A Novel Optimization Approach: Bacterial-GA Foraging

Tai-Chen Chen, Pei-Wei Tsai, Shu-Chuan Chu*, and Jeng-Shyang Pan
National Kaohsiung University of Applied Sciences, Taiwan
*Cheng Shiu University, Taiwan**
pwtsai@bit.kuas.edu.tw, scchu@csu.edu.tw, jspan@cc.kuas.edu.tw

Abstract

In this paper, we proposed a novel optimization model, which combines Bacterial Foraging with Genetic Algorithm. Though these two well-known optimization algorithms have their own good points, they also have their own drawbacks respectively. In our work, a combined evolutional model, Bacterial-GA Foraging, is proposed. Via applying this new model, experimental results indicate that the new combined model performs much better performance than applying any of these two algorithms singly.

1. Introduction

Both Bacterial Foraging (BF) [1,4-5] and Genetic Algorithm (GA) [2,3] are well-known popular algorithms for solving optimization problems. BF simulated the life cycles and the foraging behaviors of bacterial, which are called E. coli, to solve the optimization problems, and GA uses chromosomes to represent the solution sets in the algorithm. However, these algorithms still have their own weaknesses on finding the solutions due to the characteristics of their own evolutional models.

In 2000, Kevin M. Passino proposed the idea of BF [1] for solving optimization problems. The framework of BF was based on parroting the behaviors of bacterium, i.e., the way they search nutrients, evade noxious environments, and the moving circumstance. By way of imitating the existence of the bacterium, the optimization problem can be solved.

When analyzing an optimization algorithm, the searching ability can be roughly separated into two types, namely, the global search ability and the local search ability. Through observing the processes of BF and GA, we noticed that BF emphasizes the local search ability, but GA performs high searching ability on global search. In addition, the solution sets of these two algorithms function in different ways. The solution set in BF is a single bacterium, and it basically finds optimal solutions on its own. On the contrary, the solution set in GA is a chromosome, which carries the solution as genes, but it searches the optimal solutions by exchanging the information with others. In other words, the bacterial in BF work along, but the chromosomes in GA somehow whack the solutions they have. Eventually, by the phenomena we inspected, the idea of combining BF with GA was generated.

2. Bacterial Foraging

Assume that we are going to find the minimum of a fitness function. In equation (1), θ is a coordinate in the solution space, and $J(\theta)$ is an attractant-repellant profile, which represents where nutrients and noxious substances are located.

$$J(\theta),\ \theta \in \Re^{p} \tag{1}$$

$$P(j,k,l) = \{\theta^{i}(j,k,l) | i = 1,2,...,S\} \tag{2}$$

Equation (2) denotes the positions of each bacterium in the population. For the i_{th} bacteria, it represents the bacteria at the j_{th} chemo-tactic step, k_{th} reproduction step, and the l_{th} elimination-dispersal event.

When evaluating the fitness value of each bacterial, the users can choose whether to consider the cell-to-cell signaling via attractant and repellent coefficients. And these coefficients are calculated by equation (3), and then combined with the fitness value in equation (4), where J_{cc} represents the combined cell-to-cell attraction and repelling effects, and $\theta = [\theta_1, \theta_2, ..., \theta_P]^T$. The *d*, *h* and w are all the attractant and repellent coefficients, which we mentioned above.

The movement of each bacterium in BF is applied after the evaluation process. Every bacterial in the population will tumble once, and the bacterial, which have better fitness values after the tumble, will go on the swim step under the swim length limit N_s. The movement can be described as equation (5), and $C(i)$ denotes the step size, and $\phi(j)$ is a random variable in $[0,1]$. $C(i)\phi(j)$ is the moving length that the i_{th} bacterium takes.

The evaluation and movement processes take turns to execute until the lifetime of the bacteria reach the limit, and then the reproduction process executes. In the process of reproduction, the whole population will only keep half of the bacteria, which perform better healthy values, and directly reproduce them to replace the bacteria with worse healthy values. Once the reproduction is done, the process goes back into the cycles of evaluation and movement for the reproduced bacteria.

