand, natural evolution works on the fitness landscapes that changes over the evolutionary time. From this point of view, co-evolution algorithms have much attractions in intelligent systems.

Predator-prey relation is the most well-known example of natural co-evolution. Hillis[4] proposed this concept with a problem of finding minimal sorting network for a given number of data. And co-evolution between neural networks and training data was proposed in the concept of predator and prey[5]. And fitness measure in co-evolution is studied in terms of dynamic fitness landscape. L. van Valen, a biologist, has suggested that the "Red Queen effect" arising from co-evolutionary arms races has been a prime source of evolutionary innovations and adaptations[6].

Symbiosis is the phenomenon in which organism of different species live together in close association, resulting in a raised level of fitness for one or more of the organisms. In contrast of predator-prey, this symbiosis has cooperative or positive aspects between different species. Paredis[7] proposed a symbiotic co-evolution in terms of SYMBIOT, which uses two co-evolving populations. One population contains permutations (orderings), the other one consists of solution candidates to the problem to be solved.

And another approach to symbiotic co-evolution is host-parasite relation. Just as do other co-evolutionary algorithms, two co-evolving populations are used. One is called host population which consists of the candidates of solution, the other contains schemata of the solution space. This idea is based on the Schema Theorem and the Building Block hypothesis. The parasite-population searches useful schemata and delivers the genetic information to the host-population by parasitizing process.

In the context of a computational model of co-evolution, the parasitizing means that the characters of a string are exchanged by the fixed characters of a schema. And the other positions of the string, i.e., the same positions of don't-care symbol in the schema, hold their own values. The fitness F_y of a string y in the parasite-population is determined as follows:

$$\hat{f}_{iy}(k) = \max[0, f(\hat{x}_{iy}, k) - f(x_i, k)] \quad (i = 1, \cdots, n) \quad (2)$$

$$F_y = \sum_{i=1}^{n} \hat{f}_{iy} \quad (3)$$

where x_i is a randomly sampled string in the host-population, $\hat{x}_{iy}$ is a parasitized string that is a sampled string after parasitized by a schema y, $f(x_i, k)$ is the fitness of a string x_i at generation k, and $f(\hat{x}_{iy}, k)$ is the fitness of a string $\hat{x}_{iy}$.

By exchanging a string x_i for $\hat{x}_{iy}$ which is a string having maximum value of $\hat{f}_{iy}$, still one of the strings parasitized by a schema y, the genetic information acquired by parasitizing is delivered to the host-population.

If a string y in the parasite-population represents a schema H, it is clear that the above parasitizing process can be interpreted, in the context of useful schemata, as a process of increasing the number of instances of a schema H in the host-population. When the co-evolution is considered the number of instances $m'(H, k)$ of a schema H in the host-population at the generation k is expressed by

$$m'(H, k) = m(H, k) + \hat{m}(H, k) \quad (4)$$

where $m(H, k)$ is the original number of instances of a schema H in the host-population. And $\hat{m}(H, k)$ is the increased number of instances by the parasitizing process and can be stated as follows:

$$\hat{m}(H, k) = \frac{1}{2} \sum_{i=1}^{n} \{ sgn[f(\hat{x}_{iH}, k) - f(x_i, k)] + 1 \} \quad (5)$$

where $sgn(u)$ is a sign function that equals +1 for positive u and -1 for negative u. Note that since we focus on the newly generated instances after parasitizing, the case that x_i is identified with $\widehat{x_{iH}}$ is excluded from the equation (5).

Also we can formulate the fitness of a schema H associated with host-parasite co-evolution from its definition. Let us denote by $f'(H,k)$ the fitness of a schema H after parasitized. That is

$$f'(H,k)=\frac{\sum_{x_i\in I_H} f(x_i,k)+\sum_{x_i\in I_H} f(\widehat{x_{iH}},k)}{m(H,k)+\widehat{m}(H,k)} \qquad (6)$$

where I_H is a set of instances of a schema H at the generation k and $\widehat{I_H}$ is a index set of increased instances of a schema H after parasitized. Combining the above equations, the schema theorem can be rewritten by

$$m(H,k+1)\geq m'(H,k)\cdot\frac{f'(H,k)}{\bar{f}(k)}\cdot\left(1-p_c\frac{{}_lC_n-{}_{l-1-\delta(H)}C_n}{{}_{l-1}C_n}-p_m o(H)\right). \qquad (7)$$

Since the fitness of a schema H is defined as the average fitness of all strings in the population matched by that schema H, the fitness $f'(H,k)$ can be approximated by $f'(H,t)\simeq f(H,t)$. Especially, if the number of strings in the host-population $N_H\gg n$, the above approximation makes sense for the large number of generation sequences[1].

