A New Genetic Algorithm Using Large Mutation Rates and Population-Elitist Selection (GALME)

Hisashi Shimodaira
Department of Information and Communication, Bunkyo University
1100 Namegaya, Chigasaki-City, Kanagawa 253, Japan

Abstract

Genetic algorithms (GA's) are one of promising means for function optimization. Methods for function optimization are required to perform local search as well as global search in a balanced way. It is, however, recognized that the traditional GA is not well suited to local search. I have tested algorithms combining various ideas to develop a new genetic algorithm to obtain the global optimum effectively. As a result, it is turned out that the performance of a genetic algorithm using large mutation rates and population-elitist selection (GALME) is superior. This paper describes the GALME and its theoretical justification, and presents the results of experiments, compared to the traditional GA. Within the range of the experiments, it is turned out that the performance of GALME is remarkably superior to that of the traditional GA.

1. Introduction

Genetic algorithms (GA's) are one of stochastic multi-point search methods and are promising methods for function optimization. Methods for function optimization are required to converge to the global optimum not getting trapped on local optima. For this purpose, they are required to perform local search as well as global search in a balanced way. Grefenstte [1], however, pointed out the following concerning this.

··· it is widely recognized that GA's are not well suited to performing finely-tuned local search. Like natural systems, GA's progress by virtue of changing the distribution of high performance substructures in the overall population; individual structures are not the focus of attention. Once the high performance regions of the search space are identified by the GA, it may be useful to invoke a local search routine to optimize the members of the final population.

From such a background, many researches have been performed to improve the performance of GA's. For example, variants of GA's different from the traditional GA were proposed by Eshelman [2], Syswerda [3], Whitley [4], and Mahfoud [5]. In the framework of the traditional GA, attempts at controlling mutation or crossover rate to improvement the performance of GA have been performed by Sirag [6], Fogarty [7], Bäck [8], and Booker [9]. Research on methods combining a GA and simulated annealing for local search were performed by Brown [10], and so on. Research on a GA based on a new framework was performed by Goldberg [11].

I have performed experiments on algorithms combining various ideas to develop a GA by which the global optimum of functions is obtained effectively. As a result, I have developed a new GA using large mutation ,rates (in some cases, they are controlled by a decreasing function of generation) and population-elitist selection whose performance is superior to that of the traditional GA. I call this algorithm GALME (Genetic Algorithm Using Large Mutation Rates and Population-Elitist Selection). This paper describes the GALME and its theoretical justification and presents the results of the experiments, comparing with the traditional GA.

2. The traditional GA and the simulated annealing method

I describe the outline of the traditional GA and the simulated annealing method to facilitate the later explanation.

Fig.1 shows the framework of the traditional GA (based on Eshelman [2], but modified), where t is generation, structure the genotype of an individual, and P(t) and C(t) populations of gen-

0-8186-7686-8/96 $05.00 © 1996 IEEE

```
begin
   $t=0$;
   initialize P($t$);
   evaluate structures in P($t$);
   while (termination condition not satisfied) do
   begin
      $t=t+1$;
      select$_r$ P'($t-1$) from P($t-1$);
      recombine structures in P'($t-1$) forming C($t$);
      evaluate structures in C($t$);
      select$_s$ P($t$) from C($t$) and P($t-1$);
   end;
end;
```

Fig. 1 Skeleton of the traditional GA

eration t. In the traditional GA, the following processing is performed. (1) The number N of individuals of the population is constant and the population is initialized using random numbers. (2) The selection for reproduction (select$_r$) is biased toward selecting the better performing individuals. (3) The recombination is performed using crossover and mutation based on probability. The recombination operator is either one- or two-point crossover. A low rate of mutation is used in the recombination stage to maintain population diversity. (4) The selection for survival (select$_s$) is usually unbiased, typically replacing the entire parent population P(t-1) with the child population C(t). When a generation gap G [12] is considered, N(1-G) individuals of P(t-1) are chosen (at random) to survive intact into P(t). When an elitist strategy is employed, the individual with the best performance always survives intact into the next generation.