When the reproduction times exceed the limit which user defined, the elimination-dispersal even takes part in the process. This process eliminates the bacteria with probability, if a bacterium is eliminated, a bacterium, which fits in with the initial conditions, is generated to replace the eliminated one. Then the whole processes described above repeat until all the process times are exceeded.

$$
\begin{aligned}
J_{cc}(\theta, P(j,k,l)) &= \sum_{m=1}^{S} J_{cc}^{i}(\theta, \theta^{i}(j,k,l)) \\
&= \sum_{m=1}^{S}\left[-d_{attract}\cdot e^{\left(-w_{attract}\sum_{m=1}^{p}(\theta_m-\theta_m^i)^2\right)}\right] \\
&+ \sum_{m=1}^{S}\left[-h_{repellant}\cdot e^{\left(-w_{repellant}\sum_{m=1}^{p}(\theta_m-\theta_m^i)^2\right)}\right]
\end{aligned}
\tag{3}
$$

$$J(\theta) = J(i,j,k,l) + J_{cc}(\theta, P) \tag{4}$$

$$\theta^{i}(j+1,k,l) = \theta^{i}(j,k,l) + C(i)\phi(j) \tag{5}$$

3. Genetic Algorithm

GA is a well known algorithm for solving optimization problems. According to Darwin’s Theory, it takes the natural evolution’s model to find the optimal solution to the evaluative function. In other words, the GA-type algorithms search based on technique of natural selection and natural genetics. Basically, GA only defines the rules for users to solve their own optimization problems, not a fixed function, which means that the authors can easily apply GA into different applicative fields.

To solve optimization problems by GA, like other optimization algorithms, the first step is to initialize the solution sets. In GA, we use chromosomes to represent the solutions in the solution space, and genes to indicate the value on all specific dimensions.

Here we list the operation steps of GA below:

1. Initiation: Generate a population of chromosomes, and randomly give them values for genes, which mean to put them into the solution space.
2. Evaluation: In this step, all the chromosomes get a fitness value according to the calculations from the user defined fitness function. The fitness function exactly fits the problem of the solution space. And the chromosome, which has the best fitness value, will be kept as the best solution set.
3. Selection: According to the selection rate, part of the chromosomes will be eliminated, and the other part of them will be kept. There are lots of methods to select some chromosomes to keep. And the most popular ways are random selection, roulette wheel selection, and essential selection.
4. Crossover: This step forces the chromosomes to exchange the information they carried to the others via switching the information on parts of the dimensions. The popular crossover methods are one-site crossover, and two-site crossover.
5. Mutation: According to the mutation rate, several chromosomes will be selected from the population, and then randomly change the value of parts of the dimensions. This will give the population a larger chance to generate new species. For optimization, it is a chance to get an abrupt evolution.
6. Termination Checking: GA repeats Step 2 to Step 5 until certain termination is met, such as pre-defined number of generation is reached, or it failed to have process for certain number of generations. Once terminated, GA reports the best chromosome it has during the course.

4. Bacterial-GA Foraging

According to the experiments, we notice that BF has higher local search ability than GA, but it is poor in global search while the solution space is huge. On the contrary, GA performs high capacity on global search, but it is difficult for GA to pinpoint the global best solution. Our idea is based on combining these two methods, and takes both the strengths of the both methods to make up the drawbacks they have. Therefore, we let the bacteria carry their own chromosomes, which mean that when the evolution goes on, both the bacteria’s positions and their genes on the chromosomes will change according to the courses of both BF and GA. The flowchart of Bacterial-GA Foraging is shown in Figure 1.

