Appl. Math. and Comp. Sci., 1996, vol.6, No.3, 415–429

THE ROLE OF VIRUS INFECTION IN A VIRUS-EVOLUTIONARY GENETIC ALGORITHM

NAOYUKI KUBOTA*, KOJI SHIMOJIMA**
TOSHIO FUKUDA*

This paper deals with a genetic algorithm based on virus theory of evolution (VEGA). The VEGA realizes horizontal propagation and vertical inheritance of genetic information in a population with virus infection operators and genetic operators. The main operator of the VEGA is a reverse transcription one, which plays the role of a crossover and a selection simultaneously. Therefore this virus infection with reverse transcription is the key mechanism of the VEGA. The convergence and genetic diversity of the VEGA depend on the frequency of the virus infection. In this paper, we apply the VEGA to function optimization problems and a knapsack problem, and discuss the effectiveness of the virus infection through numerical simulation results.

1. Introduction

Living beings in nature evolve and adapt to their external environments. If it is possible to simulate evolution on a computer, then we can realize an adaptive system. Evolutionary computation is a field of simulating evolution on a computer (Fogel, 1995b). In evolutionary computation, stochastic optimization methods simulating the process of natural evolution are divided into three main categories: genetic algorithms (GA) (Holland, 1992), evolutionary programming (EP) (Fogel, 1995b), and evolution strategy (ES) (Rechenberg, 1994). These algorithms are fundamentally iterative generation and alternation processes operating on a population of candidate solutions. Furthermore, these algorithms can be classified into two types from the viewpoint of a representation method. The EP and the ES, which are called evolutionary algorithms (EAs), mainly operate on phenotype directly. The main operator is a mutation using a Gaussian random variable with zero mean. On the other hand, the GA mainly operates on genotype as strings or bits. The main operator is a crossover between candidate solutions. A standard GA is composed of selection, crossover, and mutation (Goldberg, 1989). The schemata theorem is well-known as a fundamental theorem of the GA (Goldberg, 1989). The increase of effective schemata enables the efficient search of the solution space and makes all the population evolve toward optimal solutions.

* Dept. of Micro System Engineering, Nagoya University, 1 Furo-cho, Chikusa-ku, Nagoya 464- 01, Japan, e-mail: kubota@robo.mein.nagoya-u.ac.jp, fukuda@mein.nagoya-u.ac.jp
** National Industrial Research Institute of Nagoya, 1-1 Hirate-cho, Kita-ku, Nagoya 462, Japan, e-mail: koji@nirin.go.jp

On the other hand, there are two main methods for generating new candidate solutions to solve an optimization problem. One is based on a stochastic method and the other is based on local information peculiar to the optimization problem. A piece of local information is often useful for solving the optimization problem. For example, in the case of a traveling salesman problem, an optimization method based on local subtours is easier to solve than the one without consideration of subtours. The GA fundamentally belongs to the former method, but the GA has a characteristic termed 'implicit parallelism' (Goldberg, 1989). Crossover and selection deal indirectly with schemata, though they sometimes generate and increase effective schemata as a result. The increase of a schema is equal to the increase of local information in a population. The effectiveness of the GA has been demonstrated in various fields (Baba and Kubota, 1994; Bramlette, 1991; Fogel, 1995a; Goldberg, 1989; Kubota *et al.*, 1994; Mühlenbein *et al.*, 1991; Shimojima *et al.*, 1994; Syswerda, 1991; Tamaki *et al.*, 1994; Whitley, 1989), but the GA has a problem of premature local convergence that occurs when a genetic diversity lacks in a population. A proportional selection scheme in the GA often causes the premature convergence because of selecting an individual with high fitness value many times. Therefore, a selection must realize two different aims: 1) to select effective solutions for increasing effective schemata in a population, 2) to maintain the genetic diversity for generating new candidate solutions. The design of selection scheme is very important, since the selection scheme determines the direction of evolution. Selection schemes such as a ranking selection scheme have been proposed to realize these aims. However, the proportional selection scheme increases not only effective schemata, but also ineffective schemata simultaneously. In order to operate only on effective schemata, we have proposed a virus-evolutionary genetic algorithm (VEGA) based on virus theory of evolution (Kubota *et al.*, 1996). The VEGA has two populations: a host population and a virus population. The virus population performs two virus infection operators. One is a reverse transcription overwriting a substring of a virus individual onto the string of a host individual as horizontal propagation. The other is a transduction generating a new virus individual by transducing from a host individual. The VEGA deals directly with schemata and their characteristics are as follows:

- Reverse transcription increases directly effective schemata.
- Reverse transcription generates directionally new host individuals.
- Transduction changes virus individuals every generation.

