퍼지 및 지능시스템학회 논문지 2007, Vol. 17, No. 3, pp. 417-422

Queen-bee and Mutant-bee Evolution for Genetic Algorithms

Sung Hoon Jung[1]

Department of Information and Communication Engineering, Hansung University, Seoul 136-792, Korea

Abstract

A new evolution method termed queen-bee and mutant-bee evolution is based on the previous queen-bee evolution [1]. Even though the queen-bee evolution has shown very good performances, two parameters for strong mutation are added to the genetic algorithms. This makes the application of genetic algorithms with queen-bee evolution difficult because the values of the two parameters are empirically decided by a trial-and-error method without a systematic method. The queen-bee and mutant-bee evolution has no this problem because it does not need additional parameters for strong mutation. Experimental results with typical problems showed that the queen-bee and mutant-bee evolution produced nearly similar results to the best ones of queen-bee evolution even though it didn't need to select proper values of additional parameters.

Key words : genetic algorithms, optimization, premature convergence

1. Introduction

The exploration and exploitation in genetic algorithms are very important factors for improving the performances of genetic algorithms [2, 3, 4, 5, 6, 7]. From this point of view, we have introduced queen-bee evolution [1] for fast evolution of individuals by employing strong exploitation and strong exploration (in genetic algorithms, the crossover operation is for exploitation and the mutation operation is for exploration). This evolution has improved the performances of genetic algorithms about 200 times to 1,000 times than the simple genetic algorithms. However, it needs two additional parameters for strong mutation in order to control the exploration. This is a very critical drawback of the method because the two parameters ranged from 0 to 1 greatly affects the performances of genetic algorithms and there is no systematic method to decide proper values. Currently, only trial-and-error method, which is a very ineffective and time-consuming task, can be used for selecting proper values.

In this paper, we propose a new evolution method called queen-bee and mutant-bee evolution in order to overcome this problem. The proposed method does not need the two parameters for strong mutation no more. In the queen-bee and mutant-bee evolution, we adopt mutant-bees, the strongly mutated individuals. These mutant-bees for strong exploration are recombined with the queen-bees for strong exploitation to generate offsprings. In the previous queen-bee evolution, the queen-bee, the fittest individual, is recombined with the normal individuals and the offsprings are strongly mutated by the two parameters, strong mutation rate and strong mutation probability. However, the normal individuals in the queen-bee and mutant-bee evolution are first strongly mutated without additional parameters and then the strongly mutated individuals are recombined with the queen-bee.

Since the recombined individuals are strongly mutated in the queen-bee evolution, the good schema of recombined individuals can be destroyed. On the other hands, the good schema of queen-bee may be inherited in the queen-bee and mutant-bee evolution because selected individuals are first strongly mutated and then recombined with the queen-bee. The selected individuals are strongly mutated by the inversion of half most significant strings of them without the other parameters. The proposed method has been tested with typical four optimization problems that have been used previous papers [8, 9, 1]. It was shown from the experiments that the performances of the queen-bee and mutant-bee evolution were very similar to the best performances of the queen-bee evolution.

This paper is organized as follows. Section 2 describes the proposed queen-bee and mutant-bee evolution. In section 3, experimental parameters and results are discussed. This paper concludes in section 4.

2. Queen-bee and Mutant-bee Evolution

In order to certainly compare the previous queen-bee evolution and the proposed queen-bee and mutant-bee evolution, we describe them in Algorithm 1 and in Algorithm 2, respectively. The newly added or modified operations to the simple genetic algorithm are marked by ▲ for the queen-bee evolution and ■ for the queen-bee and mutant-bee evolution, respectively. In both evolution methods, from the first to the selection of parents to generate offsprings are same as shown in the Algorithms 1 and 2. Parents are composed of the pair of queen-bee $I_q(t-1)$ and selected individuals $I_m(t-1)$ by a selection method such as roulette wheel selection and rank selection. In queen-bee evolution, those parents are recombined and then strongly mutated with the strong mutation rate ξ and strong mutation probability p'_m. On the other hand, in queen-bee and mutant-bee evolution the individuals $I_m(t-1)$ are strongly mutated by inverting their half most significant bits of strings before recombination. For example, the 8 bits of an individual, 01100111, is inverted to 10010111. The strongly mutated individuals $I_m(t-1)^*$ (strong exploration) are recombined with the queen-bee (strong exploitation) to generate offsprings.

