Journal of the Korean Institute of Industrial Engineers
Vol. 28, No. 1, pp. 76-86, March 2002.

퍼지로직제어에 의해 강화된 혼합유전 알고리듬

윤 영 수†

대구대학교 자동차산업기계공학부

Hybrid Genetic Algorithm Reinforced by Fuzzy Logic Controller

Young-Su Yun

School of Automotive, Industrial, and Mechanical Engineering, Taegu University, Kyungsan, 712-714

In this paper, we suggest a hybrid genetic algorithm reinforced by a fuzzy logic controller (flc-HGA) to overcome weaknesses of conventional genetic algorithms: the problem of parameter fine-tuning, the lack of local search ability, and the convergence speed in searching process. In the proposed flc-HGA, a fuzzy logic controller is used to adaptively regulate the fine-tuning structure of genetic algorithm (GA) parameters and a local search technique is applied to find a better solution in GA loop. In numerical examples, we apply the proposed algorithm to a simple test problem and two complex combinatorial optimization problems. Experiment results show that the proposed algorithm outperforms conventional GAs and heuristics.

Keywords: hybrid genetic algorithm, fuzzy logic controller, local search, combinatorial optimization problem

1. Introduction

Genetic algorithms (GAs) have proved to be a versatile and effective approach for solving optimization problems. Nevertheless, there are many situations where simple genetic algorithms do not perform particularly well: GA has a weakness in taking too much time to adjust fine-tuning structure of GA parameters and once the optimum solution region is identified by GA, finding a true optimum in the region using GA becomes inefficient or impossible due to the random nature of GA search.

To improve parameter-tuning problem, Gen and Cheng (2000) suggested using fuzzy logic controller (FLC). The pioneering work in extending the fuzzy logic technique to adaptively adjustment of the strategy parameters of GA were those of Lee and Takagi (1993), Xu and Vukovich (1994), Zeng and Rabenasolo (1997). Zeng and Rabenasolo used crossover rate, mutation rate, and crossover position of GA using a FLC. These parameters are considered input variables of GA and the parameters are taken the output variables of the controller.

Wang *et al.* (1997) used two FLCs: one for the crossover rate and the other for the mutation rate, and they suggested that the heuristic updating principle of the crossover rate and the mutation rate is to consider changes in the average fitness of the population.

To strengthen the lack of local search ability after convergence of GA and speed up the convergence speed, various methods for hybridization with GA have been developed. Davis (1991) and Ishibuchi *et al.* (1993) suggested random generating and test algorithm and multi-start descent algorithm for initializing GA populations, respectively. Gen and Cheng (1997) suggested the theoretical methodologies, Lamarckian evolution and memetic algorithms, for the complementary properties of GA and conventional heuristics. Li and Jiang (2000) presented a new stochastic approach SA-GA-CA based on proper integration of simulated annealing algorithm (SA), GA, and chemotaxis algorithm (CA) for solving complex optimization

† Corresponding author : Dr. Young-Su Yun, School of Automotive, Industrial, and Mechanical Engineering, Taegu University, Kyungsan, Kyungbook, 712-714, Korea; Fax +82-53-850-6549; e-mail joy629@hitel.net
Received September 2001, accepted December 2001. This paper was with the author 1 week for 1 revision.

problems. Choi *et al.* (1997) suggested a hybrid algorithm mixed by a binary integer programming and a genetic algorithm

According to these methodologies for various hybridizations, recently hybridized GAs have more robustness than conventional GAs or conventional heuristics (Gen and Cheng, 2000; Li and Jiang, 2000; Wu and Chow, 1995), Thus, we develop in this paper hybrid genetic algorithm with fuzzy logic controller (flc-HGA) to solve complex combinatorial optimization problems. In Section 2, the methodologies and the basic searching procedures of the proposed flc-HGA for hybrid search are suggested. The overall heuristic searching procedures of the flc-HGA are suggested in Section 3. Numerical examples to demonstrate the effectiveness of the flc-HGA are presented in Section 4 and conclusion follows in Section 5.

2. Hybrid Search

In this section, we suggest the methodologies and the basic searching procedures for the hybrid search of the proposed flc-HGA. first, GA procedures, main algorithm in this paper, are employed. Secondly, the local search to find better solution and the FLC to adaptively regulate the rates of crossover and mutation are proposed.

2.1 Genetic Algorithm

2.1.1 Representation and initialization

- **Representation**: In general, there are two types in representing variables of GA. The first type is bit-string representation and the second type is real-number representation (Gen and Cheng, 2000). Bit-string representation is the most common representation method used by GA researchers. However, this representation method probably is not employed in hybrid algorithms since most of conventional heuristics do not tend to use this method. This method also requires so much time for representing encoding and decoding procedures. Moreover, It probably gives the lack of precision search ability because of the precision problem in GA. In the proposed flc-HGA, we use thus the real number representation instead of bit-string representation in order to guarantee precision search of solution and speed up search process.
- **Initialization**: For initial populations, we use the population obtained by random search.

2.1.2 Genetic operators

- **Crossover**: non-uniform arithmetic crossover method (Gen and Cheng, 1997). We denote the two string selected randomly for crossover operation as v_1 and v_2, the offspring O_1 and O_2 will be

$$O_1 = [c \cdot v_1 + (1-c) \cdot v_2]$$
$$O_2 = [c \cdot v_2 + (1-c) \cdot v_1]$$

  where c is a random number in range [0, 1].
- **Mutation**: uniform mutation (Michalewicz, 1994). For a chosen parent v, if its gene m_3 is randomly selected for mutation, the resulted offspring is, $v' = [m_1, m_2, m_3', \cdots, m_t]$, where m_3' is a random (uniform probability distribution) value within $[m_3^L, m_3^U]$
- **Selection**: we use the mixed strategy with $(\lambda + \mu)$-selection and elitist selection in the enlarged sampling space (Gen and Cheng, 1997). This mixed strategy can make new population as many as population size for next generation after arranged as rank both μ parents and λ offspring.