The generation and the horizontal propagation of effective schemata are very important, since the local information is often useful for solving the optimization problem. Coevolution of the host population and the virus population permits quick solution of the optimization problem. The performance of the VEGA depends on the virus infection operators, since the number of reverse transcription of a virus individual to host individuals determines the frequency of the horizontal propagation in the host population. This paper discusses the effectiveness of the virus infection operators through some numerical simulations of a knapsack problem and function optimization problems.

This paper is organized as follows. Section 2 presents the genetic algorithm based on virus theory of evolution. The virus infection operators are defined and incorporated into the GA. Section 3 presents applications to a knapsack problem and function optimization problems together with simulation results.

2. Virus-Evolutionary Genetic Algorithm

2.1. Virus Theory of Evolution

How have living beings evolved in nature? Though the fact of evolution is certain, it is difficult to explain the process of evolution well. With the recent progress in molecular biology, various theories of evolution, such as Neo-Darwinism, neutral theory of molecular evolution, Imanishi's evolutionary theory, serial symbiosis theory, and virus theory of evolution, have been proposed (Ridley, 1993). However, most of evolutionary theories cannot explain all evolutionary evidences, though they explain the process of evolution in a sense.

Virus theory of evolution is based on the idea that virus transduction is a key mechanism for transporting segments of DNA across species (Anderson, 1970). Here the transduction means the genetic modification of a bacterium by genes from another bacterium carried by a bacteriophage (Primrose and Dimmock, 1980). Most of viruses in nature can easily cross species barriers and are often transmitted directly from individuals of one phylum to another by horizontal propagation. Furthermore, whole virus genomes may be incorporated into germ cells and transmitted from generation to generation as vertical inheritance.

2.2. Virus-Evolutionary Genetic Algorithm Architecture

A virus-evolutionary genetic algorithm (VEGA) simulates evolution with both horizontal propagation and vertical inheritance (Fig. 1). The VEGA has two populations: a host population and a virus population. Here they are defined as a set of candidate solutions and a substring set of the host individuals, respectively. As mentioned before, a virus has a capability to transmit segments of DNA between species. Therefore, the virus infection realizes the horizontal propagation in the host population. To incorporate the virus infection mechanism, we adopt a steady-state genetic algorithm (SSGA) (Syswerda, 1991). In general, the SSGA replaces a pair of individuals with new individuals generated by the crossover every generation. The procedure of the VEGA is as follows:

```
Initialization
    repeat
        Selection
        Crossover
        Mutation
        Virus_infection
    until Termination_condition = True
end.
```

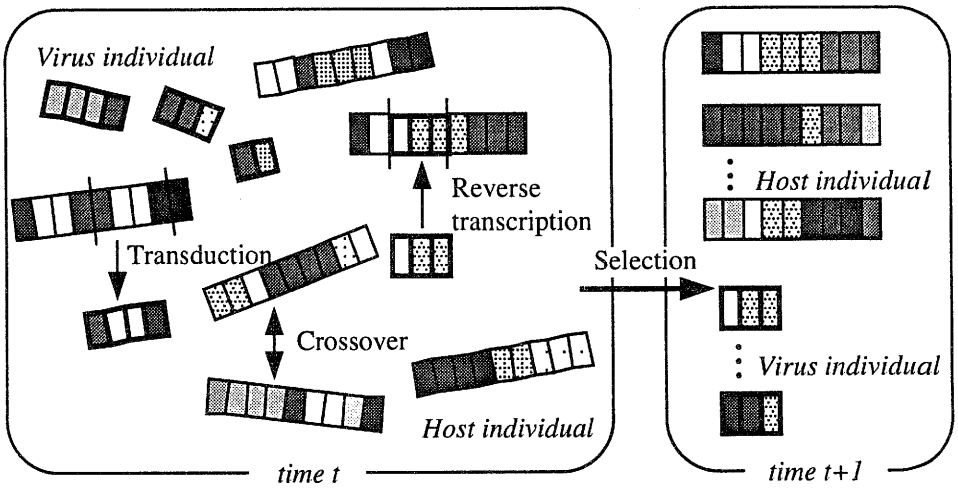


Fig. 1. Virus evolutionary genetic algorithm.

