Target Searching for Multiple Robots Using Hybrid Particle Swarm and Bacterial Foraging Optimization

Qian Zhang¹,*, Xu Wu²,a and Xiaoqian Qi¹,b

¹School of Control and Mechanical Engineering, Tianjin Chengjian University, Tianjin, China
²School of Art and Design, Tianjin University of Technology, Tianjin, China

*Corresponding author e-mail: zhangqian12399@163.com, a739775737@qq.com, bxiaoqian_seeety@126.com

Abstract. Particle Swarm Optimization (PSO) and Bacterial Foraging Optimization (BFO) are both important algorithms in target searching tasks. A hybrid optimization algorithm with the advantages of PSO and BFO is proposed to overcome local minimal and slow global convergence in target searching for multiple robots. Simulation on an example of target searching is conducted to show the effectiveness of the proposed approach.

1. Introduction
Multiple moving robots have many advantages over a single robot under certain circumstances, such as rescue, target searching, and transportation. Multiple robots can cooperate more efficiently in unknown environment without human interference. In target searching problems, how to find the target and approaching it rapidly and safely remains an updating issue. Particle Swarm Optimization (PSO) is a typical global optimization algorithm [1, 2], which is inspired by bird flocking behaviors. The velocity-displacement model in PSO operates simply, and the memory of the particles can help trace the exploration and adjust the strategy accordingly. Because of the effectiveness, this algorithm has been applied comprehensively in many research areas, for example neural network [3]. In terms of target searching problems, the robots can be regarded as the particles in PSO. However, PSO is with low population diversity, and may fall into local minimal, so researchers dedicated to improve the algorithm over the past decades [4].

There are also many other optimization algorithms that applied into target searching tasks [5]. Bacterial Foraging Optimization (BFO) is a relatively new optimization technique, which is inspired by the foraging behavior of Escherichia coli [6]. It is not sensitive to the initial values and parameters, and the technique is easy to be realized. The process can be divided into four steps: chemotaxis, swarming and tumbling, reproduction, and elimination-dispersal [7-10]. Because of the movement mechanism, BFO has a better local searching ability than other similar algorithms. Based on the analysis of the PSO and BFO, a new method for target searching problem is proposed in this paper, which combined the advantages of both PSO and BFO to avoid local minimum and operate more efficiently.
2. Analysis of PSO and BFO

2.1. Particle Swarm Optimization

Particle swarm optimization is a kind of evolutionary algorithm and swarm intelligence. An optimization problem can be considered as a problem of searching for the best particle among a set of possible alternatives [1]. The initial position of each particle is generated randomly in \( n \)-dimensional space. Suppose there are \( m \) particles in the space, the set of swarm can be described as

\[
X = \{X_1, X_2, X_3, \cdots, X_m\}
\]  

(1)

Where the position of the \( i \)th particle is

\[
X_i = \{x_{i1}, x_{i2}, x_{i3}, \cdots, x_{in}\}
\]  

(2)

An objective function \( f \) is defined to specify the aim of the task, and optimization turns to finding the minimization of \( f \) by adjusting the value of parameters.

Each particle moves across the solution space iteratively, and searching for better solutions by changing their own positions. The renewed velocity of each particle depends on three terms. The first term is the current velocity of the particle, which defines the inertia and momentum. The second term considers the best position that is found by the particle itself previously. The third term accounts for the best position that is found by the whole swarm. The velocity update function is as follow

\[
v_i(t + 1) = \omega v_i(t) + \eta_1 (P_i - x_i(t))R_1 + \eta_2 (P_g - x_i(t))R_2
\]  

(3)

Where \( v_i(t) \) is the velocity of \( i \)th particle at time \( t \), while \( P_i \) is the best position of the particle previously found, and \( P_g \) is the best position of the swarm. \( \omega \) is the inertia weight, and \( \eta_1 \) and \( \eta_2 \) are acceleration constants. \( R_1 \) and \( R_2 \) are diagonal matrices comprised of random numbers in \([0, 1]\). The parameters \( \omega, \eta_1 \) and \( \eta_2 \) decide the relative importance of each term in function (3). So the position of particle can be renewed as

\[
x_i(t + 1) = x_i(t) + v_i(t + 1)
\]  

(4)

Iterations are conducted according to this rule. The algorithm will be ended if the optimal solution is found or the maximum iteration number is reached.

Because the initial values are generated randomly, the progress is semi-random searching. The particles may be stuck into local minimum. However, all the particles move toward the global optimal position, which results in a faster convergence.

2.2. Bacterial Foraging Optimization

Bacterial Foraging Optimization is an algorithm inspired by bacterial foraging processes. The bacteria move toward the food sources and away from the toxic ones. The main behavior includes four steps. (1) Looking for the area of food sources; (2) whether to access to the area; (3) foraging in the selected area; (4) deciding whether to continue foraging in this area or to move to other areas after consuming some nutrition. BFO contains four basic components: chemotaxis, gathering, reproduction, and elimination-dispersal [11]. In chemotaxis, the movement of the \( i \)th bacterium can be shown as follow

\[
\theta^i(j + 1, k, l) = \theta^i(j, k, l) + C(i) \frac{\Delta(i)}{\sqrt{\Delta^i(i)\Delta(i)}}
\]  