2.2 Local Search Technique

GA can do global search in entire space but there are no ways for local search around the convergence area generated by GA loop, thus GA is sometimes impossible or insufficient finding a true optimum in the problems requiring complex and precision values. To overcome this weakness, various local search methods for hybridization of GA have been developed (Ishibuchi *et al.*, 1993; Li and Jiang, 2000)

One of the most common forms of hybrid GA is to incorporate a local search technique to the conventional GA loop. With this hybrid approach, local search technique is applied to each newly generated offspring to move it to a local optimum before injecting it into the new population (Davis, 1991). In the proposed flc-HGA, we also incorporate the improved hill climbing method to GA loop. This method improves the conventional hill climbing method suggested by Michelewicz (1994). The main difference between two methods is that the proposed method selects a optimal string among each string as many as population size in each GA loop, but the conventional method selects a current string at random, which makes the former having various search ability and good solution more than the latter. Thus, GA carries out global search and the improved hill climbing method carries out local search around the convergence area in each GA loop. In this hybrid strategy, the improved hill climbing method is used as an accelerator of GA.

The detailed procedures for the improved hill climbing method are suggested in Step 3 of Section 3.1.

2.3 Adaptively Regulation of Strategy Parameters using FLC

FLCs have been proved very useful when the search processes are too complex for analysis using conventional techniques or when the available sources of information are interpreted qualitatively, inexactly, or uncertainty. Recently, using these FLC's features there are many researches for fine-tuning of GA parameters. Most of these researches can adaptively regulate the GA parameters (crossover rate, mutation rate, and others), thus much time for fine-tuning of these parameters can be saved and the searching ability of GA in finding a global optimum can be improved more than conventional GA without FLC.

In the flc-HGA, we use Wang *et al.* concepts (1997). This method can make GA parameters, crossover (p_c) and mutation rate (p_m), adaptively regulate. The heuristic updating principle for the crossover rate and mutation rate is to consider changes in the average fitness of the population. For minimization problem, if we set the changes in average fitness at generation t and $t-1$ to $\Delta\overline{eval}(v;t)$ and $\Delta\overline{eval}(v;t-1)$, respectively. These two equations can be expressed as follows:

$$\Delta\overline{eval}(V;t) = \left(\frac{\sum_{k=1}^{pop\_size} eval(v_k;t)}{pop\_size} - \frac{\sum_{k=pop\_size+1}^{pop\_size+off\_size} eval(v_k;t)}{off\_size} \right) \times \lambda$$

$$\Delta\overline{eval}(V;t-1) = \left(\frac{\sum_{k=1}^{pop\_size} eval(v_k;t-1)}{pop\_size} - \frac{\sum_{k=pop\_size+1}^{pop\_size+off\_size} eval(v_k;t-1)}{off\_size} \right) \times \lambda$$

where $V=\{v_1, v_2, \cdots, v_k\}^T$

pop_size: population size satisfying constraints

off_size: offspring size satisfying constraints

λ: a scaling factor that regulates average fitness as problem type

The 'constraints' mentioned above mean the constraints subjected to the objective function for minimization or maximization. *pop_size* is the population size generated before crossover and mutation operators in GA loop and *off_size* is the offspring size generated after crossover and mutation operators. Especially, all the strings of *pop_size* and *off_size* in GA must satisfy the constraints mentioned above.

These values must be considered to adaptively regulate p_C and P_M as follows:

Procedure: *the changes of* p_C and P_M *using average fitness*

begin

if $\varepsilon \le \Delta\overline{eval}(V;t-1) \le \gamma$ and $\varepsilon \le \Delta\overline{eval}(V;t) \le \gamma$ then increase p_C and P_M for next generation;

if $-\gamma \le \Delta\overline{eval}(V;t-1) \le -\varepsilon$ and $-\gamma \le \Delta\overline{eval}(V;t) \le -\varepsilon$ then decrease p_C and p_M for next generation;

if $-\varepsilon < \Delta\overline{eval}(V;t-1) < \varepsilon$ and $-\varepsilon < \Delta\overline{eval}(V;t) \le \varepsilon$ then rapidly increase p_C and p_M for next generation;

end

end

where ε: a given real number in the proximity of zero

γ: a given maximum value of fuzzy membership function

$-\gamma$: a given minimum value of fuzzy membership function

The implementation strategy for crossover FLC is as follows:

2.3.1 Inputs and output of crossover FLC

The inputs to the crossover FLC are $\Delta\overline{eval}(V;t-1)$, $\Delta\overline{eval}(V;t)$ and the output of which is the change in crossover rate $\Delta c(t)$.

2.3.2 Membership functions of $\Delta\overline{eval}(V;t-1)$, $\Delta\overline{eval}(V;t)$ and $\Delta c(t)$

Membership functions of fuzzy input and output linguistic variables are illustrated in <Figure 1> and <Figure 2>. $\Delta\overline{eval}(V;t-1)$ and $\Delta\overline{eval}(V;t)$ are respectively normalized into the range in [−1.0, 1.0] and $\Delta c(t)$ is normalized into the range in [−0.1, 0.1] according to their corresponding maximum values.