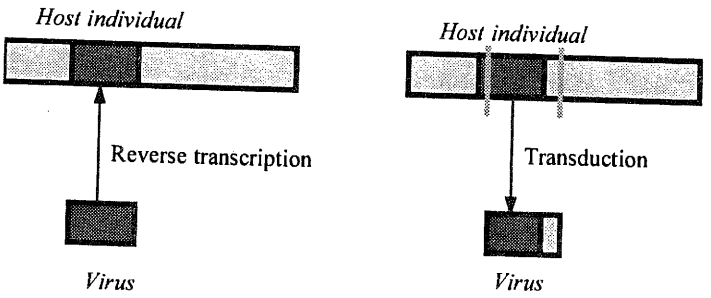
Initialization randomly generates an initial host population, and then a virus individual is generated as a substring of a host individual. 'Delete least fitness' (Syswerda, 1991) is used as the selection scheme. Crossover and mutation are genetic operators dependent on the optimization problem. The string length of the host individual is basically predefined as a constant. The length of a virus individual, which is defined as a variable, extends with evolution of the host population.

2.3. Virus Infection Operators

This section defines virus infection operators. There are many characteristics regarding a virus infection. In this paper, we assume that the main process of a virus infection is horizontal propagation of a substring among host individuals. Therefore a virus transduces the genes from a host individual and transcribes to another host individual. The VEGA has two virus infection operators as follows:

- Reverse transcription operator: The virus transcribes its substring on the string of a host individual (Fig. 2(a)).
- Transduction operator: The virus transduces a substring from a host individual. As its fundamental operation, the virus takes out a substring in addition/reduction to some genes on the host string (Fig. 2(b)).

The number of infection times of each virus is controlled under its virus infection rate. Each virus has a parameter, $fitvirus_i$, regarding the virus infection. We assume that $fithost_j$ and $fithost'_j$ are the fitness value of host individual j before and after



(a) Reverse transcription (b) Transduction

Fig. 2. Virus infection operators.

the infection, respectively. The $fitvirus_{ij}$ denotes the difference between $fithost_j$ and $fithost'_j$, which is equal to the value obtained by infecting the host individual:

$$fitvirus_{ij} = fithost'_j - fithost_j \tag{1}$$

$$fitvirus_i = \sum_{j \in S} fitvirus_{ij} \tag{2}$$

where i is the virus number and S is a set of host individuals which are infected by virus i. Furthermore, each virus has the life force as follows:

$$life_{i,t+1} = r \cdot life_{i,t} + \alpha \cdot fitvirus_i \tag{3}$$

where t, r and α mean the generation number, the life reduction rate and a coefficient, respectively.

The procedure of virus infection is shown in Fig. 3. First, a virus does the reverse transcription to an individual selected randomly out of the host population. Next, the VEGA evaluates the fitness of the infected host individual and calculates $life_{i,t+1}$. If $life_{i,t}$ takes a negative value, the virus individual transduces a new substring with the transduction from a randomly selected host individual. Otherwise, the virus individual transduces a partially new substring from that of the infected host individuals with the transduction.

If the virus infection rate is high, the times of the virus infection would increase, i.e. the VEGA mainly performs the local search when the virus population has effective schemata. Otherwise, the VEGA mainly performs the global search with genetic operators. In this way, the VEGA can self-adaptively change the searching ratio between the local search and the global search according to the state of the host and virus populations.

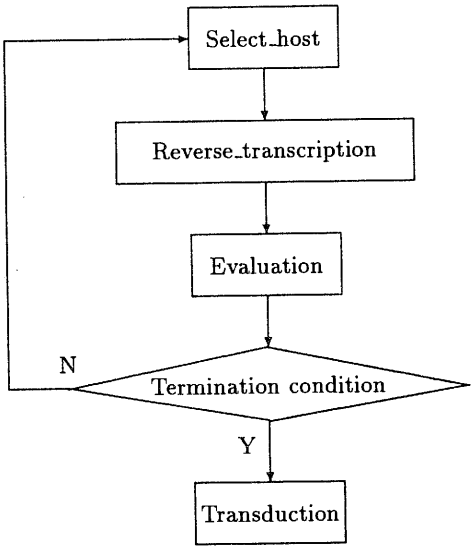


Fig. 3. Procedure of virus infection.

