International Journal of Fuzzy Logic and Intelligent Systems, vol. 5, no. 1, March 2005 pp. 21-28

Derivative Evaluation and Conditional Random Selection for Accelerating Genetic Algorithms

Sung Hoon Jung*

*** Department of Information and Communication Engineering, Hansung Univ.**

Abstract

This paper proposes a new method for accelerating the search speed of genetic algorithms by taking derivative evaluation and conditional random selection into account in their evolution process. Derivative evaluation makes genetic algorithms focus on the individuals whose fitness is rapidly increased. This accelerates the search speed of genetic algorithms by enhancing exploitation like steepest descent methods but also increases the possibility of a premature convergence that means most individuals after a few generations approach to local optima. On the other hand, derivative evaluation under a premature convergence helps genetic algorithms escape the local optima by enhancing exploration. If GAs fall into a premature convergence, random selection is used in order to help escaping local optimum, but its effects are not large. We experimented our method with one combinatorial problem and five complex function optimization problems. Experimental results showed that our method was superior to the simple genetic algorithm especially when the search space is large.

Key words : Genetic Algorithms, Derivative Evaluation, Premature convergence, Random Selection, Optimization

1. Introduction

Genetic Algorithms (GAs) as robust and systematic optimization paradigms have been applied to many scientific and engineering problems including even those that conventional methods could not solve [1-9]. The performances of GAs are dependent on encoding schemes, recombination and evaluation operations, parameters of operations, and the algorithm itself [1,3,6,7]. A lot of researches for improving performances of GAs by adopting new methods have been introduced so far [9-18]. They are classified with two main streams such as adapting operator probabilities of GAs [9,10,12-16] and modifying crossover and mutation operators themselves [11,13,17,18].

This paper proposes a new method for improving the performances of GAs by accelerating the search speed of GAs. Unlike previous methods our method brings the evaluation operation of GAs into focus. In most GAs, individuals (in other words, chromosomes) are evaluated by a fitness function that represents how much each individual fits into the goal. Such evaluation, however, often makes GAs fall into a premature convergence [3,5,6,8,17]. The premature convergence occurs when a few comparatively highly fit (but not optimal) individuals are rapidly come to dominate the population, causing it to converge on local optima [5]. When a GA falls into a premature convergence, it is very difficult for the GA to escape this because further optimization by mutation is quite slow. This premature convergence degrades the performances of GAs [17].

In this paper, we propose a derivative evaluation that assigns fitness of individuals calculating how much offsprings are improved from their parents. This fitness is called derivative fitness. In case that offsprings are worse than their parents, their derivative fitness becomes zero. Therefore, even if an individual has very large fitness, if the individual is worse than its parents, its derivative fitness becomes zero. This makes GAs focus only on individuals whose fitness is rapidly increased. This derivative evaluation has similar features to the steepest descent methods. That is, that makes GAs approach to global optima quickly by enhancing exploitation but may also increase the possibility of a premature convergence. On the other hand, the derivative evaluation after reaching local optima helps GAs get out of the convergence by enhancing exploration. If all individuals have zero fitness by the derivative evaluation, the parents for next generation are selected by random. We call this conditional random selection. This conditional random selection reinforces exploration of GAs and results in making GAs escape the local optima and search new areas to find global optima. But, it was found from experiments that its effect is not large because the frequency of random selection is quite small.

Our method is experimented with one combinatorial problem and five complex function optimization problems. Experimental results showed that our method was superior to simple genetic algorithm especially when search space is large. Our method is simple but effective, so it can be easily incorporated into existing GAs for improving their performances.

This paper is organized as follows. Section 2 describes the derivative evaluation and conditional random selection method.

Manuscript received Feb. 17, 2005; revised Mar. 3, 2005.
This research was financially supported by Hansung University in the year of 2004.