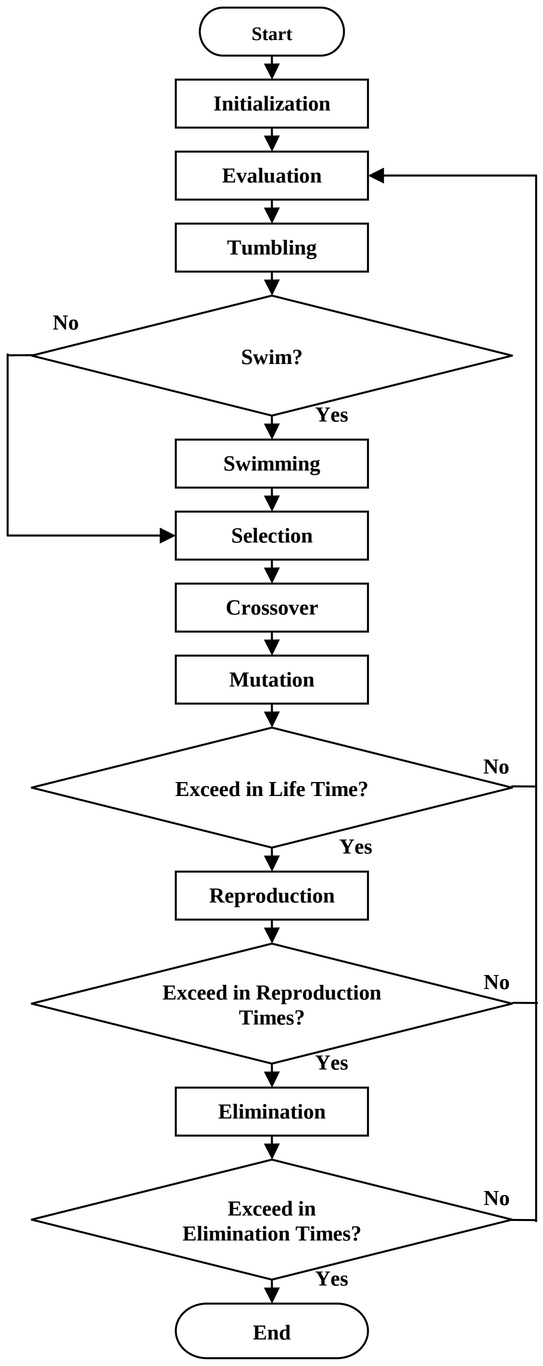

Figure 1. The flowchart of Bacterial-GA Foraging

As shown in Figure 1, the courses of tumbling, swimming, reproduction are exactly the same as how they work in BF, and the courses of crossover and mutation are the same as they work in GA. The evaluation course here we used takes as the model of BF without considering the cell-to-cell attraction and repelling effects.

However, the selection process in Bacterial-GA Foraging selects the individuals according to their swim lengths, which were summed up in the swimming process, not the fitness values. Furthermore, the elimination course is also a bit different from the process in BF. In Bacterial-GA Foraging, when an individual is eliminated, the course does not generate another one via the initialization process, but it generates a new individual via mutating all the dimensions from the eliminated one.

By way of experiments, we find that Bacterial-GA Foraging performs better searching results, and the convergence is also good.

5. Experiments and Conclusions

During the experiments, we applied BF, GA and Bacterial-GA Foraging into four test functions for examining the searching abilities of these algorithms.

The test functions are listed as follows. And the initial constrain of each dimension of the solutions are listed in Table 1. The parameter setting for BF and GA is listed in Table 2. Since Bacterial-GA Foraging follows all the parameter setting in BF and GA, we do not need to list them additionally.

In addition, when applying BF in the experiments, we consider about the cell-to-cell attraction and repelling effects.

$$\begin{aligned} f_1(x) = {} & 5\cdot e^{-0.1\left[(x_0-15)^2+(x_1-20)^2\right]} - 2\cdot e^{-0.1\left[(x_0-20)^2+(x_1-15)^2\right]} \\ & + 3\cdot e^{-0.1\left[(x_0-25)^2+(x_1-10)^2\right]} + 2\cdot e^{-0.1\left[(x_0-10)^2+(x_1-10)^2\right]} \\ & - 2\cdot e^{-0.1\left[(x_0-5)^2+(x_1-10)^2\right]} - 4\cdot e^{-0.1\left[(x_0-15)^2+(x_1-5)^2\right]} \\ & - 2\cdot e^{-0.1\left[(x_0-8)^2+(x_1-25)^2\right]} - 2\cdot e^{-0.1\left[(x_0-21)^2+(x_1-25)^2\right]} \\ & + 2\cdot e^{-0.1\left[(x_0-25)^2+(x_1-16)^2\right]} + 2\cdot e^{-0.1\left[(x_0-5)^2+(x_1-14)^2\right]} \end{aligned} \tag{6}$$

$$f_2(x) = \sum_{i=1}^{n-1}\left[100\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2\right] \tag{7}$$

$$f_3(x) = \sum_{i=1}^{n}\left[x_i^2 - 10\cos\left(2\pi \cdot x_i\right)^2 + 10\right] \tag{8}$$

$$f_4(x) = \frac{1}{400}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \tag{9}$$

The population sizes and the dimension of solution spaces for BF, GA and Bacterial-GA Foraging are all the same. For the first test function, the population size is set to 50, and the dimension of solution space is set to 2; for the rest of the test functions, the population sizes are set to 160, and the dimension of solution spaces are set to 30.For Bacterial-GA Foraging, we still use essential selection and one-site crossover to handle the related courses.