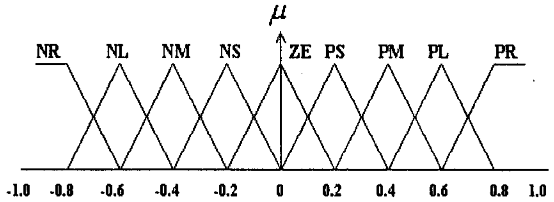


Figure 1. Membership function of $\Delta\overline{eval}(V;t-1)$ and $\Delta\overline{eval}(V;t)$.

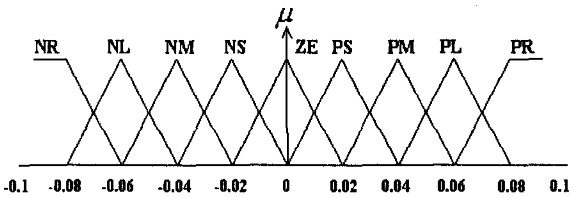


Figure 2. Membership function of $\Delta c(t)$.

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.

2.3.3 Fuzzy Decision Table

Based on a number of experiments and domain expert opinions, the fuzzy decision table was drawn as shown in <Table 1>.

2.3.4 Defuzzification table for control actions

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

Table 1. Fuzzy decision table for crossover

$\Delta c(t)$		$\Delta \overline{eval}(V;t-1)$								
		NR	NL	NM	NS	ZE	PS	PM	PL	PR
$\Delta \overline{eval}(V;t)$	NR	NR	NL	NL	NM	NM	NS	NS	ZE	ZE
	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	PM	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								
		-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

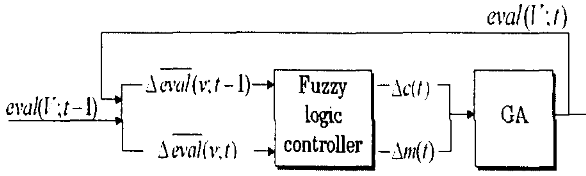


Figure 3. Coordinated strategy of FLC and GA.

The inputs of mutation FLC are the same as the crossover FLC and output of which is the change in mutation rate. The combination strategy used in this paper between FLC and GA is shown in <Figure 3>. The detailed procedures for adaptively regulation of GA strategy parameters are suggested in Step 4 of Section 3.1.

3. Proposed Algorithm

3.1 Heuristic procedure

In this section, we suggest the overall procedure of the proposed flc-HGA with a local search and a FLC. The heuristic search procedure for the flc-HGA is as follows:

■ flc-HGA procedure with a local search and a FLC

Step 1: Initial population
We use the populations obtained by random search

Step 2: Genetic Operators
- Crossover: non-uniform arithmetic crossover operator
- Mutation: uniform mutation operator
- Selection: the mixed strategy with $(\lambda+\mu)$-selection and elitist selection in enlarged sampling space

Step 3: Local Search Technique
Using offspring obtained after mutation operation of GA, the improved hill climbing method is applied as follows:

Procedure: the improved hill climbing method
begin
  $t \leftarrow 0$;
    repeat
      local ← FALSE;
      select a optimum string $\mathbf{v}_c$ among each string as many as population size of GA;
      repeat
        make new strings as many as population size in the neighborhood of $\mathbf{v}_c$;
        select a string $\mathbf{v}_n$ with the best fitness

```
        value of objective function f from
        the set of new strings;
        if f(v_c) < f(v_n)
           then v_c ← v_n
           else local ← TRUE
        end if
      until local
      t ← t+1;
   until t = population size
end
```

Step 4: Adaptively regulation of GA strategy parameters using FLC

- The input variables of the FLC for GA parameters are the changes in average fitness at generation t and $t-1$ as follows.

$$\overline{\Delta eval}(V;t),\ \overline{\Delta eval}(V;t-1)$$

- Determine control actions to $\overline{\Delta eval}(V;t)$ and $\overline{\Delta eval}(V;t-1)$ using fuzzy decision table (Wang *et al.*, 1997).
- After scaling the control actions by assigning the indexes i and j corresponding to the control actions in the defuzzification table (Wang *et al.*, 1997), calculate the changes of crossover rate $\Delta c(t)$ and 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 $\overline{\Delta eval}(v;t-1)$ and $\overline{\Delta eval}(v;t)$ for defuzzification.

- Update the changes of crossover rate and mutation rate by the following equations:

$$p_C(t) = p_C(t-1) + \Delta c(t),$$
$$p_M(t) = p_M(t-1) + \Delta m(t)$$

where $p_C(t)$ and $p_M(t)$ are respectively crossover rate and mutation rate at generation t.

Step 5: Termination condition
If the maximum generation number is satisfied, then stop; otherwise, go to Step 2.

Overall procedure of the flc-HGA is as follows:

Procedure: Overall procedure of flc-HGA

```
begin
   t ← 0;
   initialize P(t) using random search;
   while (not termination condition) do
      apply genetic operators into the P(t) to
      yield C(t);
      apply local search technique into C(t) to
      improve C(t);
      evaluate C(t);
      adaptively regulate genetic operation rates
      using FLC;
      select P(t+1) from P(t) and C(t) using
      enlarged selection strategy;
      t ← t+1;
   end
end
```

The concept of hybrid search strategy for the proposed flc-HGA is displayed in <Figure 4>.