The simulated annealing method (SA) is a optimization method in which simulation similar to physical annealing process is performed. It deals with only one solution unlike GA's. In the SA, a candidate solution is evaluated by the cost. In case of searching for the minimum value, a good solution has smaller cost than bad one. When a solution j with cost f(j) are produced by applying neighborhood search operator to the present solution i with cost f(i), the probability p with which the solution j is accepted as the next solution, depends on the temperature T and the difference of the costs as follows:

$$\begin{aligned}&\text{if } (f(j) \leqq f(i)) \quad p=1.0,\\ &\text{if } (f(j) > f(i)) \quad p=\exp[f(i)-f(j)/T]. \qquad (1)\end{aligned}$$

Approximate solutions are searched according to such a process until the conditions for convergence are satisfied. In the initial stage, temperature is set high and is gradually reduced. When temperature is high, even solutions with large cost are accepted. As temperature becomes lower, solutions with large costs are not almost accepted and the solution converges. The feature of SA is that a solution that is worse than the present one is accepted as the next solution with probability controlled by temperature. The convergence of SA has been proved under some conditions [13].

3. Previous researches

To clarify the differences between the GALME and previous research, I describe outlines of previous researches attempting to improve the performance of GA's, compared with the traditional GA.

For Eshelman's CHC [2], in the select$_r$ (see Fig.1), P'(t-1) is formed by randomly choosing two individuals as a pair without replacement from P(t-1) and by mating them. Before mating, however, the Hamming distance between them is calculated, and if half that distance does not exceed a difference threshold, they are not mated (incest prevention). The threshold is reduced according to the condition of convergence of the population. As the recombination operator, a highly destructive variant HUX of uniform crossover, that produces offsprings that are maximally different from both parents, is employed. In the stage of recombination, mutation is not used. In the selects, C(t) and P(t-1) must compete with each other and P(t) is formed by choosing N individuals from them according to their fitness values. Eshelman calls this population-elitist selection. When the population converges without reaching the global optimum, the cycle is restarted after individuals of the population are generated by applying mutation with a large rate to the individual with the largest fitness value. The feature of the CHC is that cross-generation competition is employed in the survival selection instead of biasing toward selecting the better performing individuals in the reproduction selection. The performance of the CHC is remarkably superior that of the traditional GA. Its algorithm is, however, considerably complicated and the computational cost for calculating Hamming distances is expensive.

For Syswerda's steady-state GA [3], in the select$_r$, two individuals are chosen as a pair according to the fitness and two offsprings are produced by applying crossover and mutation to them. Two individuals of the population are deleted and the produced offsprings are inserted. The selection for deletion is done through inverse ranking, starting with the worst individual in the population. For Whitley's GENITOR algorithm [4], one of the offsprings is inserted into the population. The feature of these algorithms is that one or two offsprings at a time are inserted into the population without using the concept of generation. They, however, lack in theoretical clarity, because the relation between the magnitudes of fitness values of the deleted and inserted individuals is not considered.

For Mahfoud's parallel simulated annealing [5], all individuals of the population are paired by randomly choosing two individuals without replacement. Two offsprings are generated by applying crossover and mutation to each pair. Tournaments between the offsprings and their parent are held and individuals in the next generation are chosen using the acceptance probability like Eq.(1). The mutation rate is a decreasing function of generation. The feature of this algorithm is that the technique of SA is employed for the survival selection and the mutation rate is externally controlled by a decreasing function of generation. In the selection for survival, however, the relation among the magnitudes of fitness values of each individuals in the entire population is not considered.

The outlines of researches attempting to improve the performance of the traditional GA are as follows. Sirag [6] proposed a method that employs crossover and mutation rates externally controlled by expressing them by functions of temperature varied like the SA. Fogarty [7] reported that the performance improved by changing the mutation rate in cases of particular applications. Bäck [8] proposed a method that employs mutation rates adaptively changed by making a part for expressing it in the genotype. Booker [9] proposed a method that maintains diversity of the population by adaptively changing the crossover rate.

