한국지능정보시스템학회논문지 제8권 제2호 2002년 12월(pp.189~203)

Comparison of Adaptive Operators in Genetic Algorithms*

YoungSu Yun
School of Automotive, Industrial & Mechanical Engineering Daegu University
(joy629@hitel.net)

Seung-Lock Seo
School of Automotive, Industrial & Mechanical Engineering Daegu University
(seosl@taegu.ac.kr)

In this paper we compare the performances of adaptive operators in genetic algorithm. For the adaptive operators, the crossover and mutation operators of genetic algorithm are considered. One fuzzy logic controller is developed in this paper and two heuristics is presented from conventional works for constructing the operators. The fuzzy logic controller and two conventional heuristics adaptively regulate the rates of the operators during genetic search process. All the algorithms are tested and analyzed in numerical examples. Finally, the best algorithm is recommended.

Key words: Adaptive operators, genetic algorithm, crossover operator, mutation operator, and fuzzy logic controller.

1. Introduction

Despite the success of applying genetic algorithms (GAs) to solve numerous problems, the identification of the correct settings of genetic parameters (such as crossover and mutation operators) for the problems is not an easy task. This is mainly because the performance of GAs highly depends on the setting values of the parameters. Therefore, in order to identify the correct setting of the values, many conventional works have been performed (De Jong, 1975; Grefenstette, 1986; Eiden, Hinterding, and Michalewicz; 1999). Most of these works can be classified into two major forms: parameter tuning and parameter control.

In the parameter tuning approach, various efforts to find good values for the parameters before the run of GA should be performed and then the algorithm is executed using these values, which remains fixed during the run. These efforts are usually performed by trial-and-error. De Jong

* 본 연구는 대구대학교 학술연구지원사업에 의해서 수행되었음.

(1975) and Grefenstette (1986) recommended reasonable setting values for the parameters using the parameter tuning approach. However, several technical drawbacks on the approach are also reported in the literatures (Syswerda; 1991, Back: 1992, Eiden, Hinterding, and Michalewicz; 1999). The general drawback of the approach, regardless of how the parameters are tuned, is based on the observation that the run of GA is an intrinsically dynamic and adaptive process according to the problems applied. Therefore, the use of rigid parameters that does not change their setting values is in contrast to this spirit. Parameter control forms an alternative since it starts a run with initial parameter values which are changed during the run.

By using the parameter control approach, the parameters of GAs can be dynamically regulated according to the problems applied. Especially, the idea to adaptively regulate the crossover and mutation operators of GAs has been employed in several works (Hong and Wang; 1996, Hong, et. al.; 2002, Srinvas and Patnaik; 1994, Mak, Wong, and Wang; 2000, Wu, Cao, and Wen; 1998, Wang, Wang, and Hu; 1997). These works can be classified into two modes: one employs artificial intelligent techniques such as fuzzy logic controllers (FLCs) and the other uses heuristics. The genetic parameters controlled by these two modes are adaptively regulated during genetic search process. Therefore, much time for the fine-tuning of the parameters can be saved, and the search ability of GAs can be improved in finding global optimum.

For the first mode, Gen and Cheng (2000) surveyed various adaptive methods using several FLCs. Subbu, Sanderson, and Bonissone (1998) proposed a fuzzy logic controlled genetic algorithm (FLC-GA), and the FLC-GA can regulate the rates of crossover and mutation operators. Wang, Wang, and Hu (1997) used two FLCs: crossover FLC and mutation FLC. These parameters are considered as the input variables of GAs and are also taken as the output variables of the FLC. For successfully applying FLC to GA, the key is to produce well-formed fuzzy sets and rules. Recently, Cheong and Lai (2000) suggested an optimization scheme for the sets and rules. The GAs controlled by these fuzzy logics are more efficient in search speed and search quality than the GAs without them.

For the second mode, Mak, Wong, and Wang (2000) adaptively regulated the crossover and mutation rates according to the performances of GA operators in a manufacturing cell formulation problem. This heuristic scheme bases on the fact that it encourages well-performing operators to produce more efficient offspring, while reducing the chance for the poorly performing operators to destroy the potential individuals, during genetic search process. Srinivas and Patnaik (1994), Wu, Cao, and Wen (1998) also controlled the rates with respect to the fitness values of the population at each generation. These two works recommended the use of adaptive crossover and mutation rates to maintain diversity in the population of each generation and to sustain the convergence property of GA.