Algorithm 1 Queen-bee evolution
```
   // $t$ : time //
   // $n$ : population size //
   // $P$ : populations //
   // $\xi$ : normal mutation rate //
   // $p_m$ : normal mutation probability //
   // $p'_m$ : strong mutation probability //
   // $I_q$ : a queen-bee individual//
   // $I_m$ : normal individuals //
1  t ← 0
2  initialize $P(t)$
3  evaluate $P(t)$
4  while (not termination-condition)
5  do
6      t ← t + 1
7      select $P(t)$ from $P(t-1)$ (▲)
8          $P(t) = \{(I_q(t-1), I_m(t-1))\}$
9      recombine $P(t)$
10         do crossover
11         do mutation (▲)
12             for $i = 1$ to $n$
13                 if $i \le (\xi \times n)$
14                     do mutation with $p_m$
15                 else
16                     do mutation with $p'_m$
17                 end if
18             end for
19     evaluate $P(t)$
20 end
```

Algorithm 2 Queen-bee and Mutant-bee evolution
```
   // $t$ : time //
   // $n$ : population size //
   // $P$ : populations //
   // $I_q$ : a queen-bee individual//
   // $I_m$ : normal individuals //
   // $I^*_m$ : inverted individuals (mutant-bees) //
1  t ← 0
2  initialize $P(t)$
3  evaluate $P(t)$
4  while (not termination-condition)
5  do
6      t ← t + 1
7      select $P(t)$ from $P(t-1)$ (■)
8          $P(t) = \{(I_q(t-1), I_m(t-1))\}$
9      make mutant-bees (■)
10         invert $I_m(t-1)$ to $I_m(t-1)^*$
11         set $P(t) = \{I_q(t-1), I_m(t-1)^*\}$
12     recombine $P(t)$
13         do crossover
14         do mutation
15     evaluate $P(t)$
16 end
```

Intuitively, the queen-bee and mutant-bee evolution is better than the previous queen-bee evolution in that it relatively keeps the good schema of good individuals in the previous generation. In the queen-bee evolution, the good schema of recombined individuals may be destroyed by strong mutation because the strong mutation is done after crossover operation. Whereas the mutant-bees are recombined with the queen-bees in the queen-bee and mutant-bee evolution, the good schema of queen-bees can be relatively inherited to the offsprings. This is an advantage of proposed evolution method.

Moreover, the queen-bee and mutant-bee evolution does not need additional parameters such as ξ and p'_m which are empirically selected by a trial-and-error method without systematic decision methods. This is main advantage of the proposed method compared to the previous method. Even though the queen-bee and mutant-bee evolution is very simple and has no additional parameters, it shows nearly same performances to the best results of queen-bee evolution as shown in next section.

3. Experimental Results

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

$$f_2 = 100(x_1^2 - x_2)^2 + (1 - x_1)^2, \text{ where } -2.048 \leq x_i \leq 2.048 \tag{2}$$

$$f_3 = 0.5 - \frac{sin(\sqrt{x_1^2 + x_2^2})sin(\sqrt{x_1^2 + x_2^2}) - 0.5}{(1.0 + 0.001(x_1^2 + x_2^2))(1.0 + 0.001(x_1^2 + x_2^2))}, \text{ where } -10 \leq x_i \leq -10 \tag{3}$$

$$f_4 = (x_1^2 + x_2^2)^{0.25} sin(50(x_1^2 + x_2^2)^{0.1} + 1)^2, \text{ where } -10 \leq x_i \leq -10 \tag{4}$$