Consequently we obtain an extended schema theorem associated with host-parasite co-evolution:

$$m(H,k+1)\geq[m(H,k)+m'(H,k)]\cdot\frac{f(H,k)}{\bar{f}(k)}\cdot\left(1-p_c\frac{{}_lC_n-{}_{l-1-\delta(H)}C_n}{{}_{l-1}C_n}-p_m o(H)\right) \qquad (8)$$

Compared with the original Schema Theorem in equation (1), the above equation means that the short, low-order, above-average schema H would receive an exponentially increasing number of strings in the next generation with higher order than SGA. Additionally the parasitizing process gives more reliable results in finding an optimal solution. When the schema containing a solution does not exist in the population, SGA may fail to find global optima. In the other hand, because the useful schema can be found by the parasite-population, co-evolution gives much more opportunities to converge to global optima.

III. Experiments

A deceptive function is a function for which SGA is prone to be trapped at a deceptive local optimum. In this section, we consider only a false-peaks function which has several deceptive peaks. If there are ten boolean variable $x_1x_2\cdots x_{10}$ which are used as a string, the function and its fitness are defined as

$$f=(x_1\wedge x_2\wedge\cdots\wedge x_{10})\vee(x_1\wedge\overline{x_1}\wedge\overline{x_2}\cdots\wedge\overline{x_{10}}) \qquad (9)$$

$$Fit(f)=\max\left\{\sqrt{\frac{x_1^2+\cdots+x_{10}^2}{10}},\ \sqrt{\frac{x_1^2+(1-x_1)^2+\cdots+(1-x_{10})^2}{11}}\right\}. \qquad (10)$$

We plot the landscape of its fitness function where the horizontal axis is the decimal number of the binary string. As shown in Fig. 1, there is one optimal solution which are all 1's. But it is easy to see that there are several deceptive local optima including all 0's.

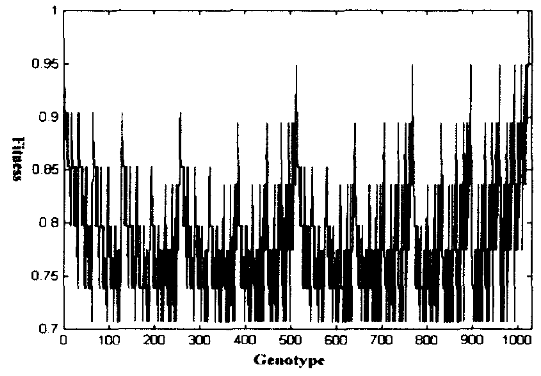


Fig. 1. Landscape of a false-peaks function in search space

The population size of SGA is set for 80, the crossover rate is 0.6, and the mutation rate is set for 0.02. And the host and parasite-population sizes of co-evolution are set for 20, and the same rates of crossover and mutation are used. And the sampling size is set for 3.

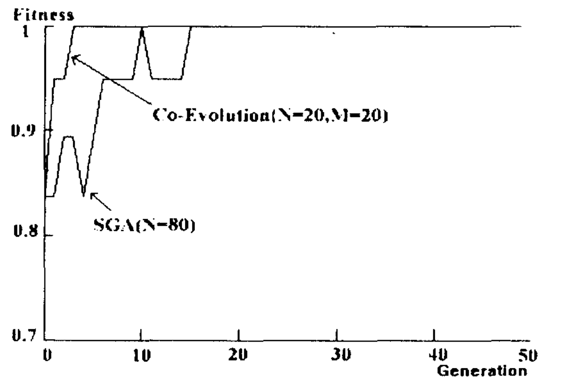
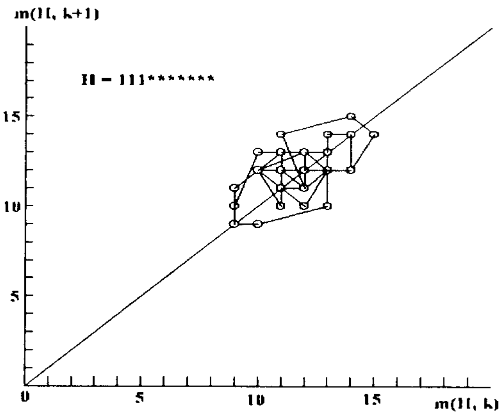