Although most of the works mentioned in the first and second modes have proved their effectiveness in each work, comparisons or analyses among them under same condition have not yet been provided in previous works. Therefore, the aim of this paper is to compare the performance among them using the GA with a same condition.

Adaptive genetic operators using one FLC and two heuristics are suggested in Section 2. In Section 3, three adaptive GAs using the concepts suggested in Section 2 are described, respectively. To compare the performance of each algorithm, three test functions are presented and analyzed in Section 4, and conclusion is followed in Section 5.

2. Adaptive genetic operators (AGOs) for parameter control

In this section, we suggest adaptive crossover and mutation operators using one FLC and two heuristics. The rates of the two operators are adaptively regulated according to the fitness values calculated by each method.

2.1 AGO using conventional heuristics

For the AGO using conventional heuristics, we use two concepts from the previous works (Mrk, Wong, and Wang, 2000; Srinivas and Patnaik; 1994). The inputs of these heuristics for crossover and mutation operators use one or several fitness values, and their outputs adaptively regulate the rates of the operators, at each generation of GAs. Their concepts and logics are as follows.

2.1.1 AGO using the fitness values of parent and offspring at each generation: heuristic 1

For the heuristic 1, we use the concept of Mrk, Wong, and Wang (2000). They employed the fitness values of parent and offspring at each generation in order to construct adaptive crossover and mutation operators: this scheme increases the occurrence rates of the crossover and mutation operators, when it consistently produces a better offspring during genetic search process; however, it also reduces the occurrence rates of the operators, when it produces a poorer offspring. This scheme bases the fact that it encourages the well-performing crossover and mutation operators to produce more offspring, while also reducing the chance for the poorly performing operators to destroy the potential individuals during genetic search process. It is the main scheme for constructing this AGO. The detailed procedure for a minimization problem is as follows:

Procedure: regulation of the rates of crossover and mutation operators

begin

if $(\overline{f_{par\_size}}(t)/\overline{f_{off\_size}}(t)) - 1 \geq 0.1$ **then**

$p_C(t+1) = p_C(t) + 0.05$, $p_M(t+1) = p_M(t) + 0.005$

if $(\overline{f_{par\_size}}(t)/\overline{f_{off\_size}}(t)) - 1 \leq 0.1$ **then**

$p_C(t+1) = p_C(t) - 0.05$, $p_M(t+1) = p_M(t) - 0.005$

if $-0.1 < (\overline{f_{par\_size}}(t) / \overline{f_{off\_size}}(t)) - 1 < 0.1$ **then**

$p_C(t+1) = p_C(t), \quad p_M(t+1) = p_M(t)$

end

end

where *par_size* and *off_size* are the parent size and offspring size satisfying constraints, respectively. $\overline{f_{par\_size}}(t)$ and $\overline{f_{off\_size}}(t)$ are the average fitness values of the parents and offspring at generation t, respectively. $p_C(t)$ and $p_M(t)$ are respectively the rates of crossover and mutation operators at generation t. In the cases of $(\overline{f_{par\_size}}(t) / \overline{f_{off\_size}}(t)) - 1 \geq 0.1$ and $(\overline{f_{par\_size}}(t) / \overline{f_{off\_size}}(t)) - 1 \leq 0.1$, the adjusted rates should not exceed the range [0.5, 1.0] for the $p_C(t+1)$ and the range [0.00, 0.10] for the $p_M(t+1)$.

The procedure defined above is evaluated in every generation during genetic search process, and the occurrence rates of crossover and mutation operators are adaptively regulated according to the result of the procedure.

2.1.2 AGO using various fitness values at each generation: heuristic 2

For the heuristic 2, Srinivas and Patnaik (1994) developed an adaptive scheme that considers both the exploitation and exploration properties in the convergence process of GA: the capacity to converge to a global optimum after locating the region containing the optimum, and the capacity to explore the new regions of solution space in the search of the optimum. The balance between these characteristics of GA is adaptively regulated by the values of P_c and P_M at each generation: increasing the values of the P_c and P_M promotes exploration at the expense of exploitation. By this basic scheme, the P_c and P_M are increased when the population tends to get stuck at a local optimum and are decreased when the population is scattered in the search space of GA. The detailed scheme for a maximization problem is as follows:

$$p_C = \begin{cases} \alpha_1 (f_{\max} - f_{cro}) / (f_{\max} - f_{avg}), & f_{cro} > f_{avg} \\ \alpha_3, & f_{cro} \leq f_{avg} \end{cases}$$