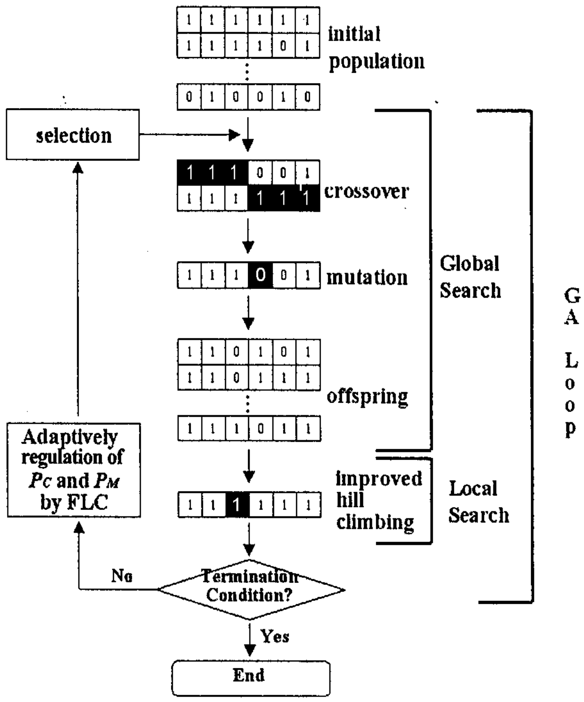


Figure 4. Hybrid search strategy of flc-HGA.

4. Numerical Example

In numerical example, we suggest a simple test problem and two complex combinatorial optimization problems. In the simple test problem we apply various experimental conditions to prove an ability of the proposed flc-HGA. In the two examples, we compare flc-HGA to several conventional heuristic algorithms and a conventional GA. The proposed procedures are implemented in Visual Basic language on IBM-PC P400 computer with 256M RAM.

4.1 A Simple Test Problem

This problem is firstly suggested by Michelewicz

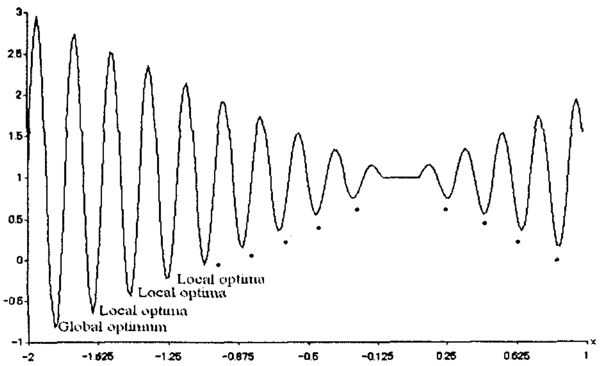


Figure 5. Searching space of test problem.

(1994) and usually used as test problem. The function is as follows:

$$\text{minimize } f(x) = -x_1 \sin(10\pi x_1) + 1$$
$$\text{subject to } -2.0 \le x_1 \le 1.0$$

This problem is a multimodel function with many local optima as <Figure 5> and optimal value has been known as $f(x) = -0.85027$ when x_1 is 1.8505.

We analyzed this problem under three conditions as <Table 3> and separated our flc-HGA into three parts: local search (LS), genetic algorithm with the FLC (GA with FLC), and the flc-HGA respectively, to prove the ability of hybrid algorithm and fuzzy logic controller. We also applied simple genetic algorithm (SGA) suggested by Goldberg (1989) to compare the flc-HGA with conventiónal genetic algorithm. The results applied shown in <Table 4>.

Table 3. Experimental conditions

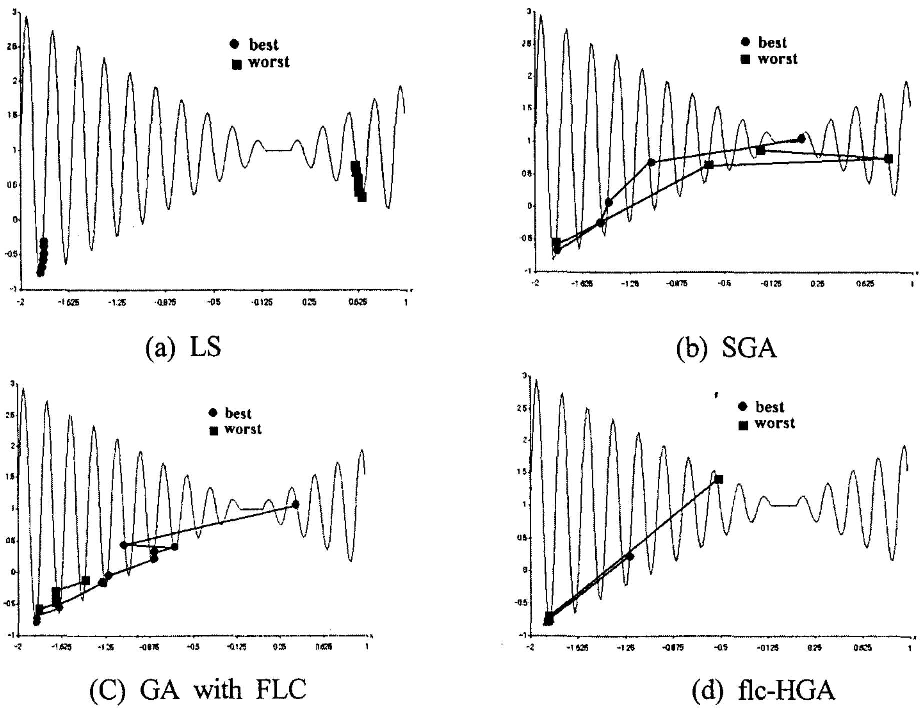
	No. of iteration	*Pop_size*	P_c	P_m	Searching range for local search
Case 1	1,000	20	0.5	0.1	0.01
Case 2	3,000	50	〃	〃	〃
Case 3	5,000	100	〃	〃	〃