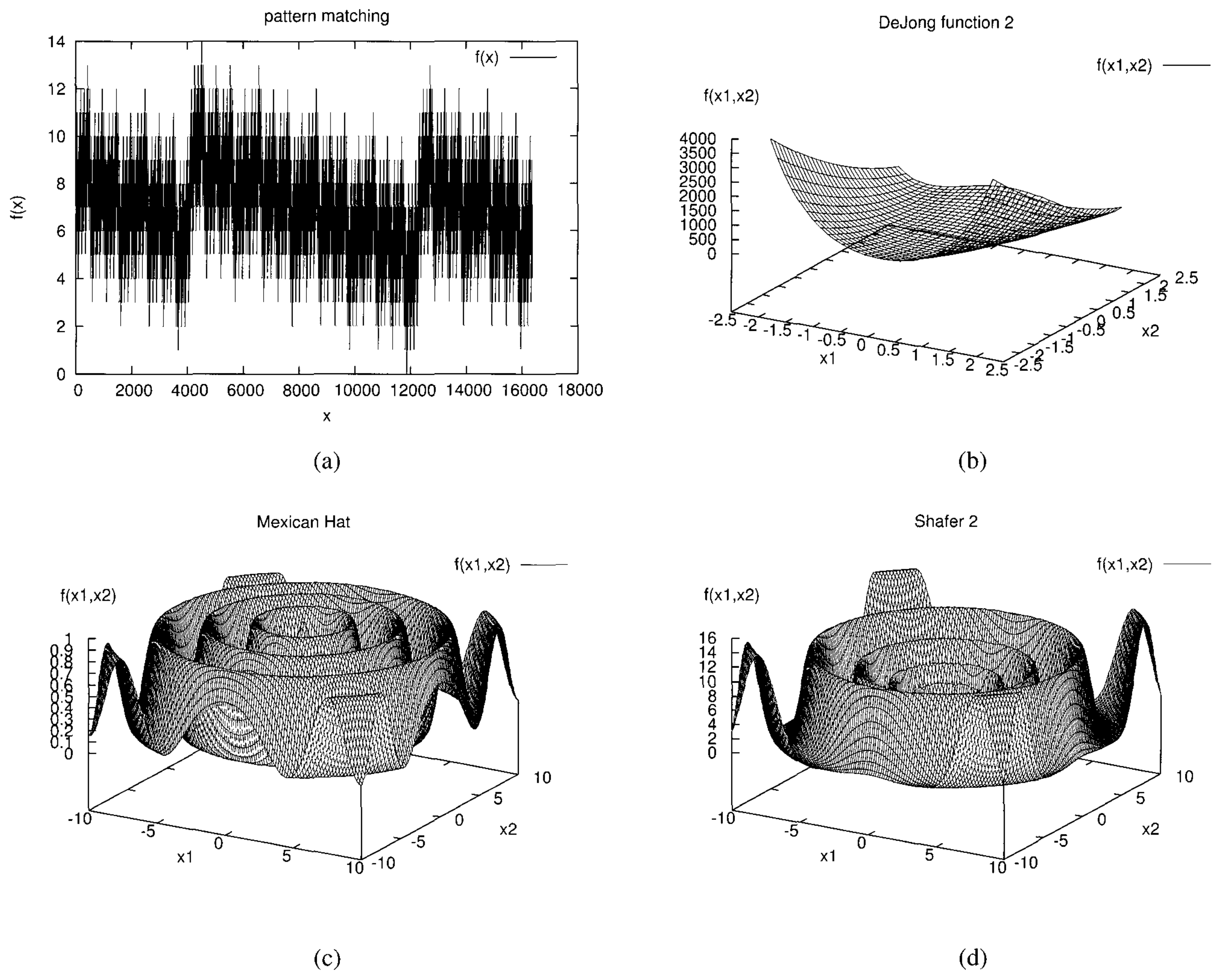

Figure 1: Experimental functions (a) f_1 (b) f_2 (c) f_3 (d) f_4)

Table 1: Parameters for experiments

Parameters	Values
Selection method	roulette wheel selection
Crossover probability (p_c)	0.6
Mutation probability (p_m)	0.05
Population size	10
Individual length	24 bits

The proposed method was tested on typical four function optimization problems that have been used at previous papers [8, 9, 1]. All functions except for f_1 are function optimization problems. On the other hand, the function f_1, a bit pattern matching problem, is a combinatorial optimization problem between the target pattern T and an individual's pattern I. If all bits between T and I are same, then it is optimum. Therefore, optimum fitness is same as the number of bits h. Functions f_2 to f_4 are DeJong function 2 (f_2), Mexican hat function (f_3), Shafer function 2 (f_4), respectively.

Figure 1 shows the input-output relations of four functions. Function f_2 is relative simple in that it has a few local optimum; unlike function f_2, functions f_3 and f_4 have many local optimum in continuous spaces. The optimum value of f_2 is only one at $x_1 = -2.048$ and $x_2 = -2.048$ and optimum value is about 3905.9 and local optimum is at $x_1 = 2.048$ and $x_2 = -2.048$. The function f_3 has only one global optimum (its value is about 0.99) at the center of the smallest circle of Mexican hat. But the f_3 has many local optima around the smallest circle of Mexican hat. It is very difficult for genetic algorithms to come out the local optimum because the values of local optima are very close to that of global optimum. On the other hands, function f_4 has multiple optima at four peaks near ($x_1 = -10$ and $x_2 = -10$, $x_1 = -10$ and $x_2 = 10$, $x_1 = 10$ and $x_2 = -10$, $x_1 = 10$ and $x_2 = 10$) and its value is about 14.3. Similar to the f_3, f_4 has many local optima inside the four peaks.