and

$$p_M = \begin{cases} \alpha_2 (f_{\max} - f_{mut}) / (f_{\max} - f_{avg}), & f_{mut} > f_{avg} \\ \alpha_4, & f_{mut} \leq f_{avg} \end{cases}$$

where f_{max} and f_{avg} are the maximum fitness and average fitness values at each generation, respectively. f_{cro} is the larger of the fitness values of the individuals to be crossed. f_{mut} is the fitness value of the ith individual to which the mutation with a rate P_M is applied. The values of α_1, α_2, α_3, and α_4, are 1, 0.5, 1, and 0.5, respectively (Srinivas and Patnaik; 1994).

The adjusted rate should not exceed the range [0.5, 1.0] for the P_c and the range [0.001, 0.05] for the P_M.

2.2 AGO using FLC

To adaptively regulate GA operators using

FLC, we use the concept of Wang, Wang and Hu (1997) and improve it. Its main scheme is to use two FLCs: the crossover FLC and mutation FLC are implemented independently to adaptively regulate the rates of crossover and mutation operators during genetic search process.

The heuristic updating strategy for the rates is to consider the change of the average fitness of population in each generation. For example, in a minimization problem, we can set the change of the average fitness at generation t-1, $\Delta f_{\text{avg}}(t\text{-}1)$ and the change at generation t, $\Delta f_{\text{avg}}(t)$ as follows:

$$\Delta f_{avg}(t-1) = (\overline{f_{par\_size}}(t-1) - \overline{f_{off\_size}}(t-1)) \times \lambda$$

$$= \left(\frac{\sum_{k=1}^{par\_size} f_k(t-1)}{par\_size} - \frac{\sum_{k=par\_size+1}^{par\_size+off\_size} f_k(t-1)}{off\_size}\right) \times \lambda$$

$$\Delta f_{avg}(t) = (\overline{f_{par\_size}}(t) - \overline{f_{off\_size}}(t)) \times \lambda$$

$$= \left(\frac{\sum_{k=1}^{par\_size} f_k(t)}{par\_size} - \frac{\sum_{k=par\_size+1}^{par\_size+off\_size} f_k(t)}{off\_size}\right) \times \lambda$$

where λ is a scaling factor to normalize the average fitness for applying defuzzification in two FLCs and is varied according to the problem under consideration.

The coefficient was not used in the original work (Wang, Wang and Hu; 1997). However, the λ should be definitely required for normalizing the average fitness because the fitness is varied according to the problem under consideration. The $\Delta f_{\text{avg}}(t\text{-}1)$ and $\Delta f_{\text{avg}}(t)$ are used to regulate P_c and P_M as follows:

Procedure: regulation of P_c and P_M using the average fitness

begin

if $\varepsilon \le \Delta f_{avg}(t-1) \le \gamma$ **and**

$\varepsilon \le \Delta f_{avg}(t) \le \gamma$ **then**

increase p_C and p_M for next generation ;

if $-\gamma \le \Delta f_{avg}(t-1) \le -\varepsilon$ **and**

$-\gamma \le \Delta f_{avg}(t-1) \le -\varepsilon$ **then**

decrease p_C and p_M for next generation ;

if $-\varepsilon < \Delta f_{avg}(t-1) < \varepsilon$ **and**

$-\varepsilon < \Delta f_{avg}(t-1) < \varepsilon$ **then**

rapidly increase p_C and p_M for next generation ;

end

end

where ε is a given real number in the proximity of zero, γ and $-\gamma$ are respectively a given maximum and minimum values of a fuzzy membership function. The implementation strategy for the crossover FLC is as follows:

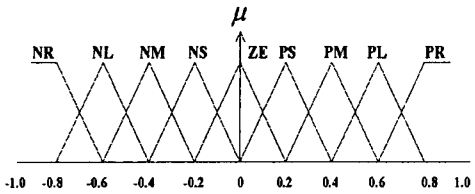
- ***Input and output of crossover FLC***

  The inputs of the crossover FLC are the $\Delta f_{avg}(t\text{-}1)$ and $\Delta f_{avg}(t)$. Its output is a change in the crossover rate $\Delta c(t)$.