Fig. 2. Fitness changes

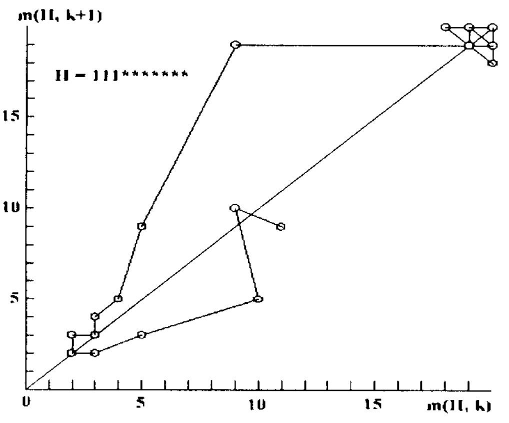
The results are plotted in Fig. 2 which shows the best individual's fitness versus generation when searched by SGA and the host-parasite co-evolution, respectively. In this results, we can see that the host-parasite co-evolutionary algorithm converges to an optimal solution more rapidly than SGA.

Fig. 3 shows the return map of the useful schema which starts with 1. As shown in Fig. 3 (a), the instance number of the useful schema does not converge to a point, but repeats increasing and decreasing along the stable line in SGA. This is caused by false peaks which start with 0. Specially, if the initial population consists of the deceptive schemata mainly, frequently SGA fail to find an optimal solution.

In the other hand, the host-parasite co-evolution gives more reliable guarantee of the convergence irrespective of the initial population. As shown in Fig. 3 (b), even though there exists a small number of the useful schema 111******* in the population the number of instances of that schema increase exponentially. Also it shows that the instances number of that schema converges to the maximum number which is the population size. This results imply that the parasitizing process plays an important role in escaping the local optima and reaching global optima rapidly.



(a) SGA



(b) Co-evolution
Fig. 3. Schema return map

IV. Conclusions

In this paper we derived an extended schema theorem associated with host-parasite co-evolution and showed some comparative results. Even though the original Schema Theorem and the Building Block Hypothesis give theoretical foundations to SGA, some problems, such as deceptive functions, are hard to be solved by SGA. But co-evolutionary algorithm where two populations constantly interact and evolve cooperatively in contrast with traditional single population evolutionary algorithms solved those problems more reliably. Also it gives much more chances to find global optima than SGA because the parasite-population searches the schema space.

Acknowledgement

This work was supported by a grant No. KOSEF 96-01-02-13-01-3 from Korea Science and Engineering Foundation.

References

[1] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Third Edition, Springer-Verlag, pp. 265-281, 1995.

[2] J. Kinzel, F. Klawonn, R. Kruse, " Modifications of Genetic Algorithm for Designing and Optimizing Fuzzy Controllers," *ICEC* '94, Vol. 1, pp. 28-33, 1994.

[3] T. Kuo and S. Y. Hwang, "A Genetic Algorithm with Disruptive Selection," *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 26, No. 2, pp.299-307, 1996.

[4] W. Daniel Hillis, "Co-Evolving Parasites Improve Simulated Evolution as an Optimization Procedure," *Artificial Life II*, Vol. X, pp.313-324, 1991.

[5] D.W. Lee, H.B. Jun, K.W. Sim, "A Co-Evolutionary Approach for Learning and Structure Search of Neural Networks," *Proc. of KFIS Fall Conference* '97, Vol. 7, No. 2, pp. 111-114, 1997.

[6] D. Cliff, G. F. Miller, "Tracking The Red Queen: Measurements of adaptive progress in co-evolutionary simulations," *COGS Technical Report CSRP363*, University of Sussex, 1995.

[7] Jan Paredis, "Co-evolutionary Computation," *Artificial Life*, Vol. 2, No. 4, pp.355-375, 1995.