In <Table 4>, the proposed flc-HGA finds the optimal value under all the conditions and is run faster than SGA and GA with FLC in average time. In the aspect of fitness, average fitness, best fitness, and worst fitness, the flc-HGA finds optimal solution under all the conditions. In the comparison of S. D., the flc-HGA and GA with FLC showed very small variations rather than the other two methods, the LS and SGA. These results prove that GA with FLC can get a better solution than GA without FLC and also the proposed flc-HGA is more effective and robust than

Table 4. Experimental results for a simple test problem

			LS	SGA	GA with FLC	flc-HGA
Case 1	Fitness	Best	-0.85027	-0.85006	-0.85027	-0.85027
		Average	-0.39044	-0.83853	-0.85026	〃
		Worst	+0.34922	-0.82234	-0.85019	〃
		S.D.	3.53 E-01	1.1 E-02	2.63 E-05	8.74 E-07
	Average Time		1	1	1	1
	Number of Optimal Search		None	None	6	10
Case 2	Fitness	Best	-0.85027	-0.85026	-0.85027	-0.85027
		Average	-0.59032	-0.84998	〃	〃
		Worst	-0.25040	-0.84928	〃	〃
		S.D.	2.12 E-01	3.01 E-04	3.11 E-07	3.98 E-07
	Average Time		1	16.8	4	4
	Number of Optimal Search		3	None	10	10
Case 3	Fitness	Best	-0.85027	-0.85027	-0.85027	-0.85027
		Average	-0.77028	-0.85026	〃	〃
		Worst	-0.45034	-0.85020	〃	〃
		S.D.	1.40 E-01	2.1 E-05	6.40 E-08	8.76 E-09
	Average Time		2	115.4	21	20.7
	Number of Optimal Search		7	3	10	10

* Time unit: second
* Average Time: average time after10 times run
* S.D: standard deviation after 10 times run.



(a) LS (b) SGA

(C) GA with FLC (d) flc-HGA

Figure 6. Convergence process to optimal solution.

the other three algorithms.

<Figure 6> shows the convergence process to optimal solution in best fitness and worst fitness of Case 1 in <Table 3>. In <Figure 6(a)>, we can find the global optimum in best result, but we can't find the global optimum in worst result. These results show the typical property of local search. In the case of SGA of <Figure 6(b)>, both the results can't reach the global optimum. This shows that SGA does well global search but can't do local search within convergence area. In <Figure 6(c)> and <Figure 6(d)>, the FLC is applied, which makes the methods finding the global optimum. However, <Figure 6(d)> can reach the global optimum very faster than <Figure 6(c)>. This shows that the flc-HGA has the ability of local search and thus is converged to global optimum very fast.

4.2 Design of the Gear Train

The problem of gear train was applied by Sandgran (1990), Fu *et al.* (1991), and Wu and Chow (1995). The mathematical formulation for this problem can be written as follows:

$$\text{minimize } f(x) = (1/6.931 - x_1 x_2 / x_3 x_4)^2$$
$$\text{subject to } 12 \le x_i \le 60 \quad i = 1, 2, 3, 4$$

Each variable must have integer value ranging from 12 to 60. For this problem Sandgran and Fu *et al.* applied nonlinear branch and bound (NB & B) and integer-discrete-continuous non-linear programming (IDCNLP), respectively. Wu and Chow suggested the Meta genetic algorithm (Meta-GA) that four parameters are to be optimized: population size, crossover rate, mutation rate, and crossover operator. Using these four combinations, they tried to find a optimal solution. <Table 5> shows optimal solutions obtained by the conventional heuristics (NB & B, IDCNLP), Meta- GA, and the flc-HGA of this paper. It is noted that the results of the flc-HGA and Meta-GA have a better solution than those of conventional heuristics (NB&B, IDCNLP). In the flc-HGA, the parameters used are as follows: population size = 10, initial crossover rate = 0.5, initial mutation rate = 0.1, termination condition = 2000, and search range = 0.5 for the improved hill climbing method.

For more detailed comparison of the flc-HGA, we used Simple GA (SGA) suggested by Goldberg (1989).

Table 5. Experimental results for the design of gear train

	NB&B	IDCNLP	Meta-GA	flc-HGA	Type of variables
x_1	18	14	19	16	Integer
x_2	22	29	16	19	Integer
x_3	45	47	43	49	Integer
x_4	60	59	49	43	Integer
$f(x)$	5.7×10^{-6}	4.5×10^{-6}	2.7×10^{-12}	2.7×10^{-12}	

Table 6. Fitness of the flc-HGA and SGA for the design of gear train

Fitness	Population Size			
	10			
	SGA	Time (sec.)	flc-HGA	Time (sec.)
Best	3.98E-08	2	2.3E-11	1.3
Average	3.40E-07	2.5	9.05E-10	1

Fitness	Population Size			
	50			
	SGA	Time (sec.)	flc-HGA	Time (sec.)
Best	1.16E-09	29	2.7E-12	9
Average	5.2E-08	30	1.28E-10	9.3

Fitness	Population Size			
	100			
	SGA	Time (sec.)	flc-HGA	Time (sec.)
Best	1.12E-08	53	2.7E-12	24
Average	7.40E-08	54	3.03E-11	24.4

The parameters of same condition are used in the flc-HGA and SGA, excepting population size. For population size, we divide it into three types: population size = 10, 50, 100, and each has 10 times run, the best and average fitness of the flc-HGA and SGA are listed in <Table 6>. Also for the SGA, crossover rate = 0.3, 0.5, 0.7 are used respectively and mutation rate = 0.05, 0.1, 0.3 are used, respectively. Thus the total number of parameter combination for the SGA, the number of different SGA, is 9(3 × 3). According to the population size (10, 50, 100), we also have 10 times run under each different SGAs for searching an optimal combination in these parameters. As applying result, crossover and mutation rates were 0.7, 0.03 respectively for each population size 10, 50, and 100. In <Table 6>, the best and average fitness of three types obtained by applying the flc-HGA are better than those of SGA, and computational times of flc-HGA in all population size is also better than those of SGA.