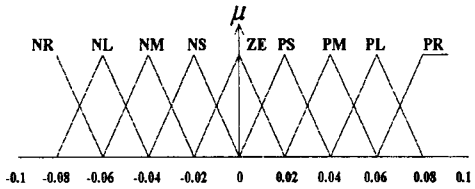
- ***Membership functions of,*** $\Delta f_{avg}(t\text{-}1)$, $\Delta f_{avg}(t)$, and $\Delta c(t)$

  The membership functions of the fuzzy input and output linguistic variables are illustrated in Figures 1 and 2, respectively. The $\Delta f_{avg}(t\text{-}1)$ and $\Delta f_{avg}(t)$ are respectively normalized into the range [-1.0, 1.0]. The $\Delta c(t)$ is normalized into

the range [-0.1, 0.1] according to their corresponding maximum values.



<Figure 1> Membership functions for Δfavg(t-1) and Δfavg(t)



<Figure 2> Membership function of Δc(t)

Where: NR - Negative larger, NL - Negative large, NM - Negative medium, NS - Negative small, ZE - Zero, PS - Positive small, PM - Positive medium, PL - Positive large, PR - Positive larger.

• ***Fuzzy decision table***

Based on a number of experiments and domain expert opinions, the fuzzy decision table was drawn as shown in Table 1 (Wang, Wang and Hu; 1997).

• ***Defuzzification table for control actions***

For simplicity, the defuzzification table for determining the action of the crossover FLC was setup. It is formulated as shown in Table 2.

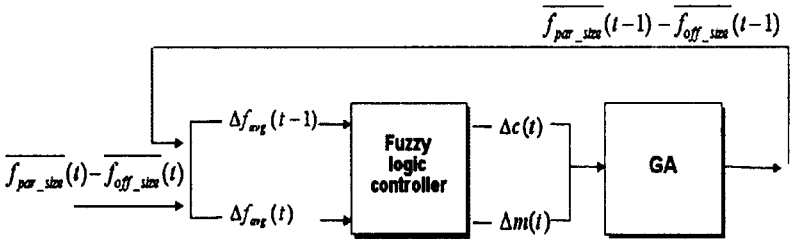
Table 1. Fuzzy decision table for crossover

$\Delta c(t)$		$\Delta f_{avg}(t-1)$								
		NR	NL	NM	NS	ZE	PS	PM	PL	PR
$\Delta f_{avg}(t)$	NR	NR	NL	NL	NM	NM	NS	NS	ZE	NE
	NL	NL	NL	NM	NM	NS	NS	ZE	ZE	PS
	NM	NL	NM	NM	NS	NS	ZE	ZE	PS	PS
	NS	NM	NM	NS	NS	ZE	ZE	PS	PS	PM
	ZE	NM	NS	NS	ZE	PE	PS	PS	PM	PM
	PS	NS	NS	ZE	ZE	PS	PS	PM	PM	PL
	PM	NS	ZE	ZE	PS	PS	PM	PM	PL	PL
	PL	ZE	ZE	PS	PS	PM	PM	PL	PL	PR
	PR	ZE	PS	PS	PM	PM	PL	PL	PR	PR

Table 2. Defuzzification table for control action of crossover

$Z(i,j)$		i								
		-4	-3	-2	-1	0	1	2	3	4
j	-4	-4	-3	-3	-2	-2	-1	-1	0	0
	-3	-3	-3	-2	-2	-1	-1	0	0	1
	-2	-3	-2	-2	-1	-1	0	0	1	1
	-1	-2	-2	-1	-1	0	0	1	1	2
	0	-2	-1	-1	0	2	1	1	2	2
	1	-1	-1	0	0	1	1	2	2	3
	2	-1	0	0	1	1	2	2	3	3
	3	0	0	1	1	2	2	3	3	4
	4	0	1	1	2	2	3	3	4	4

The inputs of the mutation FLC are the same as those of the crossover FLC, and its output is the change in the mutation rate, $\Delta m(t)$. The combination strategy between the FLC and GA is shown in <Figure 3>.



<Figure 3> Coordinated strategy of the FLC and GA

The detailed procedure for the application of the FLC is as follows:

Step 1. The input variables of the FLC for regulating the GA operators are the changes of the average fitness in two continuous generations (t-1 and t) as follows:

$$\Delta f_{avg}(t\text{-}1),\ \Delta f_{avg}(t)$$

Step 2. $\Delta f_{avg}(t\text{-}1)$ and $\Delta f_{avg}(t)$ are assigned to the indexes i and j corresponding to the control actions in the defuzzification table, after normalizing them (see Table 2).