4. GALME

While the traditional GA has good performance for optimization of unimodal functions, it has a problem in case of multimodal functions that the solutions are apt to being trapped on local minima and difficult to escape from them. Therefore, in order to improve the performance of GA's the algorithm needs to have ability to robustly explore the solution space and to escape from local mimina like the SA when being trapped on them. From such a standpoint, I have performed experiments on algorithms combining various ideas. As a result, it is turned out that the performance of the GALME, that employs large mutation rates (in some cases, they are controlled by a decreasing function of generation) and population-elitist selection, is remarkably superior to that of the traditional GA.

```
begin
   $t$=0;
   initialize P($t$);
   evaluate structures in P($t$);
   while (termination condition not satisfied) do
   begin
      $t$=$t$+1;
      calculate $p_m$ for $t$;
      select$_r$ P'($t$-1) from P($t$-1) by randomly
        pairing all individuals;
      apply mutation with $p_m$ and crossover to each
        pair in P'($t$-1) forming C($t$);
      evaluate structures in C($t$);
      select$_s$ P($t$) from C($t$) and P($t$-1) according
        to their fitness values;
   end;
end;
```

Fig.2 Skeleton of the GALME

The skeleton of the GALME is shown in Fig.2. The number of individuals of the population is N. The population is initialized using uniform random numbers. The mutation rate p_m is constant or externally controlled by a decreasing function of generation t. In the selection for reproduction select$_r$, all individuals of P(t-1) are paired by choosing two individuals without replacement to form P'(t-1). By applying mutation with probability pm and crossover to individuals of each pair, C(t) is produced. In the selection for survival selects, N individuals are chosen according to their fitness values from the population merging C(t) and P(t-1) to form P(t). The cycle in Fig.2 is repeated within the maximum number of generations (*GMAX*). If the global optimum is not obtained within *GMAX*, the cycle is ended and restarted after initializing the population.

In the traditional GA, mutation is a background operator, assuring that the crossover has a full range alleles so that the adaptive plan is not trapped on local optima [14] and generally small mutation rates are used. On the other hand, in the GALME, in order to form C(t) mutation is applied with considerably large probability to pairs of P'(t-1) and crossover is always applied to them. The reason is to produce offsprings that are as different as possible from their parents and to examine regions of the search space not yet explored. Expressing the mutation rate by a function of generation and taking large values in the initial stage also enables the GALME to robustly explore the search space and to escape from local optima when getting trapped on them. Namely the GALME can escape from local optima by performing searches with a large range like the SA. Gradually reducing the mutation rate enables the GALME to find the global optima by performing local search using good solutions obtained so far.

Generally mutation and crossover have a side-effect that it destroys good performing schemata. Because the influence of this side-effect is small for the population-elitist selection [2], the GALME can use large mutation rates and always apply crossover. According to the result of simulations by Bäck [8] in which a self-adaptive mutation rate is used, the value of the optimal mutation rate is considerably large and is a decreasing function of generation. According to the theoretical research by Hesser [15], the optimal mutation rate is a negative exponential function of time. These results theoretically and empirically justifies the way of expressing and taking the mutation rate employed in the GALME.

In the experiments in Section 5, I employed the following types of controlling mutation rate:

$$\begin{aligned}
&\text{Type 1} \quad p_m=\text{constant};\\
&\text{Type 2} \quad p_{m,t+1}=\beta p_{m,t}, \quad \text{if } p_0 \geq p_{m,t} > p_{min};\\
&\text{Type 3} \quad p_{m,t+1}=\beta_1 p_{m,t}, \quad \text{if } p_0 \geq p_{m,t} \geq p_b;\\
&\qquad\qquad p_{m,t+1}=\beta_2 p_{m,t}, \quad \text{if } p_b > p_{m,t} > p_{min}. \qquad (3)
\end{aligned}$$

where p_0, p_b and p_{min} are the initial, boundary and minimum values of mutation rate, respectively and β, β_1 and β_2 are the reduction coefficients. Type 2 is a way of reducing the mutation rate in one stage and type 3 in two stages. If $p_{m,t+1}$ becomes smaller than p_{min}, p_{min} is used as p_m thereafter.