The parameters of genetic algorithms for experiments are shown in Table 1. We used typical parameters that have been used at a lot of previous works. Experimental results are shown in Table 2. All results in the Table 2 are average values of 10 runs with different random number seeds. In Table 2, avg., dev., NE, QBE, and QBMB mean average values of 10 runs, standard deviation of 10 runs, normal evolution used in the simple genetic algorithm, the queen-bee evolution [1], and the queen-bee and mutant-bee evolution, respectively Note that the QBMB outperforms NE of original genetic algorithm in all functions and shows similar performances to the best of QBE. Since selected individuals as parents in queen-bee evolution are first recombined and then strongly mutated, the offsprings are mainly affected by the strong mutation. This makes the queen-bee evolution dependent on the strong mutation rate and strong mutation probability. However, the queen-bee and mutant-bee evolution shows relatively stable and good performances because its strong mutation method is more simple than the queen-bee evolution and generated offsprings are not affected by the strong mutation unlike the queen-bee evolution. This makes it possible for QBMB to keep the good schema of the queen-bee, whereas the good schema of the queen-bee can be destroyed at the QBE.

4. Conclusion

In this paper, we proposed a new evolution method, queen-bee and mutant-bee evolution. It was found from experiments that the proposed evolution showed very similar results to the best ones of the queen-bee evolution even though the proposed evolution does not need selecting of proper values of additional two parameters by a trial-and-error method. This indicates that the proposed method can be largely applied to existing genetic algorithms for improving their performances with simple modification and without additional efforts.

Acknowledgement. This research was financially supported by Hansung University in the year of 2007.

References

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

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

Table 2: Experimental results

			f_1		f_2	
NE			64201.3	40495.1	59078.5	55933.1
	ξ	p_m'	avg.	dev.	avg.	dev.
QBE[1]	0.4	0.6	40039.6	30811.1	50147.8	60182.2
		0.8	56686.6	40370.3	24768.4	15588.8
		1.0	6933.1	5944.4	10347.2	8666.2
	0.6	0.6	1115.3	695.8	1560.8	1486.0
		0.8	654.9	511.8	2354.5	2428.4
		1.0	351.0	392.3	962.6	893.7
	0.8	0.6	84.2	51.6	732.4	657.4
		0.8	97.5	59.6	349.2	259.6
		1.0	77.5	53.2	(*) 298.0	199.8
	1.0		(*) 58.4	19.9	1840432.7	2234964.3
	best		58.4	19.9	298.0	199.8
QBMB			89.7	80.7	204.8	172.6

			f_3		f_4	
NE			776085.3	624688.8	250111.6	228965.8
	ξ	p_m'	avg.	dev.	avg.	dev.
QBE[1]	0.4	0.6	944407.3	627457.7	197116.1	166948.2
		0.8	613873.5	534078.5	226828.3	251973.4
		1.0	7701.1	7366.8	38822.7	25534.8
	0.6	0.6	355116.1	309387.3	175004.1	162650.8
		0.8	295721.0	261750.9	130466.9	101402.1
		1.0	4170.3	3277.9	16472.4	14200.6
	0.8	0.6	72121.8	70986.7	26486.6	19529.5
		0.8	37867.0	26954.4	37916.2	49822.8
		1.0	(*) 3767.3	2191.4	(*) 18000.0	14679.7
	1.0		17779.8	16727.3	29603.1	17851.4
	best		3767.3	2191.4	18000.0	14679.7
QBMB			6630.1	7145.5	19193.2	22130.5

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

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

[5] J. A. Vasconcelos, J. A. Ramirez, R. H. C. Takahashi, and R. R. Saldanha, "Improvements in Genetic Algorithms," *IEEE Transactions on Magnetics*, vol. 37, pp. 3414–3417, Sept. 2001.

[6] E. Alba and B. Dorronsoro, "The exploration/exploitation tradeoff in dynamic cellular genetic algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 9, pp. 126–142, Apr. 2005.

[7] V. K. Koumousis and C. Katsaras, "A saw-tooth genetic algorithm combining the effects of variable population size and reinitialization to enhance performance," *IEEE Transactions on Evolutionary Computation*, vol. 10, pp. 19–28, Feb. 2006.

[8] K. DeJong, *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, University of Michigan, 1975.

[9] J. Andre, P. Siarry, and T. Dognon, "An improvement of the standard genetic algorithm fighting premature convergence in continuous optimization," *Advances in engineering software*, vol. 32, no. 1, pp. 49–60, 2001.

저 자 소 개

Sung Hoon Jung

1991년 - 1995년: 한국과학기술원 전기및전자공학과 (공학박사)
1995년 - 1996년: 한국과학기술원 전기및전자공학과 위촉연구원
1996년 - 현재: 한성대학교 정보통신공학과 교수
관심분야: 진화연산, 신경망, 퍼지, 시스템생물학, 생물지능
E-mail : shjung@hansung.ac.kr