Step 3. Calculate the changes of the crossover rate $\Delta c(t)$ and the mutation rate $\Delta m(t)$ as follows:

$\Delta c(t) = Z(i, j)\times 0.02 \quad \Delta m(t) = Z(i, j)\times 0.002$

where the contents of $Z(i, j)$ are the corresponding values of $\Delta f_{avg}(t\text{-}1)$ and $\Delta f_{avg}(t)$ for the defuzzification (see Table 2). The values 0.02 and 0.002 are given values to regulate the increasing and decreasing ranges of the rates of crossover and mutation operators

Step 4. Update the changes of the rates of crossover and mutation operators by using the following equations:

$$p_c(t) = p_c(t\text{-}1) + \Delta c(t),$$
$$p_M(t) = p_M(t\text{-}1) + \Delta m(t)$$

The adjusted rates should not exceed the range [0.5, 1.0] for the $p_c(t)$ and the range [0.0, 0.1] for the $p_M(t)$.

3. Adaptive Genetic Algorithms (AGAs)

In this section, we suggest two AGAs with the logics and procedures defined in section 2.1 and propose one AGA with the procedure in section 2.2. For experimental comparison, we also present a canonical genetic algorithm without any adaptive scheme. First, the canonical genetic algorithm (CGA) is suggested and then the three AGAs are followed.

3.1 Canonical Genetic algorithm (CGA)

For the CGA, we use a real-number representation instead of a bit-string one, and the detailed heuristic procedure for the CGA is as follows:

Step 1: Initial population

We use the population obtained by random number generation

Step 2: Genetic operators

Selection: elitist strategy in enlarged sampling space (Gen and Cheng, 2000)

Crossover: non-uniform arithmetic crossover operator (Michalewicz, 12994)

Mutation: uniform mutation operator (Michalewicz, 12994)

Step 3: Stop condition

If a fixed maximum generation number is reached or an optimal value is located during search process, then stop; otherwise, go to Step 2.

This procedure is also summarized in <Figure 4> (a).

3.2 Adaptive Genetic Algorithms

The first adaptive genetic algorithm (AGA1) combines the CGA and the procedure of the heuristic 1 suggested in sub-section 2.1.1. The combined heuristic procedure for the AGA1 is as follows:

Step 1: apply the first step of the CGA (i.e. initial population)

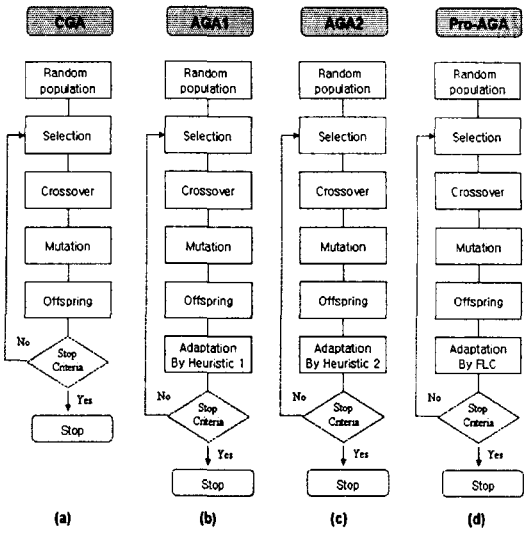
Step 2: apply the second step of the CGA (i.e. selection, crossover, and mutation)

Step 3: apply the heuristic 1 for adaptively regulating GA parameters (i.e., the rates of crossover and the mutation operators) in current generation.

Step 4: repeat Steps 2 and 3 until the stop condition of the CGA is reached.

For the second and last AGAs (AGA2 and pro-AGA), we consider i) the scheme of the heuristic 2 suggested in sub-section 2.1.2 and ii) the improved one of the FLC in Section 2.2. These two schemes are respectively used for the Step 3 among the steps defined in this section 3.2.

All the procedures defined above are summarized in <Figure 4>.



<Figure 4>. Various genetic algorithms

4. Numerical Examples

In this section, one CGA and three AGAs proposed in Section 3 are tested in order to compare their performances. These algorithms are applied to three test functions. The obtained results are compared with each other and are analyzed using various measures of the performances.

For the performances, three measures are used in the test functions: i) number to getting stuck (NGS), ii) average number of generations (ANG), and iii) time to stop condition (TSC). The NGS means total number that gets stuck at a local optimum in each algorithm, the ANG is obtained by the average number of generations that reaches to a given stop condition, and the TSC means the average CPU time when each algorithm reaches to a given stop condition. The adaptive abilities

among the three AGAs are also tested by using the behaviors of crossover operator, mutation operator, and average fitness, during the search process of each algorithm.