<Figures 7> and <Figure 8> also show that the flc-HGA and Meta-GA are more stable than SGA and converge rapidly. <Figures 9> and <Figure 10> show the behaviors of crossover rate (Pc) and mutation rate (P_m) as generations are proceed. In these Figures, the flc-HGA has various variations in P_c and P_m because of applying the FLC. These results can make the flc-HGA searching various search space. Thus, the flc-HGA seems to be more robust than conventional heuristics, Meta- GA, and SGA. To compare the evolution behaviors of average fitness, we checked the case with the FLC and the case without the FLC in the flc-HGA. In <Figure 11>, the case with FLC (a) converges more rapidly and variously than that of the case without the FLC (b). This means that the GA populations with the FLC are more diversified than those without the FLC. Thus the case with the FLC has more possibility to converge to an optimal solution than the case without the FLC in the flc-HGA.

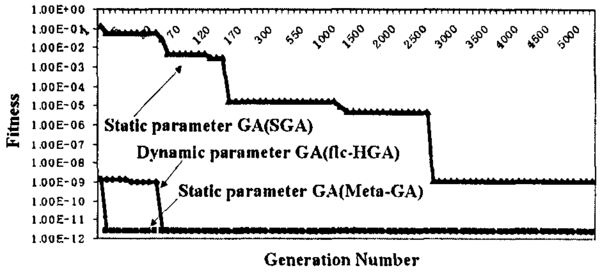


Figure 7. Convergence situation to optimal solution.

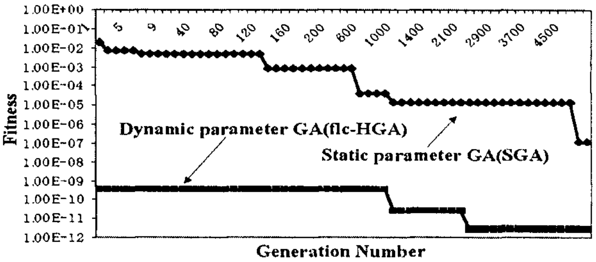


Figure 8. Convergence situation at population size = 100.

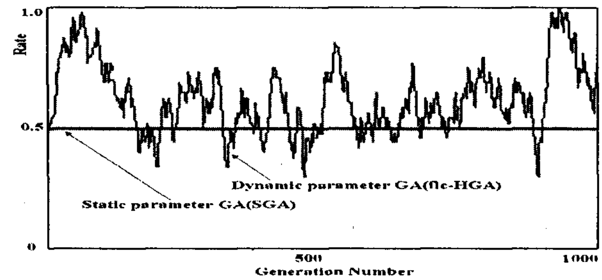


Figure 9. Behaviors of crossover rate in flc-HGA and SGA.

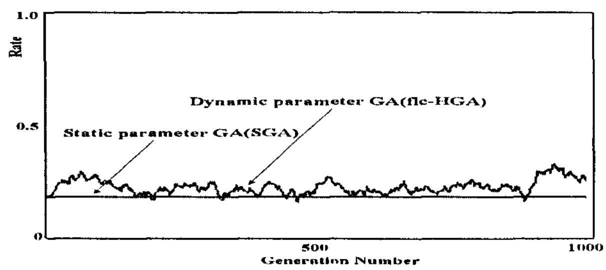


Figure 10. Behaviors of mutation rate in flc-HGA and SGA.

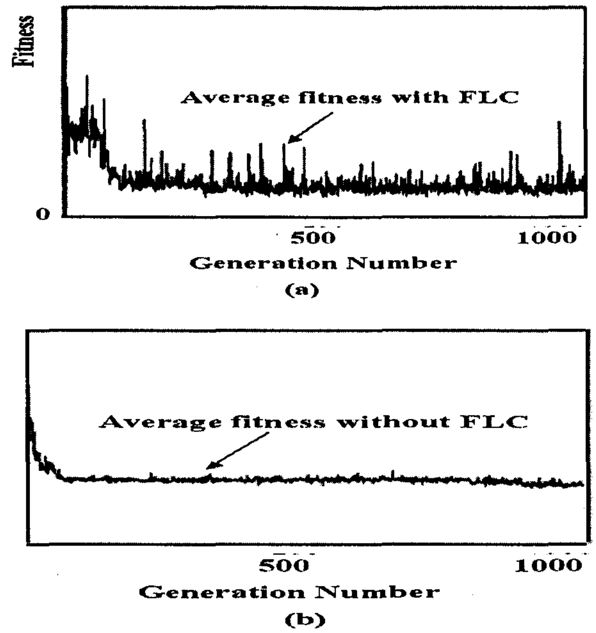


Figure 11. Evolutionary behaviors of the average fitness.

<Figure 9>, <Figure 10>, and <Figure 11> show the adaptively regulation abilities of the FLC. In <Figure 11(a)>, it is shown that the average fitness rapidly converges between the initial generation and the 100th generation. This means that the rates of P_c and P_m should be rapidly increased at the same generation. We can also find these same situations in <Figure 9> and <Figure 10>. According to these facts, we can prove that the proposed flc-HGA satisfy well the concepts of the FLC used in this paper.