In the traditional GA, better performing individuals produce multiple offsprings. On the other hand, in the GALME, the chance for each individual to become a parent is one time in spite of its performance and the parents and the produced offsprings must compete to survive. They can survive if their fitness values are within N from the top. In the traditional GA, therefore the number of individuals containing better performing schemata increases much faster than that containing worse performing schemata. In the GALME, however this speed of the increase is not as fast but its number steadily increases. Eshelman [2] called this the weak version of implicit parallelism. This effectively acts to avoid getting trapped on local optima. Namely in the traditional GA schemata concerning the dominating local optimum increase rapidly and eventually dominate the population, whereas in the GALME because the speed is not as rapid, changing individuals of the population by applying mutation with large rates enables it to escape from that local optimum. On the other hand, although that the speed of increase in better performing individuals is not high reduces the speed of reaching the global optimum, the improvement of the speed by avoiding local minima compensates this and the total improvement of the performance could eventually achieved.

Crossover and mutation produce offsprings different from their parents to explore regions of the search space not yet explored, whereas they may side-effectively destroy better performing schemata obtained so far. The influence of this side-effect on the GALME is decisively different from that on the traditional GA. In the traditional GA, even an offspring that has worse performance than its parent always replaces its parent. In case of elitist strategy, only small elite can survive intact into the next generation. On the other hand, in the GALME if the performance of an offspring is worse than that of its parent, it can never replace its parent. Therefore an individual containing better performing schemata can survive until the order of its fitness value becomes smaller than N from the top.

In the GALME individuals that survived can always become parents and produce their offsprings in turn. Because crossover and mutation rates are large, variations of from small to large ranges are applied to individuals. When a small variation is applied to an individual, its neighborhood can be examined. When a large variation is applied to an individual, a region not yet explored can be examined to result in global search. In such a way local as well as global searches can be performed in parallel and better performing individuals are accumulated in the population.

In the GALME even an individual whose performance is not as good can survive, if the order of its fitness value is within N from the top and always becomes a parent. Even if the performance of an individual containing a schema concerning the global optimum is not as high in a stage, this gives the individual a chance by which it can produce an offspring having a fitness near to the global optimum. This mechanism is similar to that of the SA that can escape from local optima by accepting a solution whose performance is worse than the present solution. It is conjectured, from the similarity between such an effect of the population-elitist selection and the SA, that escaping from local optima could be further promoted by the effect arising from that the mutation rate is changed by a decreasing function of generation.

That methods employed in the GALME cause the above effects is a hypothesis and needs to be theoretically and experimentally verified.

If the number of repetitions reaches *GMAX* and the global optimum is not yet obtained, the cycle is restarted after resetting the mutation rate to the initial values and initializing the population. For the method of initializing population, I performed preliminary experiments with the multimodal function shown in 5.1 using the following three methods: (1) the method of using the very population obtained so far, (2) the method of surviving intact two individuals with the best performance and generating all other individuals using random numbers, (3) the method of generating all individuals using random numbers. When getting trapped on a local optimum, with the method (1) the population could not escape from it irrespective of repetitions of restart, whereas with the methods (2) and (3) the population could reach the global optimum by restarts. In the following experiments, the method (3) was employed, because the performance of the method (3) was a little better than that of the method (2).

The comparison of the amount of computations for the traditional GA and the GALME is as follows. The number of function evaluations is N times per a generation for both the methods. The amount of computation for the selection for reproduction is almost the same for both the methods. That for the selection for survival for the GALME is larger than that for the traditional GA by the amount corresponding to the computation for choosing N individuals from $2N$ individuals.

The originality of the GALME is to have developed a new GA to obtain global optimum effectively by combining ideas proposed in previous researches: Eshelman's ideas [2] of the selection for reproduction and survival and Mahfoud's [5] and Hesser's [15] ideas of expressing the mutation rate by a decreasing function of generation. The algorithm of the GALME is almost as simple as that of the traditional GA and is suitable for parallel processing.

5. Experimental results

The performance of the GALME has been compared with that of the traditional GA on two functions: a multimodal function [2], [16] and a deceptive function [2], [11] that were often used because they are difficult to optimize. Each function has only one global optimum and it was searched by GA's. I performed 20 runs per a computation condition changing seed values for the random number generator. The same 20 seed values were used for each corresponding run in different cases. The run was continued until the global optimum was found by at least one individual (I call this the convergence) or until the maximum number of repetitions was reached. The maximum number of repetitions was $GMAX$=10000 for the GALME without restart and the traditional GA and was 10000 in total for the GALME with restart. The performance was evaluated by the number of runs out of 20 that the algorithm succeeded in finding the global optimum value (convergence) and the average number of function evaluations to find the optimum in those runs where it found the optimum. (An algorithm performs better on a function if it finds the global optimum more often, or it finds the global optimum the same number of times as its competitor but in fewer evaluations.) The reason why the number of function evaluations was used is because the amount of computation for function evaluations is generally larger than that of GA itself.