For experimental comparison under same condition, the parameters of each algorithm are set at population size: 20, crossover rate: 0.5, mutation rate: 0.05, generation number: 2,000. The "crossover rate" and "mutation rate" for the CGA procedure are fixed at same values during its search process. However, the rates for the three AGAs are adaptively regulated during their search process. Altogether 20 iterations were executed to eliminate the randomness of the searches. The procedures of all the algorithms were implemented in Visual Basic language under IBM-PC Pentium-1.6Ghz computer with 256RAM.

4.1 Test Problems

For the test problems, three test functions are considered as follows:

Test function 1(T-1): this is a multi-model function of two continuous variables and reaches its global maximum of 1.0 at the point (χ_1=χ_2=0.0), and the search ranges of χ_1, χ_2 have the same range -100.0 to +100.0. (Davis, 1991). The expression is

$$f(x_1,x_2)=0.5-\frac{(\sin\sqrt{x_1^2+x_2^2})^2-0.5}{1.0+0.001(x_1^2+x_2^2)^2}$$

Test function 2(T-2): this is to minimize a function of two variables called the Rosenbrock function. This function is a De Jong's F2 (De Jong, 1975) and it has a global minimum of zero at χ_1= χ_2=1.0, and each of χ_1 and χ_2 have the continuous values within the range -2.048 to +2.048.

$$f(x_1,x_2)=100(x_1^2-x_2)^2+(1-x_1)^2$$

Test function 3(T-3): this considers a function with five continuous variables, called the Rastrigin function (Hoffmeister and Bäck, 1991). It has a global minimum of zero at χ_1=χ_2=χ_3=χ_4=χ_5=0.0, and all of the variales should be considered as continuous values within the range 5.12 to +5.12.

$$f(x_1,x_2,x_3,x_4,x_5)=15+\sum_{i=1}^{5}(x_i^2-3\cos(2\pi x_i))$$

The applied results on the three test functions are listed in Table 3.

Table 3. Computational results for three test functions

	NGS			ANG			TSC (unit: sec.)		
	T-1	T-2	T-3	T-1	T-2	T-3	T-1	T-2	T-3
CGA	12	9	20	3840.1	2344.5	2000.0	0.40	0.10	1.20
AGA1	6	10	18	1744.4	2701.7	1864.7	0.13	0.20	1.13
AGA2	10	7	15	2952.3	1513.9	1789.2	0.47	0.17	2.33
Pro-AGA	5	4	10	1671.5	1346.6	1567.3	0.27	0.13	1.23

In terms of the NGS of Table 3, the performances of the AGA1 and AGA2 are slightly better than that of the CGA except for the case of the CGA and AGA1 in T-2. This means that the adaptive schemes of the AGA1 and AGA2 do not be well regulated in the three test functions. However, the performance of the proposed algorithm (pro-AGA) is significantly superior to those of the others, which implies that if the adaptive scheme using the FLC is applied to the CGA, the search quality of the CGA to locate the global optimum will be improved rather than that of the CGA without the adaptive scheme.

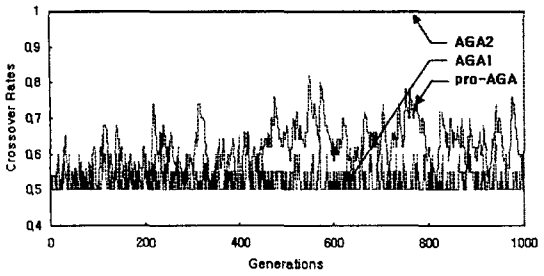
In the ANG, the three algorithms with the adaptive schemes are converged to the stop condition more rapidly than the CGA, except for the case of CGA and AGA1 in T-2. Especially, the pro-AGA shows the best performance when compared with the CGA and even the AGA1 and AGA2. This is closely related to the analysis result of the NGS: the more each algorithm is getting stuck to local optima, the more the average number of generation is increased.

In the TSC, the search times of all the algorithms show almost similar results, which means that the search schemes of the three AGAs are not as complicate as that of the CGA.