4.3 Design of the Coil Compression Spring

The problem of designing coil compression spring was applied by Sandgran (1990), Chen and Tsao (1993), and Wu and Chow (1995). The mathematical formulation of this problem can be written as follows:

minimize $f(x) = \pi^2 x_2 x_3^2 (x_1 + 2)/4$

subject to

$$g_1(x) = (8C_f F_{max} x_2 / \pi x_3^3) - s \le 0$$
$$g_2(x) = l_f - l_{max} \le 0$$
$$g_3(x) = d_{min} - x_3 \le 0$$
$$g_4(x) = x_2 + x_3 - D_{max} \le 0$$
$$g_5(x) = 3.0 - (x_2/x_3) \le 0$$
$$g_6(x) = \delta_p - \delta_{pm} \le 0$$
$$g_7(x) = \delta_p + (F_{max} - F_p)/K + 1.05(x_1 + 2)x_3 - l_f \le 0$$
$$g_8(x) = \delta_w + (F_{max} - F_p)/K \le 0$$

The parameters used above are as follows:

$\delta_p = F_p / K$ $\quad K = G x_3^4 / 8 x_1 x_2^3$

$l_f = F_{max} / K + 1.05(x_1 + 2)x_3$ $\quad F_{max} = 1000$ lb

$S = 189,000$ psi. $\quad l_{max} = 14.0$ inch

$d_{min} = 0.2$ inch $\quad D_{max} = 3.0$ inch

$F_p = 300.0$ lb $\quad \delta_{pm} = 6.0$ inch

$\delta_w = 1.25$ inch $\quad G = 11.5 \times 10^6$ psi.

$$C_f = (4(x_2/x_3) - 1)/(4(x_2/x_3) - 4) - 0.165 x_3 / x_2$$

In this problem, the design variables are considered by integer, continuous, and discrete variables. Especially, discrete variables are used pre-defined discrete dimensions. For this problem, Sandgran (1990) and Chen and Tsao (1993) applied nonlinear branch and bound (NB & B) and simple GA (GA), respectively. Wu and Chow (1995) applied Meta-GA in which used same method along with the design of the gear train. And also the flc-HGA followed the same method. The results applied are shown in <Table 7>. For more detained comparison, we used the same conditions in the design of the gear train. The best and average fitness of the flc-HGA and SGA are shown in <Table 8>.

In <Table 7>, the result of the flc-HGA is better than those of the NB & B, GA, and Meta-GA. This result implies that the flc-HGA obtains a better solution than the NB & B, GA and Meta-GA in the continuous and discrete variables, and also implies that the local search technique apply into GA loops for precision search in convergence region by GA. <Table 7> also shows that the flc-HGA have more effectiveness and robustness than the NB&B, GA and Meta-GA because the best and average fitness of the flc-HGA are better than those of the NB&B, GA, Meta-GA, and SGA in <Table 7> and <Table 8> under all the conditions.

These results also proved in <Figures 12> and <Figure 13>. The various behaviors of crossover rate and mutation rate are shown in <Figures 14> and <Figure 15>. In <Figure 16>, the evolutionary behaviors of average fitness with the FLC (a) and that of average fitness without the FLC (b) in the flc-HGA are showed. Average fitness with the FLC is more various and is converged more rapidly than that without the FLC (b).

Table 7. Experimental results for the design of coil compression spring

	NB & B	GA	Meta-GA	flc-HGA	Type of variables
x_1	10	9	9	9	Integer
x_2	1.181	1.229	1.227	1.109	Continuous
x_3	0.283	0.283	0.283	0.263	Discrete
$f(x)$	2.7995	2.671	2.668	2.082	

Table 8. Fitness of flc-HGA and SGA for the design of coil compression spring

Fitness	Population Size			
	10			
	SGA	Time (sec.)	flc-HGA	Time (sec.)
Best	2.09	2	2.083	5
Average	2.19	2	2.095	5

Fitness	Population Size			
	50			
	SGA	Time (sec.)	flc-HGA	Time (sec.)
Best	2.14	41	2.082	16
Average	2.26	41	2.083	16.4

Fitness	Population Size			
	100			
	SGA	Time (sec.)	flc-HGA	Time (sec.)
Best	2.10	128	2.082	40
Average	2.11	129	2.085	40

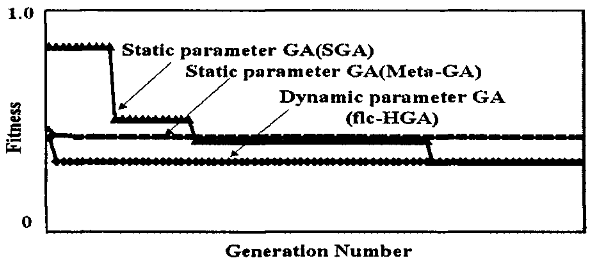


Figure 12. Convergence situation to optimal solution.

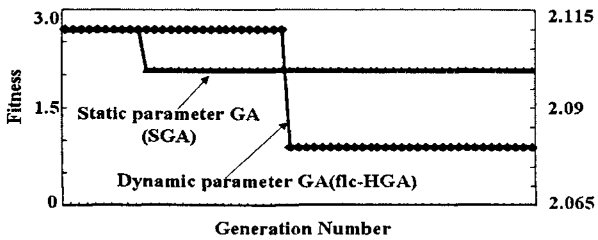


Figure 13. Convergence situation at population size = 100.

According to these three numerical examples, it is proved that the proposed flc-HGA applied by the FLC and a local search technique is more effective and robust than conventional heuristics (NB & B, IDCNLP), Meta- GA and SGA.