Two-point crossover was used for both the GALME and the traditional GA. For the GALME, preliminary experiments were performed to compare the performance of two-point crossover and the HUX [2] using the following two functions. Because two-point crossover performed better than the HUX for both the functions, the former was used in the following experiments.

For the GALME, I examined the best-performing parameter values changing little by little the mutation rate and its reduction coefficients for the three types of its controlling. For the traditional GA, I examined the best-performing parameter values changing little by little their parameter values starting with the standard parameter values by De Jong [12] and the parameter values based on off-line performance by Grefenstette [12]. The former is the set of N=50, the crossover rate p_c=0.6, the mutation rate p_m=0.001, G=1.0, scaling is not used, and elitist strategy is employed. The latter is the set of N=80, p_c=0.45, p_m=0.01, G=0.9, scaling with fitness values before one generation is used, and elitist strategy is not employed.

5.1 Multimodal function

The following function [16] was used:

$$f6 = 0.5 + \frac{0.5 - \sin^2\sqrt{x^2+y^2}}{[1+0.001(x^2+y^2)]^2} \qquad (3)$$

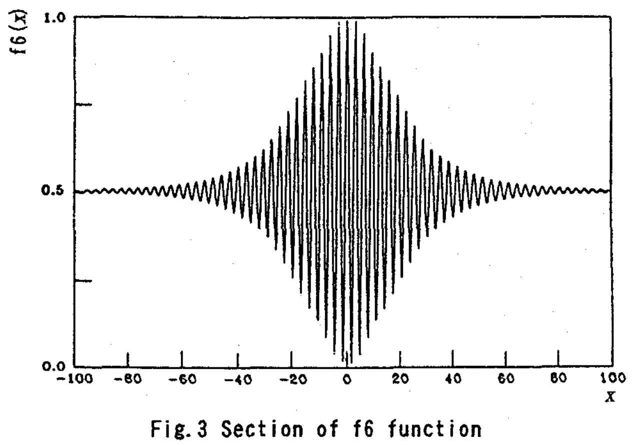

Fig. 3 Section of f6 function

Table 1 Performance for function f6 (Number of convergence to global optimum and average of function evaluation times)

Algorithm	Computation condition	Num. of converg.	Aver. of func. eval. times
GALME	N=100, Mutation=Type1 (p_0=0.08), $GMAX$=10000, Restart=No	18	67466
	N=100, Mutation=Type2 (p_0=0.3, p_{min}=0.01, β=0.9995), $GMAX$=10000, Restart=No	20 (all)	200180
	N=100, Mutation=Type3 (p_0=0.15, p_b=0.06, p_{min}=0.01, β_1=0.99088, β_2=0.99985), $GMAX$=300, Restart=Yes	20 (all)	22315
Traditional GA	N=50, p_c=0.6, p_m=0.001, G=1.0, Scaling=No, Selection strategy=Elitist selection, $GMAX$=10000	9	26066

This function is cylindrically symmetric about the z axis and has the maximum value at the origin. Fig.3 shows a section for y=0 including the z axis. The points in the search space were coded as Cartesian x and y values in the range -100 to +100 with 22-bit binary code. The maximum value to be searched by GA's in this case is the nurhber with eight 9's below the floating point.

Table 1 shows the best-performing parameter values and the results with them.

For the GALME, with the type 1 of the mutation rate control, when p_0 was larger or smaller than this value, the performance was worse than this result in Table 1. With the type 2, when p_0 and β were larger than these, although the global maximum was obtained for all runs, the average of function evaluation times was larger than this result. When po and β were smaller than these, the number of convergence was smaller than this result. With the type 3,

when p_0, p_b, β_1 and β_2 were a little larger or smaller than these, although the global optimum was obtained for all runs, the average of function evaluation times was larger than this result.