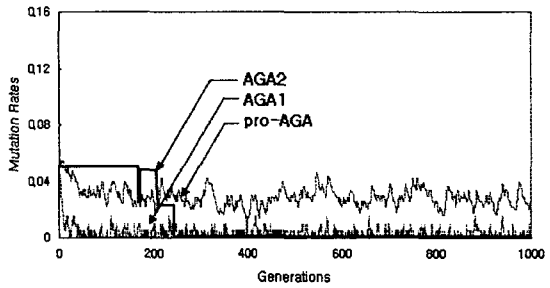
By the comparisons analysis mentioned above, we can find that the proposed algorithm (pro-AGA) shows better performances in terms of the NGS and ANG than the others. Especially, in the comparison among the adaptive algorithms, the pro-AGA outperforms the AGA1 and AGA2, which means that the adaptive scheme of the FLC used in the pro-AGA is more efficient than those of the heuristics in the AGA1 and AGA2.

For the detailed comparison on the adaptive schemes used in the three AGAs, we analyze the convergence behaviors of the crossover operator, mutation operator, and average fitness during the search process of each algorithm. Figures 5, 6, and 7 show the behaviors when the generation number of 1000 is reached.

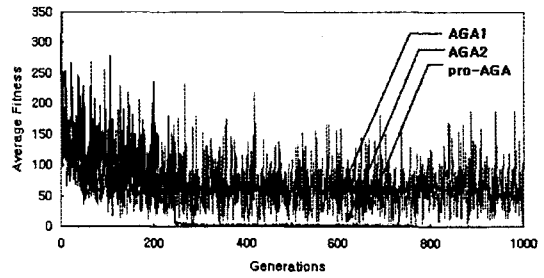
In the <Figure 5>, The AGA2 does not show any behavior in spite of using the adaptive scheme, while the behaviors of the AGA1 and pro-AGA show various behaviors. Similar result is also displayed in <Figure 6>. Especially, the pro-AGA shows more various behaviors in the two Figures 5 and 6 than the AGA1, which means that the scheme of the FLC used in the pro-AGA is able to more adaptively regulate the rates of the crossover and mutation operators than that of the heuristic in the AGA1. This is also proved in <Figure 7>: the AGA2 shows various behaviors before the generation number of 200, however, it is rapidly converged and does not show any various behaviors after the generation number; on the other hand, the behaviors of the AGA1 and pro-AGA show various behaviors over the genetic search process. In the comparison between the two algorithms (AGA1 and pro-AGA), especially, the pro-AGA has more various behaviors than the AGA1. This is mainly because the adaptive scheme of the FLC is more efficient than that of the heuristic used in the AGA1, like the analysis result of the Figures 5 and 6.



<Figure 5> Behaviors of crossover rate in T-2



<Figure 6> Behaviors of mutation rate in T-2



<Figure 7> Evolutionary behaviors of average fitness in T-2

Based on the various analysis results using the three test functions, we can conclude the following to be true:

(1) The overall comparison of the analysis results obtained from the four algorithms (CGA, AGA1, AGA2, and pro-AGA) has shown that the algorithms with the adaptive schemes are more efficient and more robust than the algorithm without it, though the CGA has better results than the AGA1 and AGA2 in several cases of Table 3.

(2) Applying the adaptive schemes in the algorithms (AGA1 and pro-AGA) enables the GA parameters (i.e., the rates of the crossover and mutation operators) to be adaptively regulated during the genetic search process, and also the average fitness to be variously changed. Especially, in the comparison among the algorithms with the adaptive schemes, the pro-AGA has showed the best performance in all the results of Table 3. This proves that the FLC used in the pro-AGA is more efficient than the heuristics in the AGA1 and AGA2

5. Conclusion

In this paper we have suggested the three GAs with adaptive schemes. For the adaptive schemes, we have used two heuristics from the conventional works and one FLC improved in this paper. Each algorithm has been tested and analyzed using three test functions, in order to compare the performance of their adaptive schemes. The analysis result has shown that the GA with any adaptive scheme is more efficient than the GA without it. Especially, the pro-AGA with the FLC as an adaptive scheme has shown considerately better performance than the AGA1 and AGA2 with

the heuristics. By these analysis results, we can conclude that applying the FLC for constructing an adaptive scheme is more robust than applying the two heuristics.

For future study, we have a plan to compare the results in large scaled problems and other adaptive schemes.

References

Bck, T., "Self-Adaptation in Genetic Algorithms, in Toward a Practice of Autonomous Systems", *Proceedings on 1st European Conference on Artificial Life*, Varela, F. J. and P. Bourgine, Eds. Cambridge, MA: MIT Press, (1992), 263-272.