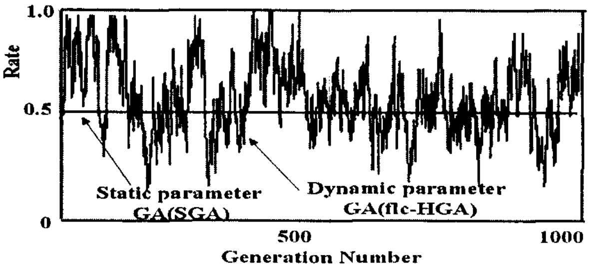


Figure 14. Behaviors of crossover rate in flc-HGA and SGA.

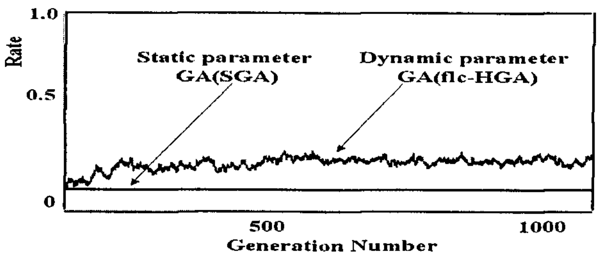


Figure 15. Behaviors of mutation rate in flc-HGA and SGA.

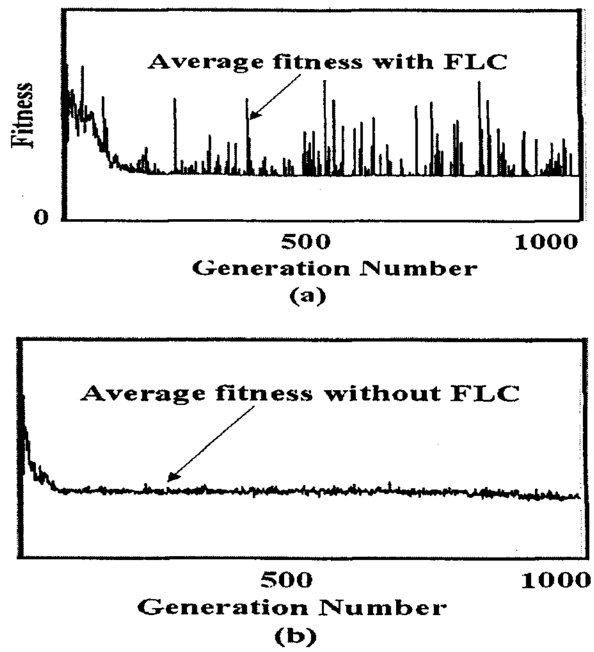


Figure 16. Evolutionary behaviors of the average fitness.

5. Conclusion

In this paper, we proposed a new genetic algorithm (flc-HGA), the hybrid genetic algorithm with a FLC and a local search technique, to overcome the weakness of conventional GAs: the problem of parameter fine-tuning and premature convergence of solution process. In the flc-HGA, the FLC to adaptively regulate GA parameters (crossover rate and mutation rate) and the local search technique to find a better solution are combined with GA, respectively. Numerical comparison experiments demonstrated that the proposed algorithm is more effective and has more robustness than the conventional heuristics and conventional genetic algorithm.

References

Chen, J. L. and Tsao, Y. C. (1993), Optimal Design of Machine Elements using Genetic Algorithms, *Journal of the Chinese Society of Mechanical Engineers*, **14**(4), 193-199.

Choi, I. C., Kim, S. I. and Hwang, D. H. (1997), New Mathematical Formulations and An Efficient Genetic Algorithm for Finding a Stable Set in a Competitive Location Problem, *Journal of Korean Institute of Industrial Engineers*, **23**(1), 223-234. (Korean Version)

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

Fu, J. F., Fenton, R. G. and Cleghorn, W. L. (1991), A Mixed Integer-Discrete-Continuous Programming Method and Its Applications to Engineering Design Optimization, *Engineering Optimization*, **17**, 263-280.

Gen, M. and Cheng, R. (1997), *Genetic Algorithms and Engineering Design*, John-Wiley & Sons.

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

Goldberg, D. E. (1989), *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison Wesley.

Ishibuchi, H., Yamamoto, N., Murata, T. and Tanaka, H. (1993), Genetic Algorithm and Neighborhood Search Algorithms for Fuzzy Flowshop Scheduling Problems, *Fuzzy Sets and Systems*, **67**, 81-100.

Lee, M. and Takagi, H. (1993), Dynamic Control of Genetic Algorithm Using Fuzzy Logic Techniques, *Proceedings of the 5th International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers, San Francisco, 76-83.

Li, B. and Jiang, W. (2000), A Novel Stochastic Optimization Algorithm, *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, **30**(1), 193-198.

Michalewicz, Z. (1994), *Genetic Algorithms + Data Structures = Evolution Programi*, Second Extended Edition, Spring-Verlag.

Sandgren, E. (1990), Nonlinear Integer and Discrete Programming in Mechanical Design Optimization, *ASME Journal of Mechanical Design*, **112**(2), 223-229.

Wu, S. J. and Chow, P. T. (1995), Genetic Algorithms for Nonlinear Mixed Discrete-Integer Optimization Problems via Meta-Genetic Parameter Optimization, *Engineering Optimization*, **24**, 137-159.

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

Xu, H. and Vukovich, G. (1994), Fuzzy Evolutionary Algorithm and Automatic Robot Trajectory Generation, *Proceedings of the First IEEE Conference on Evolutionary Computation*, IEEE Press, Piscataway, NJ, 595-600.

Zeng, X. and Rabenasolo, B. (1997), A Fuzzy Logic Based Design for Adaptive Genetic Algorithms, *Proceeding of the 5th European Congress on Intelligent Techniques and Soft Computing*, 660-664.