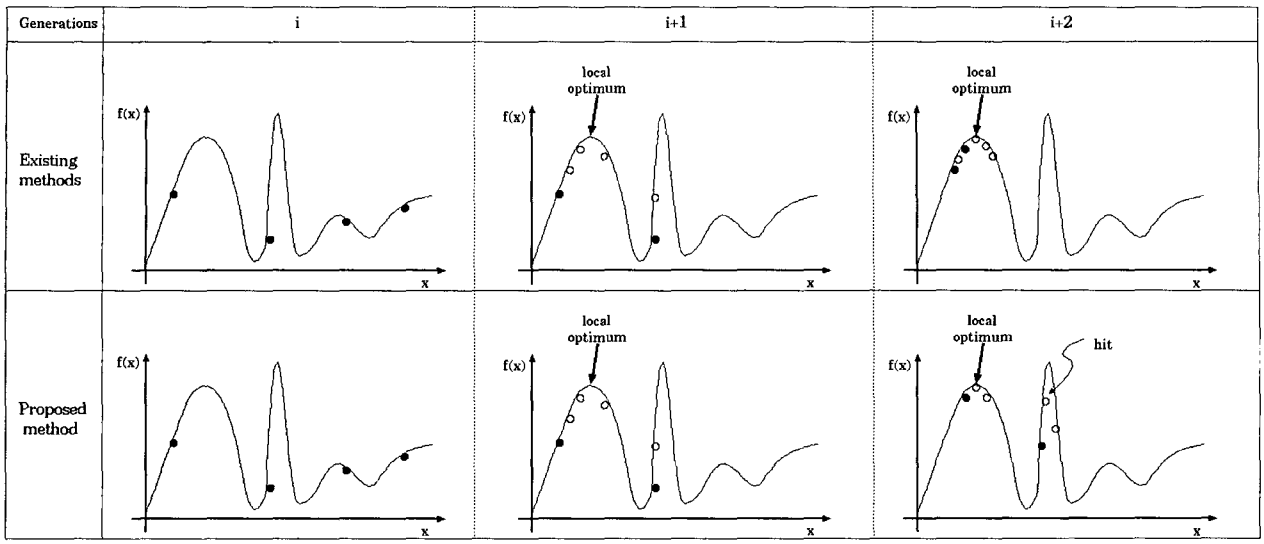

Figure 1: Comparison between existing methods and our method

In section 3, experimental environments and results are discussed. This paper concludes in section 4.

2. Derivative Evaluation and Conditional Random Selection

In most existing GAs, the evaluation operation assigns the fitness of individuals using a fitness function that represents how much individuals fit into the goal [1,9,10,13-17]. Since such evaluation calculates the fitness of an individual with only the individual, a few comparatively highly fit (but not optimal) individuals in an initial population are repeatedly selected as parents for next generation. Finally they are rapidly come to dominate the population and result in converge on local optima. This phenomenon is called premature convergence [1,3,5,6,8,17]. When a GA falls into a premature convergence, it is very difficult for the GA to escape this because further optimization by mutation is quite slow. This premature convergence that is one of the main problems in GAs research areas makes the performances of GAs degrade.

If the highly fit individuals, however, rapidly converges to a global optimum, the premature convergence is good for GAs, not a problem. That is, in order to improve the performances of GAs, the individuals of GAs should fast approach to optima because they may be global optima and if not, they should fast be distributed at which they converged. Let's consider Figure 1. In existing methods, individuals are evaluated by a fitness function $f(x)$. At generation i, four individuals depicted as dark circles in Figure 1 are evaluated as the position of y axis. At $i+1$ generation, if two individuals depicted as dark circles are selected as parents and the left one of them generates three white offsprings and right one of them generates one white offspring respectively, then the three white offsprings dominate the population because their fitness is greater than that of the right one offspring. As a result, GAs fall into a premature convergence as shown at the $i+2$ generation. In the other hands, the right one offspring at $i+1$ generation in proposed method have high probability to be selected as parents at next generation because the fitness of right offspring is greater than that in existing methods by adopting new fitness (see more detailed definition of new fitness given at Definition 2 below). As shown at the $i+2$ generation in proposed method, individuals that fall into a premature convergence have low probability to be alive because their new fitness is less than the right offsprings. From these properties, we can find that derivative evaluation using gradient informations enables GAs to fast approach to global optimum areas similar to hill climbing methods. Moreover, when most individuals fall into local optimum areas, derivative evaluation also helps GAs get out of premature convergence because prematurely convergent individuals will not be nearly selected as parents due to their small fitness. In most GAs, of course, two individuals are selected as parents and generate two offsprings. But, we explained under assumption that one individual generates one offspring for simplicity at above description.