3. Numerical Simulation

3.1. Knapsack Problem

The knapsack problem is an integer programming problem with 0–1 variables (Ecker and Kupferschmid, 1988). This problem is represented only by the values 0 or 1. Suppose that n items are to be selected for carrying in a knapsack. Item i has value v_i and weight w_i. Our aim in the knapsack problem is to select items in order to maximize their total value where their total weight is less than or equal to W. Thus the objective function to be maximized is as follows:

$$\sum_{i=1}^{n} v_i x_i \tag{4}$$

subject to

$$\sum_{i=1}^{n} w_i x_i \leq W, \qquad x_i = 0, 1, \quad i = 1, \ldots, n$$

The application of the GA to the knapsack problem is based on the following idea. A string is represented by a binary code. For example, the string '01100' means that the second and third items are selected. The fitness value of a host individual is defined as the total value of the selected items, when the weight condition is fulfilled. As genetic operators, we use a uniform crossover (Goldberg, 1989) and a bit mutation. The uniform crossover generates new individuals according to a randomly generated mask pattern. As mentioned before, a virus individual can transmit a substring

between host individuals. A substring of a virus individual consists of three characters $\{0, 1, *\}$ and is the same length as that of the host individual. The character '*' denotes the 'don't care' mark. A virus individual does not perform the reverse transcription in the same position where there is a '*'. In this case, the length of the virus is constant, but the order of the virus is variable. Figure 4 shows an example of the reverse transcription. The reverse transcription overwrites the substring of the virus individual on a randomly selected host individual. The transduction to evolve virus individuals has two types of operators (Fig. 5). The first one is to copy genes from a host individual according to a copy rate per gene. The other is to replace some genes with character '*' according to a cut rate per gene. If the virus improves the fitness values of host individuals, the copy operator is performed with the transduction rate. Otherwise, the replacement operator is performed. An initial virus population is generated from the host population with the use of the transduction operator.

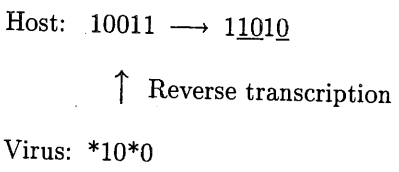
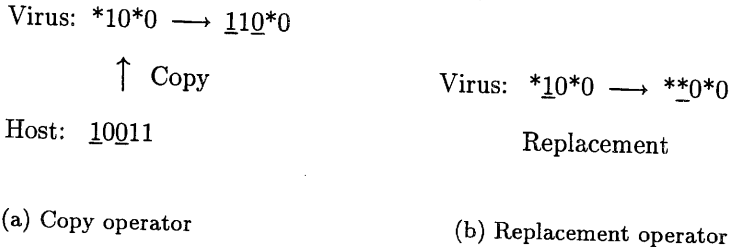


Fig. 4. Reverse transcription operator.



(a) Copy operator (b) Replacement operator

Fig. 5. Transduction operator.

In the knapsack problem, the number of items is 50. The maximal weight to be selected, W, is 80% of total weight of all items. Table 1 shows the parameters of the SSGA, VEGA and virus infection. The number of evaluations is used in the numerical simulation so that the search time of VEGA be equal to that of SSGA. Here the search time means the sum of the times of crossover and reverse transcription. The infection rate means the rate that a virus transcribes to a host population. The transduction rate means the rate at which a virus transduces after infection.

Tab. 1. Parameters of SSGA, VEGA, and virus infection.

	SSGA	VEGA
Population size	100	100
String length	50	50
Crossover rate	0.8	0.8
Mutation rate	0.001	0.001
Evaluations	10000	10000

	Virus infection
Virus population size	10
Life reduction (r)	0.9
Max infection rate	0.1
Initial infection rate	0.05
Transduction rate	0.6
Copy/Cut rate	0.05

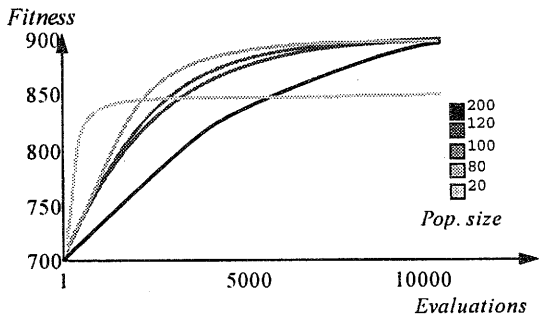


Fig. 6. Simulation results of the knapsack problem (SSGA).