For the traditional GA, the parameter values in Table 1 are those by De Jong. For the selection strategy, two individuals with the best fitness values survived intact into the next generation and other individuals of the next generation were reproduced a number of times proportional to the fitness value of that individual. When the crossover and mutation rates were larger or smaller than these, the number of convergence was smaller than this result. With the parameter values by Grefenstette, the global maximum was never obtained.

5.2 Deceptive function

I used the Goldberg's order-3 deceptive function [11] that consists of a 30-bit binary string and whose value is the sum of 8 3-bit subfunctions shown in Table 2. In the case of tightly ordered deceptive function, the bits of the subfunction are adjacent (1, 2, 3 for the first subfunction; 4, 5, 6 for the second subfunction, etc.). In the case of loosely ordered deceptive function, the first subfunction is located at positions 1, 11, and 21, the second subfunction is located at positions 2, 12, 22, etc. The search space is large: it contains 230 points. There would be 310 maxima in a ten-dimensional space because each subfunction has 3 maxima. The global maximum is 300.

Tables 3 and 4 show the best-performing parameter values and the results with them for both the deceptive functions.

For the GALME, the performance with the type 1 of the mutation rate control was the best for both the deceptive functions. When the mutation rates were a little larger or smaller than these, although the global maxima were obtained for all runs, the averages of function evaluation times were larger than these results. In the case of type 2 and 3, although the global maxima were obtained for all runs if appropriate values of the mutation rate and its reduction coefficients were used, the averages of function evaluation times were larger than those with type 1.

Table 2 Goldberg's order-3 deceptive problem

f(000)=28	f(001)=26	f(010)=22	f(011)=0
f(100)=14	f(101)=0	f(110)=0	f(111)=30

Table 3 Performance for tightly ordered deceptive function (Number of convergence to global optimum and average of function evaluation times)

Algorithm	Computation condition	Num. of converg.	Aver. of func. eval. times
GALME	N=50, Mutation=Type1 (p_0=0.095), $GMAX$=10000, Restart=No	20(all)	20895
Traditional GA	N=50, p_c=0.6, p_m=0.003, G=1.0, Scaling=No, Selection strategy=Elitist selection, $GMAX$=10000	20(all)	256207

Table 4 Performance for loosely ordered deceptive function (Number of convergence to global optimum and average of function evaluation times)

Algorithm	Computation condition	Num. of converg.	Aver. of func. eval. times
GALME	N=50, Mutation=Type1 (p_0=0.085), $GMAX$=10000, Restart=No	20(all)	68642
Traditional GA	N=50, p_c=0.6, p_m=0.0034, G=1.0, Scaling=No, Selection strategy=Elitist selection, $GMAX$=10000	20(all)	168012

For the traditional GA, the values of the parameters are the same as those by De Jong except the mutation rate. It should be noted that the optimal mutation rates are considerably larger than those by De Jong. In both the deceptive functions, when the mutation rates were smaller than these, the performances were worse than these results. With the parameter values by Grefenstte, the number of convergence out of 20 runs was 13 for the tightly ordered function and was 0 for the loosely ordered function.

5.3 summary of the results

The results for the multimodal function are summarized as follows. The GALME performed better in the case of expressing the mutation rate by a decreasing function than in the case of the constant mutation rate. With the type 2 of the mutation rate control, when the mutation rate was taken larger in the initial stage and slowly reduced, although the global optimum could be surely obtained, the speed of convergence was slow. With the type 3, when an appropriate initial value of the mutation rate was used and its reduction coefficient was taken a larger value in the first stage and a smaller value in the second stage, the global optimum could be obtained more surely and faster using restart. For the traditional GA, the number of convergence out of 20 runs was 45%.

The results for the deceptive function are summarized as follows. The GALME performed better in the case of the constant mutation rate than in the case of expressing the mutation rate by a decreasing function of generation. This result is different from that for the multimodal function and the reason needs to be investigated·in the near future. Comparing the traditional GA, the GALME required remarkably fewer function evaluations to converge for both the functions. Goldberg said that the traditional GA has difficulty to solve the deceptive function, and especially it cannot solve the loosely ordered function [11]. However these results show that it can surely solve these functions with large mutation rates using the elitist strategy.