Based on above observation, we propose a derivative evaluation and conditional random selection for enhancing the performances of GAs. In order to describe derivative evaluation, we define parents fitness and derivative fitness as follows. In this definition, we call the fitness obtained by a fitness function *original fitness*.

Definition 1: Parents fitness and derivative fitness

Let original fitness of ith individual at t generation be $f_i(t)$ and original fitness of parents of ith individual at $t-1$ generation be $f_j(t-1)$ and $f_k(t-1)$, respectively. Then, the parents fitness $f_i^p(t)$ and derivative fitness $f_i^D(t)$ of ith

individual are defined as:

$$\begin{aligned} f_i^p(t) &= \frac{f_j(t-1)+f_k(t-1)}{2} \\ &= avg(f_j(t-1), f_k(t-1)), \\ f_i^D(t) &= (f_i(t)-f_i^p(t)), \end{aligned} \quad (1)$$

where $1 \le i, j, k \le N$ and N is the number of individuals. At first generation ($t=1$), individuals are randomly created and all parents fitness $f_j(0)$, $f_k(0)$, *where* $1 \le j, k \le N$ are regarded as zero. As shown in equation 1, we take an average value of original fitness of parents individuals as typical fitness of parents. We experimented with the other operators for parents fitness such as *min* and *max*. But the average operator shows the best performance. Derivative fitness can be viewed as a measure of how an individual is improved from its parents. This fitness can also be regarded as a gradient of fitness. As gradient descendant methods, GAs focus on individuals with large derivative fitness as long as possible. The larger derivative fitness of an individual is, the more the individual must be selected as parents for next generation. From this viewpoint, we define new fitness with derivative fitness as follows.

Definition 2: New fitness

Let the derivative fitness of ith individual be defined as definition 1, then the new fitness $f_i^*(t)$ of the ith individual is defined as:

$$f_i^*(t) = \begin{cases} f_i^D(t) & \text{if } f_i^D(t) > 0 \\ 0 & otherwise. \end{cases} \quad (2)$$

We use this new fitness as fitness of individuals. If an individual is improved from its parents, then fitness of the individual is positive real value. Otherwise, fitness of the individual is zero. This evaluation called derivative evaluation has two advantages compared to original evaluation. First, this evaluation can make GAs avoid the premature convergence phenomenon because an individual whose original fitness is very large will not be selected as parents for next generation as long as it is not improved from its parents. That is, derivative evaluation makes GAs focus on evolved individuals not the individuals whose original fitness is large. Second, as a gradient method this evaluation accelerates search speed.

At initial generations, since most individuals are improved from their parents, most individuals has positive fitness not zero. However, after most individuals approaches local or global optimum areas, improvements (positive evolution) of individuals are difficult. At this situation, all individuals sometimes have zero fitness. Also, selection of parents in proportional to the fitness is not possible. Under this condition, we used random selection in order to help GAs get out of local optimum. But, it was found from experiments that its effect is not large because the condition rarely occurs. In fact, we experimented another selection (named constant selection) as well as random selection. But, none of them dominates the others.

3. Experimental Results and Discussion

Our proposed method was tested on one combinatorial problem and five typical function optimization problems. The six problems are described as follows.