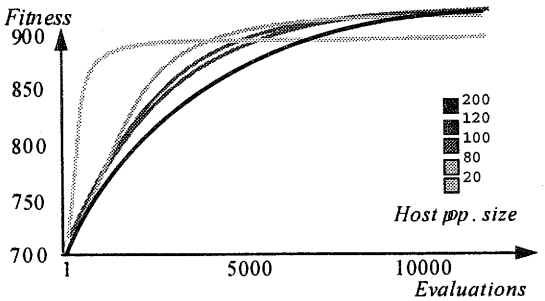


Fig. 7. Simulation results of the knapsack problem (VEGA), virus population size= 10% of host population, maximal infection rate= 0.1.

Figures 6 and 7 show simulation results of the SSGA and VEGA, respectively. The fitness value in each figure is the average of 50 trials. On the whole, the convergence of the VEGA is faster than that of the SSGA. Furthermore, when the population size is small in both the SSGA and VEGA, on the average the SSGA and VEGA attain local maxima because the premature convergence often occurs. On the contrary, the convergence of a large population size is slower, but the VEGA and SSGA reach global maxima. Figure 8 shows the comparison of on-line performance which is the average of the highest fitness value at each generation. The on-line performance of the VEGA outperforms the SSGA in all population sizes. Table 2 shows

the average of fitness values obtained after 10000 evaluations. The VEGA, whose population size is 100, attains the highest average of fitness values.

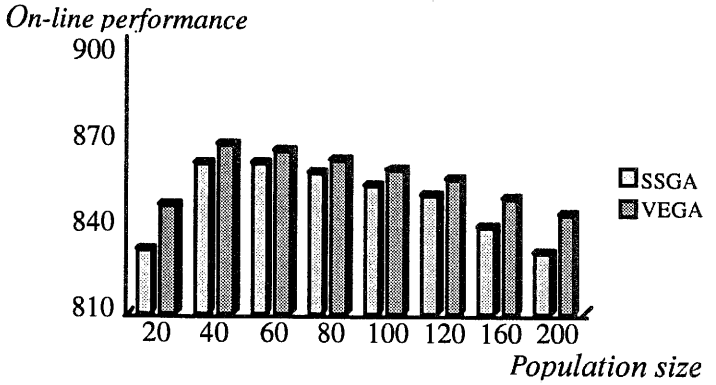


Fig. 8. On-line performance of the knapsack problem.

Tab. 2. Maximal fitness values obtained at the last generation.

Host pop. size	20	40	60	80	100	120	160	200
SSGA	848.74	879.78	885.23	887.82	888.80	889.11	888.66	887.15
VEGA	857.12	885.72	888.94	889.10	889.43	889.02	888.48	887.20

Figure 9 shows simulation results concerning virus population sizes, where the host population size and the maximal infection rate are 100 and 0.1, respectively. The larger virus population size, the slower the convergence is. Next, we consider the case of a large virus population size. Figure 10 shows simulation results concerning high infection rates, where the host and virus population sizes are 100 and 200, respectively. When the infection rate is high, the virus infects almost all host individuals. As a result, a premature convergence is easy to occur.

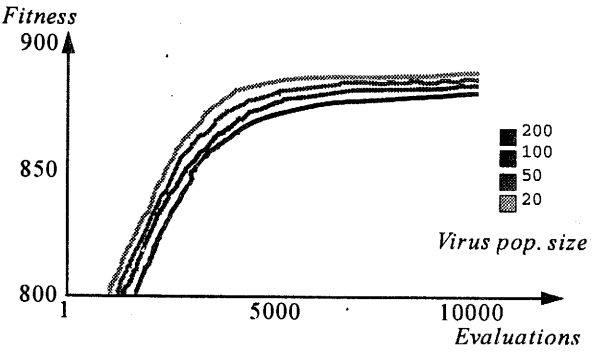


Fig. 9. Comparison of simulation results concerning the virus population size, host population size=100, maximal infection rate=0.1.

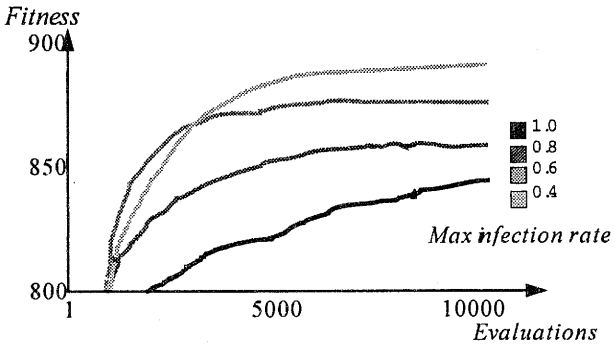


Fig. 10. Comparison of simulation results concerning the virus population size, host population size=100, virus population size=200.

