An Improvement of Genetic Algorithm for Optimization Problem

Sakkayaphop Pravesjit and Krittika Kantawong
Faculty of Information and Communication Technology
University of Phayao
Phayao, Thailand
sakkayaphop.pr@up.ac.th, krittika.ka@up.ac.th

***Abstract*—This paper proposed an improvement of genetic algorithm for optimization problem. In this study, the Gaussian function is applied in crossover and mutation operators instead of traditional crossover and mutation. The algorithm is tested on five benchmark problems and compared with the self-adaptive DE algorithm, traditional differential evolution (DE) algorithm, the JDE self-adaptive algorithm and the hybrid bat algorithm with natural-inspired. The computation results illustrate that the proposed algorithm can produce optimal solutions for all functions. Comparing to the other four algorithms, the proposed algorithm provides the best results. The finding proves that the algorithm should be improved in this direction.**

Keywords—optimization function; genetic algorithm; genetic operator; crossover operator; mutation operator

I. INTRODUCTION

Optimization is an attempt to get the best solution of the problem under the given circumstances. The main objective of the optimization is to increase the desired benefits or reduce the time. The processes of optimization methods are the act of trying to achieve optimal solution of the given objective function. In the literature, many algorithms have been brought to solve the problem. Nature-inspired algorithm as genetic algorithm is a capable algorithm which has been improved and published by many researchers. The genetic algorithm (GA), proposed by Holland [1], is the most popular algorithm in this field. The two important operators of genetic algorithm affect the performance of the algorithm directly. The first operator is crossover that selects two parents to combine together with a form of the option to create a new child. The other operator is mutation that improves new child randomly.

Like other algorithms, the GA sometime stuck in the local minimums. In order to improve the GA performance, some researchers improve the crossover operator e.g. Varnamkhasti et al. [2], Kaya et al. [3], and Vrajitoru [4] while other propose the new mutation operator e.g. Srinivas and Patnaik [5].

This paper proposed an improvement of genetic algorithm for optimization problem. The Gaussian function is applied in crossover and mutation operators instead of traditional operators. The aim is to improve the two operators of GA in order to solve the optimization function efficiently.

The Following contents include: section 2 describes the original genetic algorithm (GA). Section 3 presents the proposed algorithm in details. The experimental results are demonstrated in section 4. Finally, the conclusion is illustrated and discussed in section 5.

II. GENETIC ALGORITHM

Genetic algorithm (GA) is a meta-heuristic search algorithm based on the evolutionary. The main idea is derived the behavior of reproduction animal, consist of selection, crossover and mutation. Genetic algorithms represent an intelligent exploitation of a random search. It has been applied to solve optimization problem for many years. The original genetic algorithm is consists of the following component:

Step 1: generate an initialize of the populations (NP vector solutions) randomly.

Step 2: the fitness function/objective function of the populations are evaluated.

Step 3: the next generation is produced by the following steps:

(a) select two current populations, P_1 and P_2 (roulette wheel).

(b) apply one-point crossover operator to P_1 and P_2 with crossover rate (Pc) to obtain a child chromosome C_1 and C_2.

(c) apply mutation operator to C_1 and C_2 with mutation rate (Pm) to produce C_1' and C_2'.

(d) select the new generations by their fitness ranking.

Step 4: replace the source population with the new generations.

Step 5: if stopping criteria is not met, return to Step 2.

III. THE PROPOSED ALGORITHM

This paper aims to improve the operators of genetic algorithm (crossover and mutation) with Gaussian function for continuous optimization problem. After generates population and evaluates their fitness values, the improved procedure of crossover and mutation will be applied. The processes are explained in details as follows:

(i) The selection step, using roulette wheel for random select two current individuals, P_1 and P_2.