The GALME performed better by using considerably larger mutation rates than those used in the traditional GA. Therefore these results reveal that in the GALME mutation is not only a background operator but plays an important role as search operator. It appears that the appropriate way of changing it and its appropriate value is different according to functions.

For the above two functions, it is turned out that the GALME can reach the global optimum much more surely and faster than the traditional GA. Comparing with promising previous researches, in the case of the tightly ordered function the number of function evaluations to find the global maximum is on the average 20960 for the CHC [2], 40600 for the messy GA, and on the average 20895 for the GALME. From this result, it is conjectured that the GALME may be a promising competitor to these methods.

6. Conclusion

Although the number of parameters to tune and the amount of computation of the GALME are a little larger than those of the traditional GA, the algorithm of the former is as simple as that of the latter. Additionally it is suitable for parallel processing. In the range of the above experiments, it is turned out that the performance of the former is remarkably superior to that of the latter and it is conjectured that the GALME may be a competitor to promising methods proposed in the previous researches. However further evaluation of the GALME for various problems is required before firm conclusions may be drawn. Additionally the theoretical analysis of the GALME and the theoretical proof of its convergence is required.

References

[1] Grefenstette,J.J.: Incorporating Problem Specific Knowledge into Genetic Algorithms, *Genetic Algorithms and Simulated Annealing*, pp.42-60, Morgan Kaufmann, 1987.

[2] Eshelman, L.J.: The CHC Adaptive Search Algorithm: How to Have Safe Search When Engaging in Nontraditional Genetic Recombination, *Foundation of Genetic Algorithms*, pp.265-283, Morgan Kaufmann, 1991.

[3] Syswerda, G.: Uniform Crossover in Genetic Algorithms, *Proc. of the Third International Conference on Genetic Algorithms*, pp.2-9, Morgan Kaufmann, 1989.

[4] Whitley D.: The GENITOR Algorithms and Selection Pressure: Why Rank-Based Allocation of Reproductive Trials is Best, *Proc. of the Third International Conference on Genetic Algorithms*, pp.116-121, Morgan Kaufmann, 1989.

[5] Mahfoud, S.W. et al.: A Genetic Algorithm for Parallel Simulated Annealing, *Parallel Problem Solving from Nature*, 2, pp.301-310, Elsevier Science, 1992.

[6] Sirag, D.J. et al.: Toward a Unified Thermodynamic Genetic Operator, *Proc. of the Second International Conference on Genetic Algorithms*, pp.116-122, Lawrence Erlbaum, 1987.

[7] Fogarty T.C.: Varying the Probability of Mutation in the Genetic Algorithm, *Proc. of the Third International Conference on Genetic Algorithms*, pp.104-109, Morgan Kaufmann, 1989.

[8] Bäck, T.: Self-Adaptation in Genetic Algorithms, *Proc. of the First European Conference on Artificial Life*, pp.263-271, MIT Press, 1992.

[9] Booker, L.: Improving Search in Genetic Algorithms, *Genetic Algorithms and Simulated Annealing*, pp.61-73, Morgan Kaufmann, 1987.

[10] Brown, D.E. et al.: A Parallel Genetic Heuristic for the Quadratic Assignment Problem, *Proc. of the Third International Conference on Genetic Algorithms*, pp.406-415, Morgan Kaufmann, 1989.

[11] Goldberg D.E. et al.: Messy Genetic Algorithms: Motivation, Analysis, and First Results, *Complex Systems*, Vol.3, pp.493-530, 1989.

[12] Grefenstette, J.J.: Optimization of Control Parameters for Genetic Algorithms, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol.SMC-16, No.1, pp.122-128, 1986.

[13] Aarts E. et al.: *Simulated Annealing and Boltzmann Machines*, John Wiley & Sons, 1989.

[14] Holland, J.H.: *Adaptation in Natural and Artificial Systems*, p111, MIT Press, 1992.

[15] Hesser, J. et al.: Towards an Optimal Mutation Probability for Genetic Algorithms, *Parallel Problem Solving from Nature*, pp.23-32, Springer-Verlag, 1990.

[16] Schaffer J.D. et al.: A Study of Control Parameters Affecting On-line Performance of Genetic Algorithms for Function Optimization, *Proc. of the Third International Conference on Genetic Algorithms*, pp.51-60, Morgan Kaufmann, 1989.