$$f_1 = \sum_{j=1}^{h} m_j \begin{cases} m_j = 1 & \text{if } T_j = I_j \\ m_j = 0 & \text{if } T_j \neq I_j \end{cases}$$

$$f_2 = 100(x_1^2 - x_2)^2 + (1-x_1)^2, \quad where -2.048 \le x_i \le 2.048$$

$$f_3 = 0.002 + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a[i][j])^6}, \quad where -65.536 \le x_i \le 65.536$$

$$f_4 = 0.5 - \frac{\sin(\sqrt{x_1^2+x_2^2})\sin(\sqrt{x_1^2+x_2^2}) - 0.5}{(1.0+0.001(x_1^2+x_2^2))(1.0+0.001(x_1^2+x_2^2))}, \quad where -10 \le x_i \le -10$$

$$f_5 = (x_1^2+x_2^2)^{0.25}\sin(50(x_1^2+x_2^2)^{0.1}+1)^2, \quad where -10 \le x_i \le -10$$

$$f_6 = 1 + \sum_{i=1}^{2} \frac{x_i^2}{4000} - \prod_{i=1}^{2}\left(\cos\left(\frac{x_i}{\sqrt{i}}\right)\right), \quad where -5.12 \le x_i \le 5.12$$

The f_1 problem is a type of pattern matching problems. That is, the problem is to find target bit pattern T. Each individual is evaluated by the number of matching bits between the individual I and target T. Therefore, optimum fitness is same as the number of bits h. Five function optimization problems f_2 to f_6 used in many other papers called DeJong function 2 (f_2), DeJong function 5 (f_3), Mexican hat function (f_4), Shafer function 2 (f_5), Rastrigin function 2 (f_6), respectively. Figure 2 shows the input-output relations of six functions. The performances of proposed method have been measured and compared to those of original one. Except the derivative evaluation and conditional random selection, all other parameters--initial individuals, the probabilities of crossover and mutation, the population size, and the length of bit strings--are same in experiments. We set the parameters for experiments as shown in Table 1. Since GAs generally show somewhat different results according to the initial individuals, we measured experimental results with average values of 10 runs using different random number seeds. When GAs find the optimal solution, the number of generations at that generation is recorded and the results of 10 runs are averaged. This averaged value is an experimental result for one experiment. Table 2 shows total experimental results of six problems.

Table1: Parameter setting for experiments

Parameters	values
crossover probability (p_c)	0.6
mutation probability (p_m)	0.05
population size	10
The length of bit strings	16, 22, 28 bits