978-1-5090-5210-3/17/$31.00 ©2017 IEEE

(ii) The crossover operation is performed in this step. The two-point crossover is applied for P_1 and P2. The Gaussian function, formula (2) and (3) are applied to calculate the value in the genes of the parents. After that, formula (1) is applied to select the best value and replace it in the considering parent to produce new child as demonstrated in Fig. 1.

$$G_j^{cross} = \begin{cases} G_j^F & ;if\ f\left(G_j^F\right) < f\left(G_j^M\right) < f\left(G_{avg}\right) \\ G_j^M & ;if\ f\left(G_j^M\right) < f\left(G_j^F\right) < f\left(G_{avg}\right) \\ G_{avg} & ;if\ f\left(G_{avg}\right) < f\left(G_j^F\right) < f\left(G_j^M\right) \end{cases} \quad (1)$$

$$G_j^F = \frac{1}{\sqrt{2\pi\sigma_{P1}^2}} e^{-\left[\frac{\left(x_j^{P1}\right)^2}{2\sigma_{P1}^2}\right]} \quad (2)$$

$$G_j^M = \frac{1}{\sqrt{2\pi\sigma_{P2}^2}} e^{-\left[\frac{\left(x_j^{P2}\right)^2}{2\sigma_{P2}^2}\right]} \quad (3)$$

$$G_{avg} = \left(x_j^{P1} + x_j^{P2}\right)/2 \quad (4)$$

where $j = 1, 2, 3, \ldots, D$, D is the dimension of population vector.

G_j^{cross} the results of value in the crossover conditions.

G_j^{P1} the value of genes (P1) was calculated by Gaussian function.

G_j^{P2} the value of genes (P2) was calculated by Gaussian function.

x_j^{P1} the value of gene in order j from chromosome P1.

x_j^{P2} the value of gene in order j from chromosome P2.

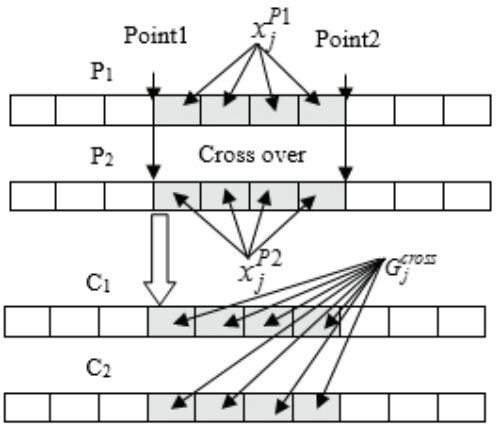

Fig. 1. The Crossover Operation (crossover rate = 0.7).

(iii) After the crossover step, mutation operation is working on all children by equations:

$$x_j^M = \begin{cases} x_j^G & ;if\ f\left(x_j^G\right) < f\left(x_j^{R1}\right) < f\left(x_j^{R2}\right) \\ x_j^{R1} & ;if\ f\left(x_j^{R1}\right) < f\left(x_j^G\right) < f\left(x_j^{R2}\right) \\ x_j^{R2} & ;if\ f\left(x_j^{R2}\right) < f\left(x_j^G\right) < f\left(x_j^{R1}\right) \end{cases} \quad (5)$$

$$x_j^G = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{-\left[\frac{\left(x_j^c\right)^2}{2\sigma_c^2}\right]} \quad (6)$$

$$x_j^{R1} = x_j^c + rand(-0.05, 0.05) \quad (7)$$

$$x_j^{R2} = x_j^c * F \quad (8)$$

where

x_j^M : the results of value in the mutation conditions.

x_j^c : the value of genes (C) was calculated by Gaussian function.

F : rand [0,1] value

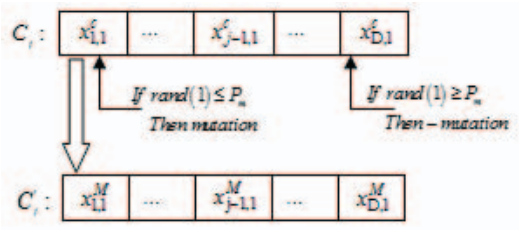

Fig. 2. Mutation Operation (permutation rate 0.3)

All initial population and children from the two operators will be evaluated their fitness value. Then, the new generations are selected by their ranking value. The new population will be improved continuously until the stopping condition is met.