Table 1. The Limitation ranges for test functions

Function Name	Limitation Range
Test Function 1	$x_i \in [-15, 15]$
Test Function 2	$x_i \in [15, 30]$
Test Function 3	$x_i \in [2.56, 5.12]$
Test Function 4	$x_i \in [300, 600]$

Table 2. Parameter settings for BF and GA

BF		GA	
Parameter	**Value**	**Parameter**	**Value**
Elimination Times	2	Selection Rate	0.5
Reproduction Times	4	Mutation Rate	0.08
Max Swim Length	5	Selection Method	Essential Selection
Elimination Rate	0.25	Crossover Type	One-site Crossover
$d_{attract}$ / $h_{repellent}$	0.1		
$\omega_{attract}$	0.2		
$\omega_{repellent}$	10		

Here we list the experimental results by the order of test functions. According to the experimental results, Bacterial-GA Foraging finds the best solution of the three algorithms. It performs high stable searching ability and good convergence. And the elimination course is quite helpful on extending the searching ability.

6. References

[1] Kevin M. Passino, “Biomimicry of Bacterial Foraging for Distributed Optimization and Control”, *Control Systems Magazine*, IEEE, June 2002, pp. 52-67.

[2] J. S. Pan, F. R. McInnes and M. A. Jack, “Application of Parallel Genetic Algorithm and Property of Multiple Global Optima to VQ Codevector Index Assignment”, *Electronics Letters, Vol. 32, No. 4*, 1996, pp.296-297.

[3] KwanWoo Kim, Mitsuo Gen and Myoung Hun Kim, “Adaptive Genetic Algorithms for Multi-resource Constrained Project Scheduling Problem with Multiple Modes”, *Information and Control, Vol. 2, No 1*, International Journal of Innovative Computing, 2006.

[4] S. Mishra, and C. N. Bhende, “Bacterial Foraging Technique-Based Optimized Active Power Filter for Load Compensation”, *IEEE TRANSACTIONS ON POWER DELIVERY VOL. 22, NO. 1*, IEEE, January 2007, pp. 457-465.

[5] W. J. Tang, Q. H. Wu, and J. R. Saunders, “Bacterial Foraging Algorithm For Dynamic Environments”, *IEEE Congress on Evolutionary Computation*, IEEE, July 2006, pp. 1324-1330.

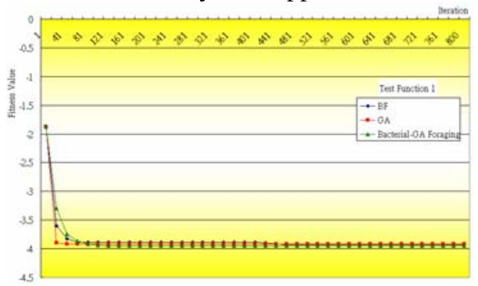

Figure 2. The experimental result of test function 1

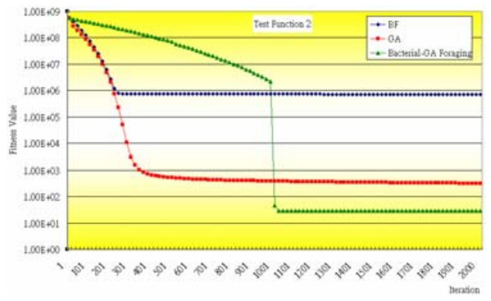

Figure 3. The experimental result of test function 2

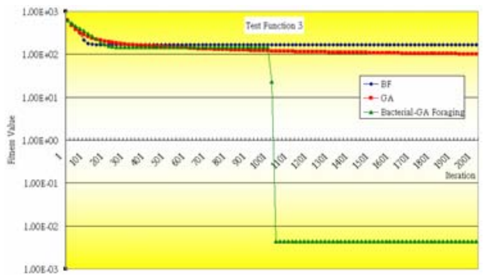

Figure 4. The experimental result of test function 3

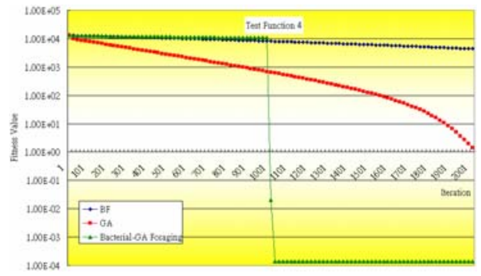

Figure 5. The experimental result of test function 4