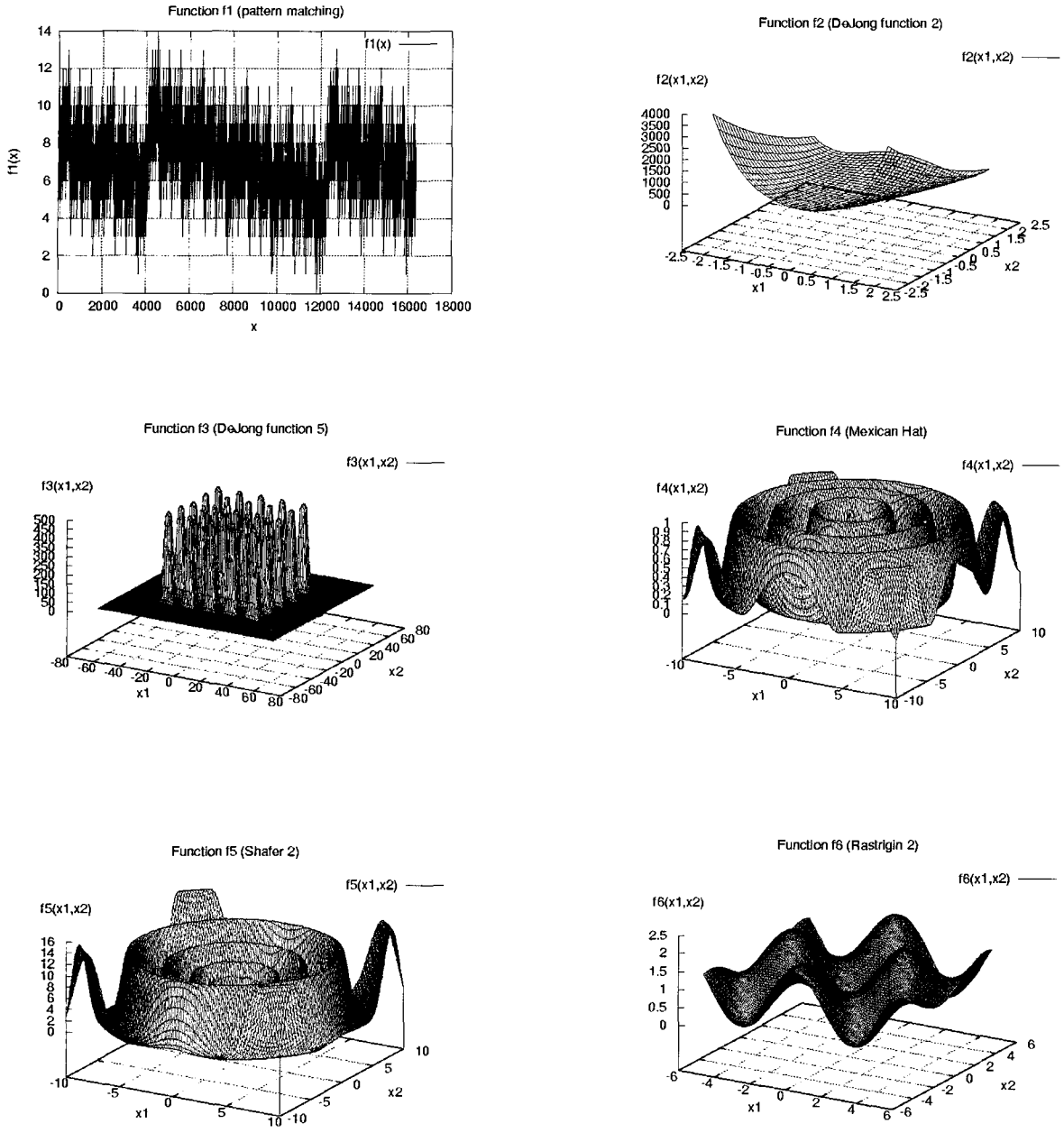

Figure 2: Experimental functions

In Table 2, NGA and SGA mean new genetic algorithm and simple genetic algorithm devised by Goldberg [1] and SGA/NGA at the seventh column is the ratio of avg. values of two methods. As shown in Table 2, NGA is superior to SGA at nearly all experiments except only one case (function: f_5 and bit strings: 16). Clearly, the optimum value of f_1 is only one and its value is the same as the length of bit strings. The optimum value of f_2 is only one at $x_1 = -2.048$ and $x_2 = -2.048$ and optimum value is about 3905.9. Similarly, function f_3 has only one optimum value (about 498.9) at $x_1 = -32.509984$ and $x_2 = -32.509984$. On the other hands, function f_4, f_5, and f_6 has multiple optima. Multiple optima of function f_4 are at the smallest circle of Mexican hat and its value about 0.99. Function f_5 has multiple optima at four peaks near ($x_1 = -10$ and $x_2 = -10$, $x_1 = -10$ and $x_2 = 10$, $x_1 = 10$ and $x_2 = -10$, $x_1 = 10$ and $x_2 = 10$) and its value is about 14.3. Function f_6 has four optima at four

Table 2: Experimental results

function	bit strings	NGA		SGA		SGA/NGA
		avg.	dev.	avg.	dev.	
f_1	16	19.6	13.81	536.7	730.76	27.4
	22	27.0	10.27	23810.2	12153.96	881.9
	28	57.1	46.11	589162.7	532250.75	10318.1
f_2	16	1023.3	1324.44	2891.9	3307.52	2.8
	22	944.9	889.46	11228.1	9526.42	11.9
	28	1332.9	1324.12	598051.8	473070.84	448.7
f_3	16	492.0	446.50	2821.7	2557.65	5.7
	22	1574.4	1698.54	16541.3	12669.00	10.5
	28	14410.5	14757.53	1140458.0	1051099.34	79.1
f_4	16	2464.6	1282.42	13457.1	8744.59	5.5
	22	11265.4	11031.59	357673.0	374413.34	31.7
	28	94974.1	56076.43	19940078.9	12961757.47	209.9
f_5	16	2182.2	1444.43	1576.6	1299.92	0.7
	22	26620.3	23979.09	85613.2	39818.25	3.2
	28	888116.5	678674.55	3918794.9	3520677.02	4.4
f_6	16	1475.9	1716.09	6694.2	6588.69	4.5
	22	2572.2	2441.18	138874.2	102389.13	53.9
	28	20772.5	20354.83	21670535.4	10932710.63	1043.2