IV. Expermental results

The proposed algorithm is tested on the different 5 benchmark functions from Yao et al [6]. The tested functions are presented in Table 1. Functions 1 to 3 are the continuous functions, function 4 is the discontinuous function, and function 5 is the multimodal function.

The computational results in Table 2 are shown in comparison with other four algorithms: (a) traditional DE algorithm and the JDE self-adaptive algorithm are taken from Brest et al [7] (b) the self-adaptive differential evolution algorithm is taken from Jitkongchuen and Thammano [8] and (c) the hybrid bat algorithm with natural-inspired algorithms from Pravesjit [9]. It indicates that the proposed algorithm produces better results than the other four algorithms. More significantly, it claims that the proposed algorithm provide the optimal solution for all tested functions.

V. Conclusions

This paper proposed an improvement of genetic algorithm for optimization function. In order to improve the performance of genetic algorithm, the Gaussian function is applied to calculate gene values in crossover and mutation operators. The testing on five various functions shows that the proposed algorithm provides the optimal value for all of them. The experimental results confirm that the proposed algorithm can solve the continuous functions, the discontinuous function, and the multimodal function efficiently.

Acknowledgment

The authors would like to acknowledge School of Information and Communication Technology, University of Phayao, Thailand for all resources and financial support.

References

[1] J. H. Holland, Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence, U Michigan Press, 1975.

[2] M. J. Varnamkhasti, L. S. Lee, M. R. A. Bakar, and W. J. Leong, "A genetic algorithm with fuzzy crossover operator and probability," *Advances in Operations Research*, vol. 2012, [Online].Available: http://dx.doi.org/10.1155/2012/956498

[3] Y. Kaya, M. Uyar, and R. Tek, "A novel crossover operator for genetic algorithms: Ring crossover," arXiv preprint arXiv: 1105.0355, 2011.

[4] D. Vrajitoru, "Crossover improvement for the genetic algorithm in information retrieval," Information Processing & Management, vol. 34, no. 4, pp. 405–415, 1998.

[5] M. Srinivas and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," IEEE Transactions on Systems, Man and Cybernetics, vol. 24, no. 4, pp. 656–667, 1994.

[6] Yao X, Liu Y and Lin G (1999), Evolutionary programming made faster, IEEE Transactions on Evolutionary Computation, Vol. 3, No. 2, pp. 82-102.

[7] Brest J, Greiner S, Boskovic B, Mernik M, and Zumer V (2006), Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems, IEEE Transactions on Evolutionary Computation, Vol. 10, No. 6, pp. 646–657.

[8] Duangjai Jitkongchuen and Arit Thammano, (2014), A self-adaptive differential evolution algorithm for continuous optimization problems, Artificial Life and Robotics, 19, pp.201-208.

[9] Sakkayaphop Pravesjit (2016), A hybrid bat algorithm with natural-inspired algorithms for continuous optimization problem, Artificial Life and Robotics, 21, pp.112-119.

TABLE I. THE BENCHMARK FUNCTIONS

Functions	Iteration	Dimension (D)	Search space	fmin
$f_1(x)=\sum_{i=1}^{D} x_i^2$	1500	30	[-100, 100]	0
$f_2(x)=\sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	2000	30	[-10, 10]	0
$f_3(x)=\sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2$	5000	30	[-100, 100]	0
$f_4(x)=\max_i \left\{ \lvert x_i \rvert, 1 \le i \le D \right\}$	1500	30	[-100, 100]	0
$f_{10}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{D} x_i^2} \right) - \exp\left(\frac{1}{30}\sum_{i=1}^{D} \cos 2\pi x_i \right) + 20 + e$	1500	30	[-32, 32]	0

TABLE II. THE EXPERIMENTAL RESULTS

Functions	DE	JDE	EVO-DE	Hybrid-Bat	Proposed
f1	8.2E-14	1.1E-28	0	0	0
f2	1.5E-9	1.0-23	0	0	0
f3	0	0	0	0	0
f4	0	0	0	0	0
f5	1.5017	7.7E-15	4.44E-16	6.59E-26	0