Cheong, F. and R. Lai, "Constraining the Optimization of a Fuzzy Logic Controller Using an Enhanced Genetic Algorithm", *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, Vol. 30, No. 1, (2000), 31-46.

Davis, L., *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, 1991.

De Jong, K. A., *Analysis of the Behavior of a Class of Genetic Adaptive Systems*, Ph.D. Thesis, University of Michigan (University Microfilms No. 76-9381), 1975.

Eiden, A. E., R. Hinterding and Z. Michalewicz, "Parameter Control in Evolutionary Algorithms", *IEEE Transactions on Evolutionary Computation*, Vol. 3, No. 2, (1999), 124-141.

Gen, M. and R. Cheng, *Genetic Algorithms and Engineering Optimization*, John-Wiley & Sons, 2000.

Grefenstette, J. J., "Optimization of Control Parameters for Genetic Algorithms", *IEEE Transactions on Systems, Man, Cybernetics*, Vol. 16, No. 1, (1986), 122-128.

Hoffmeister, F. and T. Bäck, "Genetic Algorithms and Evolution Strategies: Similarities and Differences", In Schwefel, H. P. and R. Männer, eds. *Parallel Problem Solving from Nature*, Volume 496 of *Lecture Notes in Computer Science*, Dortmund (Germany), 1-3,October 1991. Springer-Verlag, Berlin. (*Proceedings of the 1st Workshop on Parallel Problem Solving from Nature* (PPSN1)), (1991), 455-471.

Hong, T. P. and H. S. Wang, "A Dynamic Mutation Genetic Algorithm", *Proceedings on the IEEE International Conference on Systems, Man, and Cybernetics*, Vol. 3, (1996), 2000-2005.

Hong, T. P., H. S. Wang, W. Y. Lin and W. Y. Lee, "Evolution of Appropriate Crossover and Mutation Operators in a Genetic Process", *Applied Intelligence*, Vol. 16, (2002), 7-17.

Mak, K. L., Y. S. Wong and X. X. Wang, "an Adaptive Genetic Algorithm for Manufacturing Cell Formation", *International Journal of Manufacturing Technology*, Vol. 16, (2000), 491-497.

Michalewicz, Z., *Genetic Algorithm + Data Structures = Evolution Programs*, Springer-Verlag, Second Edition, 1994.

Srinvas, M. and L. M. Patnaik, "Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms", *IEEE Transaction on Systems, Man and Cybernetics*, Vol. 24, No. 4, (1984), 656-667.

Subbu, R., A. C. Sanderson and P. P. Bonissone, "Fuzzy Logic Controlled Genetic Algorithms Versus Tuned Genetic Algorithms: An Agile Manufacturing Application", *Proceedings of the 1999 IEEE International Symposium on Intelligent Control (ISIC)*, (1998), 434-440.

Syswerda, G., "Schedule Optimization Using Genetic Algorithms", in *Handbook of Genetic*

Algorithms, L. Davis, Ed. New York: Van Nostrand Reinhold, (1991), 332-349.

Wang, P. T., G. S. Wang and Z. G. Hu, "Speeding Up the Search Process of Genetic Algorithm by Fuzzy Logic", *Proceeding of the 5th European Congress on Intelligent Techniques and Soft Computing*, (1997), 665-671.

Wu, Q. H., Y. J. Cao and J. Y. Wen, "Optimal Reactive Power Dispatch Using an Adaptive Genetic Algorithm", *Electrical Power & Energy Systems*, Vol. 20, No. 8, (1998), 563-569.

요약

유전알고리즘에서 적응적 연산자들의 비교연구

윤영수 · 서승록*

이 논문에서 우리는 유전알고리즘의 적응적 연산자에 대한 수행도를 비교한다. 이러한 적응적 연산자를 위해서, 유전알고리즘의 교차변이와 돌연변이 연산자가 고려되어 지며, 이 논문에서 개발된 하나의 퍼지로직 제어기와 기존연구에서 사용된 두개의 휴리스틱 기법이 제시되어진다. 이러한 퍼지로직 제어기와 두개의 기존 휴리스틱 기법들은 유전 탐색과정 동안에 그 연산자의 비율들을 적응적으로 조절한다. 이 논문에서 제시된 모든 알고리즘들은 수치예제에서 분석되어 지며, 결론적으로 이들 알고리즘 중에서 최적의 알고리즘이 추천된다.

* 대구대학교 자동차산업기계공학부