Table 3: Comparisons between random selection and constant selection

function	bit strings	random selection		constant selection	
		avg.	dev.	avg.	dev.
f_1	16	19.6	13.81	19.0	7.20
	22	27.0	10.27	68.7	38.13
	28	57.1	46.11	159.6	125.80
f_2	16	1023.3	1324.44	67.4	53.51
	22	944.9	889.46	111.3	73.96
	28	1332.9	1324.12	619.8	622.79
f_3	16	492.0	446.50	637.3	344.14
	22	1574.4	1698.54	4396.0	5888.24
	28	14410.5	14757.53	37816.4	43015.27
f_4	16	2464.6	1282.42	3542.8	3231.97
	22	11265.4	11031.59	13247.7	8905.23
	28	94974.1	56076.43	163548.9	150119.81
f_5	16	2182.2	1444.43	3289.6	2982.51
	22	26620.3	23979.09	29898.4	23837.99
	28	888116.5	678674.55	1525107.8	1731978.42
f_6	16	1475.9	1716.09	441.4	475.30
	22	2572.2	2441.18	3013.7	3437.98
	28	20772.5	20354.83	24565.6	20391.95

peaks near ($x_1=-0.04$ and $x_2=-4.47$, $x_1=-0.04$ and $x_2=4.47$, $x_1=0.04$ and $x_2=-4.47$, $x_1=0.04$ and $x_2=4.47$) and its value is about 2.003. NGA shows very good performances when a function is simple, in other words, relatively small local optima such as f_1, f_2, and f_6. Even when the local optima is large as the cases of f_3 and f_4, NGA shows relatively good performances. In the function f_5, however, the performance of NGA is not good relatively than the others. This may be because the four peaks of global optima is far from the local optima in comparison that the

Table 4: Experimental results at 30 population size

function	bit strings	NGA		SGA		SGA/NGA
		avg.	dev.	avg.	dev.	
f_1	16	8.7	4.90	88.0	67.24	10.1
	22	11.0	3.19	2391.9	2207.02	217.4
	28	16.4	9.06	46576.2	43825.83	2840.0
f_2	16	137.1	231.20	950.8	1201.40	6.9
	22	197.1	149.37	6410.0	7021.05	32.5
	28	146.6	170.99	72868.3	76501.18	497.0
f_3	16	86.3	70.83	5840.7	5706.90	67.6
	22	715.9	690.48	4411.8	5155.09	6.1
	28	1548.6	1185.17	136731.8	102911.63	88.2
f_4	16	1530.3	1733.36	6932.6	5892.90	4.5
	22	2644.3	2525.05	116707.3	87334.37	44.1
	28	2005.1	1780.01	5490873.5	5368471.48	2738.4
f_5	16	979.2	927.21	2253.8	2009.12	2.3
	22	8751.9	8427.41	13964.1	14163.65	1.5
	28	131702.8	120836.44	751085.6	919829.84	5.7
f_6	16	394.0	624.03	4095.4	2841.02	10.3
	22	550.3	482.35	55268.4	42305.13	100.4
	28	874.5	611.58	2458014.7	1746689.23	2810.7

other functions. From this observation, we can drive some parameters for measuring the degree of difficulty of optimization as follows. Optimization will be affected by following factors.

- The number of local peaks that include local optima (the fewer, the better).
- The distances between local peaks (the nearer, the better).
- The areas of local peaks (the narrower, the better).
- The difference between local optima (the larger, the better)
- The number of global peaks that include global optima (the fewer, the better).
- The distances between global peaks (the nearer, the better).
- The areas of global peaks (the broader, the better).
- The distances between local peaks and global optima (the nearer, the better).
- The difference between the values of local maximum and global maximum (the larger, the better)