Next, in order to discuss optimal parameters of the virus infection, we carry out some simulations concerning a small virus population size. Figure 11 shows the comparison of simulation results in the case of 60 host individuals. The VEGA attains the best fitness value when the virus population size and the maximal infection rate are 6 and 0.1, respectively.

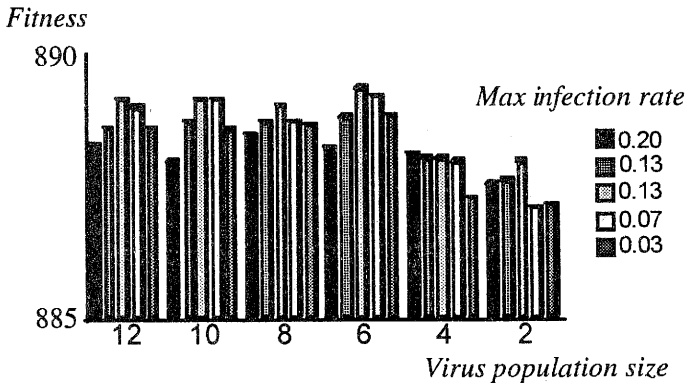


Fig. 11. Comparison of simulation results concerning the small virus population size, host population size=60.

3.2. Function Optimization Problem

Function optimization problems have frequently been applied as benchmark tests for GAs (Bramlette, 1991; Goldberg, 1989; Mühlenbein *et al.*, 1991). In this section, we apply the VEGA to two types of function optimization problems. Case 1 is as follows:

$$f(x,y) = x^2 + 2y^2 - 0.3\cos(3\pi x) - 0.4\cos(4\pi y) + 0.7 \tag{5}$$

where $-1.0 < x, y < 1.0$. This function is highly multimodal. The aim of this problem is to minimize the objective function. The global minimum is at $x = 0$, $y = 0$, which produces $f(x,y) = 0$. Figure 12 shows the shape of this objective function.

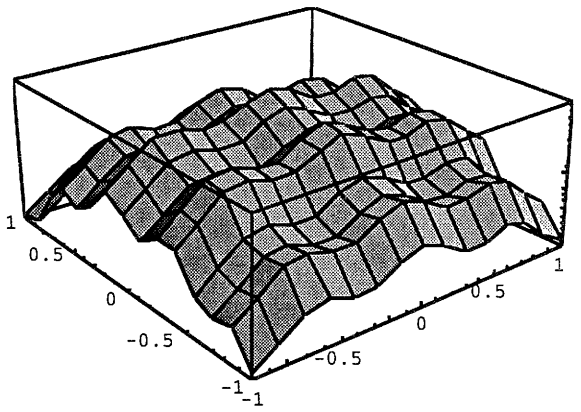


Fig. 12. 3D-space characterized by objective function $f(x, y)$ (the negative of $f(x, y)$ is depicted).

We use the binary code as the representation of genotype. A string includes two variables in this objective function. Decoding the string, we obtain two integer numbers, and substituting these numbers into X in eqn. (6), we obtain two variables:

$$z = \frac{X - Z}{Z} \tag{6}$$

Here Z denotes half the value of the maximal integer of X represented by the binary code. A uniform crossover and a bit mutation are also used as genetic operators.

Figures 13 and 14 show the simulation results of Case 1 of the SSGA and VEGA, respectively. The fitness value in each figure is the average of 50 trials. We obtain the same result as in the knapsack problem. The convergence of the VEGA is faster than that the SSGA, on the whole. Table 3 shows the average of fitness values obtained after 10000 evaluations. Since Case 1 is easy to solve, we apply a more difficult function as Case 2:

$$g(x) = 5A + \sum_{i=1}^{5} \Big(Ax_i - A\cos(2\pi x_i) \Big) \tag{7}$$

Table 4 shows the average of fitness values after 20000 evaluations of 50 trials. When the host population size is 100, the VEGA attains the best value in these simulation results. Figure 15 shows the comparison of simulation results concerning the virus population size. When the virus population size and the maximal infection rate are 10 and 0.1, respectively, the VEGA attains the best fitness value.

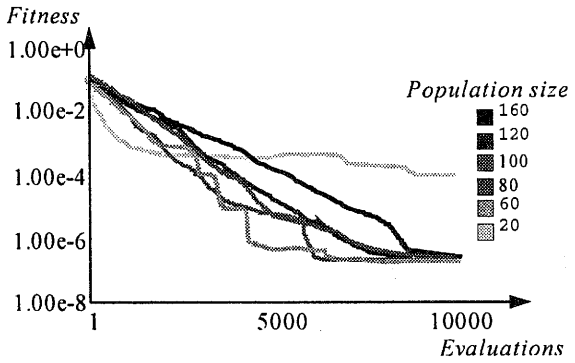


Fig. 13. Simulation results of Case 1 (SSGA).

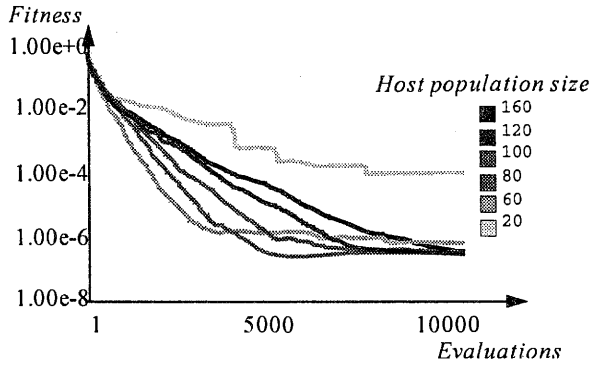


Fig. 14. Simulation results of Case 1 (VEGA).

Tab. 3. Average of fitness value after 10000 evaluations in Case 1.

Host pop. size	20	60	80	100	120	160
SSGA	1.00e-4	3.06e-7	3.42e-7	3.59e-7	4.04e-7	3.42e-7
VEGA	8.12e-5	4.24e-7	2.44e-7	2.12e-7	2.28e-7	3.23e-7

Tab. 4. Average of fitness value after 20000 evaluations in Case 2.

Host pop. size	20	60	80	100	120	160
SSGA	3.71e-5	3.74e-5	4.46e-5	3.89e-5	3.72e-5	4.99e-5
VEGA	4.38e-5	3.70e-5	3.41e-5	2.65e-5	3.65e-5	1.51e-4

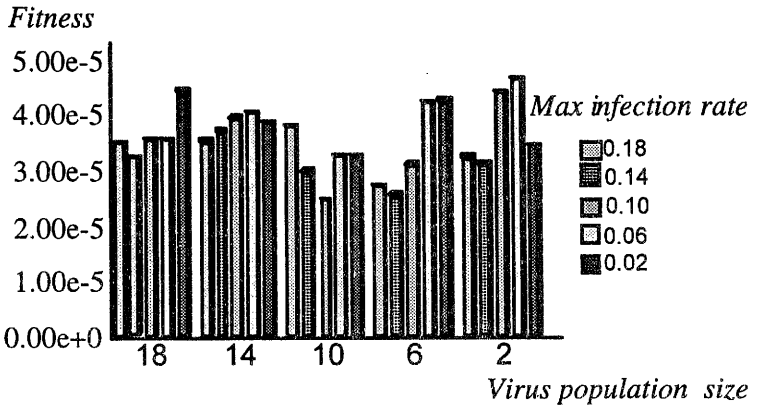


Fig. 15. Comparison of simulation results concerning the virus population size in Case 2.

The above simulation results of the knapsack and function optimization problems indicate that the VEGA attains the best solution when the host population size and the maximal infection rate are about 10% of host population size and about 0.1, respectively. The reason would be as follows. When the maximal infection rate is high, a virus individual infects almost all host individuals. As a result, a genetic diversity lacks in the host population. In contrast, when the maximal infection rate is very low, a virus individual cannot quickly propagate effective schemata in the host population.

Furthermore, the virus individuals can effectively propagate their substring, but the virus individuals are easy to change by the transduction and evolve by the transduction from the host individuals. The coevolution of the host and virus populations makes it possible to solve the optimization problem with good solution quickly.