From the viewpoints of above seventh parameters, the best simple function is the one which has broad global peak with no local peak. On the other hands, the function that has many global peaks whose areas are narrow and whose locates are broadly distributed and many local peaks whose areas are broad is the best difficult function to optimize. Figure 3 shows the best simple and relatively complicated functions to optimize. Therefore, function f_5 is the most difficult function. In this function, NGA shows relatively poor performances than the others. This indicates that the other factors for this kind of problems must be considered. Even at the function f_5, however, NGA shows better performances than SGA at the length of bit strings 22 and 28. This shows that the NGA is

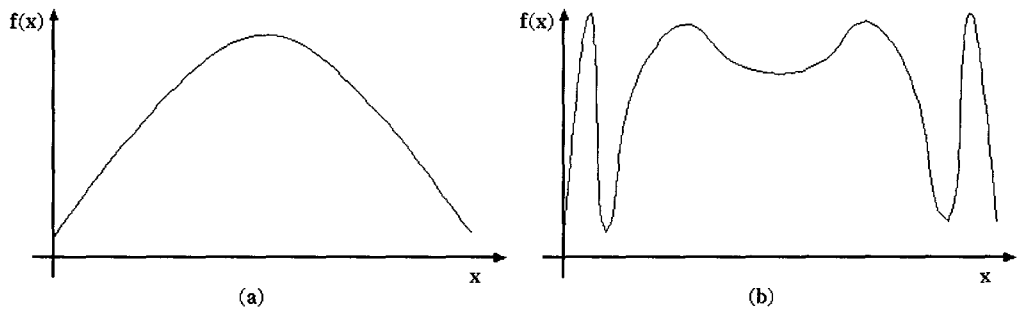

Figure 3: The best simple and complicated functions to optimize

more effective as the search space is larger. At all problems, NGA shows better performance as the search space is larger.

In order to show the effect of selection method when new fitness of all individuals is zero, a constant selection method is applied. In this selection method, a middle individual in population is unconditionally selected as parents. For example, when the population size is 10, 5th individual (10 / 2 = 5) is unconditionally selected when new fitness of all individuals is zero. This situation rarely occurs because the probability that all individuals have zero new fitness is very low. Table 3 shows comparisons between random selection and constant selection. As shown in table 3, random selection in some functions is better than constant selection, in other functions vice versa. Also, the difference between two methods is small in comparison to the SGA. We can not conclude which method dominates the others.

In order to show the effect of population size, we experimented our method with 30 population size. Table 4 shows the experimental result when population size is 30. In this experiment, we used constant selection for NGA. It is intuitively natural that as the population size is larger and larger, the number of generation that find optimum solutions is more and more decreased. As shown in table 4, all average values are less than those of table 2. In most results, the trend of results is quite similar to the table 2. The ratio between SGA and NGA is better than the table 2 in most cases. This indicates that our method is more effective as the population size is large. Very similar to the table 2, the SGA/NGA ratio at function f_5 is very low in comparison to the other results. This means that more effective method for this type of functions must be revised. We will do the work as a further work later.

4. Conclusion

In this paper, we introduced a derivative evaluation and conditional random selection method for accelerating the optimization capability of genetic algorithms. Proposed method have less possibility of a premature convergence than existing GAs because it enhances the exploitation and exploration by focusing on rapidly evolved individuals and randomly selected parents under certain condition. We tested proposed method with six typical problems--one combinatorial problem and five function optimization problems. Experimental results showed that our method at most problems was superior to the SGA especially when the search space is large. In the problems having a high degree of difficulty, NGA shows relatively poor performances. A new accelerating method for such problems is remained as further works.

References

[1] D. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison--Wesley, 1989.

[2] C. L. Karr and E. J. Gentry, ``Fuzzy Control of pH Using Genetic Algorithms,"*IEEE Trans. on Fuzzy Systems*, vol. 1, pp. 46--53, Jan. 1993.

[3] M. Srinivas and L. M. Patnaik, ``Genetic Algorithms: A Survey,"*IEEE Computer Magazine*, pp. 17--26, June 1994.

[4] J. L. R. Filho and P. C. Treleaven, `` Genetic-Algorithm Programming Environments," *IEEE Computer Magazine*, pp. 28--43, June 1994.