4. Summary

This paper discusses the role of virus infection in a virus-evolutionary genetic algorithm (VEGA). The VEGA has two main features. The first one is horizontal propagation of genetic information, the other is vertical inheritance of genetic information. The horizontal propagation is performed by virus infection operators. The essential of the VEGA is a symbolic operation with reverse transcription and transduction as the virus infection operators. The reverse transcription plays the role of a crossover and selection simultaneously. The VEGA mainly searches the solution space with reverse transcription operator by generating new candidate solutions with overwriting host individuals partially. The virus infection is similar to a proportional selection scheme, since the virus individual performs the reverse transcription according to the virus infection rate. However, the VEGA differs from the GA in the generation of new individuals. The reverse transcription generates directly a candidate solution with

overwriting its substring in host individuals, though a crossover generates new candidate solutions with randomly combining substrings in a standard GA. Therefore, the VEGA can generate new candidate solutions with the directionality in the search.

The VEGA simulates coevolution of a virus population and a host population. A virus individual evolves by transducing from the infected host population. Therefore the best parameters concerning virus infection operators exist and the convergence of the host population depends on the virus infection rate. The VEGA can self-adaptively change the searching ratio between local and global searches according to the virus infection rate.

As future subjects, we intend to extend the schema theorem under virus infection and to carry out a detailed mathematical analysis of the virus infection mechanism.

References

Anderson N. (1970): *Evolutionary significant of virus infection.* — Nature, v.227, September, pp.1346–1347.

Baba N. and Kubota N. (1994): *Collision avoidance planning of a robot manipulator by using genetic algorithm - A consideration for the problem in which moving obstacles and/or several robots are in the workspace.* — Proc. 1st IEEE Conf. *Evolutionary Computation*, Orlando, Florida, v.2, pp.714–719.

Bramlette M. (1991): *Initialization, mutation and selection methods in genetic algorithms for function optimization.* — 4th Int. Conf. *Genetic Algorithms*, Virginia, pp.100–107.

Ecker J. and Kupferschmid M. (1988): *Introduction to Operations Research.* — New York: John Wiley & Sons Inc.

Fogel D.B. (1995a): *A comparison of evolutionary programming and genetic algorithms on selected constrained optimization problems.* — SIMULATION, v.64, No.6, pp.397–404.

Fogel D.B. (1995b): *Evolutionary Computation.* — New York: IEEE Press.

Goldberg D.E. (1989): *Genetic Algorithm in Search, Optimization, and Machine Learning.* — Reading, Massachusetts: Addison Wesley.

Holland J.H. (1992): *Adaptation in Natural and Artificial Systems.* — Cambridge, Massachusett: University of Michigan Press.

Kubota N., Fukuda T., Arai F. and Shimojima K. (1994): *Genetic algorithm with age structure and its application to self-organizing manufacturing system.* — Proc. EEE Symp. *Emerging Technologies and Factory Automation*, Tokyo, pp.472–477.

Kubota N., Shimojima K. and Fukuda T. (1996): *The role of virus infection in virus-evolutionary genetic algorithm.* — 3rd IEEE Int. Conf. *Evolutionary Computation*, Nagoya, pp.182–187.

Mühlenbein H., Schomisch M. and Born J. (1991): *The parallel genetic algorithm as function optimizer.* — Proc. 4th Int. Conf. *Genetic Algorithms*, Virginia, pp.271–278.

Primrose S.B. and Dimmock N.J. (1980): *Introduction to Modern Virology.* — Boston: Blackwell Scientific Publications.

Rechenberg I. (1994): *Evolution strategy.* — Computational Intelligence Imitating Life, New York: IEEE Press, pp.147–159.

Ridley M. (1993): *Evolution.* — Boston: Blackwell Scientific Publications.

Shimojima K., Fukuda T., Arai F. and Hasegawa Y. (1994): *Unsupervised/supervised learning for RBF-fuzzy inference – Adaptive rules and membership function and hierarchical structure by genetic algorithm.* — Proc. IEEE World Wisemen/Women Workshop, Nagoya, pp.97–104.

Syswerda G. (1991): *A study reproduction in generational and steady-state genetic algorithms*, In: Foundations of Genetic Algorithms. — San Mateo: Morgan Kaufmann, pp.94–101.

Tamaki H., Kita H., Shimizu N., Maekawa K. and Nishikawa (1994): *A comparison study of genetic codings for the Traveling Salesman problem.* — Proc. 1st IEEE Conf. *Evolutionary Computation*, Orlando, Florida, v.1, pp.1–6.

Whitley D. (1989): *The GENITOR algorithm and selection pressure: Why rank-based allocation of reproduction is best.* — 3rd Int. Conf. *Genetic Algorithms*, Arlington, pp.110–1115.