[5] D. Beasley, D. R. Bull, and R. R. Martin, ``An Overview of Genetic Algorithms: Part 1, Fundamentals ," Technical Report.

[6] D. B. Fogel, ``An Introduction to Simulated Evolutionary Optimization," *IEEE Trans. on Neural Networks*, vol. 5, pp. 3--14, Jan. 1994.

[7] H. Szczerbicka and M. Becker, ``Genetic Algorithms: A Tool for Modelling, Simulation, and Optimization of Complex Systems," *Cybernetics and Systems: An International Journal*, vol. 29, pp. 639--659, Aug. 1998.

[8] R. Yang and I. Douglas, ``Simple Genetic Algorithm with Local Tuning: Efficient Global Optimizing Technique," *Journal of Optimization Theory and Applications*, vol. 98, pp. 449--465, Aug. 1998.

[9] C. Xudong, Q. Jingen, N. Guangzheng, Y. Shiyou, and Z. Mingliu, `` An Improved Genetic Algorithm for Global Optimization of Electromagnetic Problems ," *IEEE Trans. on Magnetics* , vol. 37, pp. 3579--3583, Sept. 2001.

[10] L. Davis, ``Adapting Operator Probabilities in Genetic Algorithms ," in *Proceedings of the 3rd International Conference on Genetic Algorithms and their Applications*, pp. 61--69, 1989.

[11] R. Hinterding, ``Gaussian Mutation and Self-adaption in Numeric Genetic Algorithms ," in *Proceedings of the 2nd IEEE International Conference on Evolutionary Computation*, pp. 384--389, 1995.

[12] R. Hinterding, Z. Michalewicz, and A. E. Eiben, ``Adaptation in Evolutionary Computation: A Survey ," in *Proceedings of the 4th IEEE International Conference on Evolutionary Computation*, pp. 65--69, 1997.

[13] J. E. Smith and T. C. Fogarty, `` Operator and parameter adaptation in genetic algorithms ," *Soft computing : a fusion of foundations, methodologies and applications*, vol. 92, no. 2, pp. 81--87, 1997.

[14] M. C. Sinclair, `` Operator-probability Adaptation in a Genetic-algorithm/Heuristic Hybrid for Optical Network Wavelength Allocation," in *Proc. IEEE Intl. Conf. on Evolutionary Computation (ICEC'98)*, Anchorage, Alaska, USA , pp. 840--845, 1998.

[15] A. Tuson and P. Ross, ``Adapting Operator Settings In Genetic Algorithms ," *Evolutionary Computation*, vol. 6, no. 2, pp. 161--184, 1998.

[16] C. W. Ho, K. H. Lee, and K. S. Leung, ``A Genetic Algorithm Based on Mutation and Crossover with Adaptive Probabilities ," in *Proceedings of the 1999 Congress on Evolutionary Computation*, vol. 1, pp.

768--775, 1999.

[17] J. Andre, P. Siarry, and T. Dognon, ``An improvement of the standard genetic algorithm fighting premature convergence in continuous optimization ,'' *Advances in engineering software*, vol. 32, no. 1, pp. 49--60, 2001.

[18] S. H. Jung, ``Queen-bee evolution for genetic algorithms ,'' *Electronics Letters*, vol. 39, pp. 575--576, Mar. 2003.

Sung Hoon Jung

Sung Hoon Jung received his B.S.E.E. degree from Hanyang University, Korea, in 1988 and M.S. and Ph.D. degrees from KAIST, in 1991 and 1995, respectively. He joined the Department of Information and Communication Engineering at the Hansung University in 1996, where he is an associate professor. His research interests are in the field of intelligent systems, and in particular of application of neural networks, fuzzy logic, evolutionary computation algorithms (genetic algorithms, evolutionary programming and evolution strategy) to intelligent systems such as intelligent control systems, intelligent characters of computer games, and so on. He is a member of the Korea Fuzzy Logic and Intelligent Systems Society (KFIS) and Institute of Electronics Engineers of Korea (IEEK).

Phone : 02-760-4344
Fax : 02-760-4435
E-mail : shjung@hansung.ac.kr