(5)
Where $\theta^i(j, k, l)$ is the $i$th bacterium at the $j$th chemotaxis, the $k$th reproduction and the $l$th elimination-dispersal. In the process of gathering, there are both attractive force and repellant force between bacteria. Attractive force can gather the bacteria, while repellant force can keep a bacterium at some position. This behavior can be expressed by function

$$J_{cc}(\theta, P(j, k, l)) = \sum_{i=1}^S J_{cc}(\theta, \theta^i(j, k, l)) =$$

$$\sum_{i=1}^S \left\{-d_{\text{attractant}} \exp(-\omega_{\text{attractant}} \sum_{m=1}^P (\theta_m - \theta^i_m)^2)\right\}$$

$$+ \sum_{i=1}^S \left\{-h_{\text{repellant}} \exp(-\omega_{\text{repellant}} \sum_{m=1}^P (\theta_m - \theta^i_m)^2)\right\}$$

(6)

Where $J$ is the fitness function, and $d_{\text{attractant}}, \omega_{\text{attractant}}, h_{\text{repellant}}$ and $\omega_{\text{repellant}}$ are parameters on attractive force and repellant force, which can be chosen as required. After a period of foraging, the bacteria with poor ability will be eliminated. Then the bacteria move to a new area where is more nutritious.

3. Proposed Method

Based on the advantages of PSO and BFO, a hybrid algorithm is proposed for target searching tasks. The schematic description of the method is as follow:

Step 1: Initialize the positions and velocities of the particles and other initial parameters.

Step 2: Loop for elimination-dispersal, reproduction, chemotaxis, i.e. $l = l + 1, k = k + 1, j = j + 1$.

Step 3: Calculate the fitness function $J(i, j, k, l)$, which is the distance to the target in this situation. Increase the repellant force to mimic gathering. The best fitness value is stored in $J_{\text{last}} = J(i, j, k, l)$. The velocity is updated by function (3) mentioned above to achieve rapid convergence.

Step 4: Calculate $J(i, j + 1, k, l)$, and deal with the next particle using the same method.

Step 5: Do chemotaxis step.

Step 6: Reproduction. Fitness value $J$ is sorted from small to large for each particle, and the particles with small values are eliminated, and the others with large values are reproduced.

Step 7: Elimination-dispersal. After a number of generations, the particles are distributed into the space with probability $P_{ed}$.

4. Simulation

In this paper, the target point is selected as point (1, 2), and the objective function is the distance to the target point for each particle. The moving area of particles is bounded as $x \in [-4, 8]; y \in [-4, 8]$. The proposed method is applied to search for the target point. The number of particles $S$ has impact on the effect of the algorithm. A smaller $S$ can increase the efficiency, but decrease the population diversity; a larger scale will increase computation consumption while have more opportunity to access to the target. The initial values of the parameters are chosen as shown in Table 1. The initial positions of the particles are generated randomly. The initial positions and final positions after searching are shown in Figure 1. The heights of the points on the surface imply the distances of the particles to the target point. Total number of iteration $i$ is set to be 100. Figure 2 demonstrates the positions of particles after a number of iterations when $i = 1, 5, 10, 20, 40, 100$ respectively. It can be seen that the final positions of the particles converge to the target point.

| $S$ | $\omega_{\text{max}}$ | $\omega_{\text{min}}$ | $P$ | $d_{\text{attractant}}$ | $\omega_{\text{attractant}}$ | $h_{\text{repellant}}$ | $\omega_{\text{repellant}}$ |
|-----|----------------|-------------------|----|----------------|----------------|----------------|----------------|
| 20  | 100            | 0.9              | 0.4| 2              | 0.02           | 0.05           | 0.02           | 6               |

Table 1. Initial parameters.
5. Conclusion
A hybrid optimization method is proposed in this paper, which combines the advantages of PSO and BFO. The position function in PSO converges more rapidly, so it is applied in this method and the main
steps in BFO are adopted. Simulation was conducted with an example of searching for the target in 2D space. The effectiveness of the proposed method is verified. More studies on higher dimensions can be done in the future to validate the method.

Acknowledgments
This work was financially supported by Natural Science Foundation of Tianjin Municipality (Grant No. 16JCYBJC41400), Research Program of Tianjin Educational Committee (Grant No. 2018KJ176), MOE Youth Foundation Project of Humanities and Social Sciences (Grant No. 18YJCJ760097). The authors would like to express their gratitude.

References
[1] Marini F, Walczak B. Particle swarm optimization (PSO). A tutorial [J]. Chemometrics and Intelligent Laboratory Systems, 2015, 149: 153-165.
[2] Kennedy J. Particle swarm optimization [J]. Encyclopedia of machine learning, 2010: 760-766.
[3] Bonyadi M R, Michalewicz Z. Particle swarm optimization for single objective continuous space problems: a review [J]. 2017.
[4] Tanweer M R, Suresh S, Sundararajan N. Self regulating particle swarm optimization algorithm [J]. Information Sciences, 2015, 294: 182-202.
[5] Hossain M A, Ferdous I. Autonomous robot path planning in dynamic environment using a new optimization technique inspired by bacterial foraging technique [J]. Robotics and Autonomous Systems, 2015, 64: 137-141.
[6] Passino K M. Biomimicry of bacterial foraging for distributed optimization and control [J]. IEEE control systems magazine, 2002, 22(3): 52-67.
[7] Mao L., Mao Y., Zhou C., et al. Particle swarm and bacterial foraging inspired hybrid artificial bee colony algorithm for numerical function optimization [J]. Mathematical Problems in Engineering, 2016.
[8] Das S, Biswas A, Dasgupta S, et al. Bacterial foraging optimization algorithm: theoretical foundations, analysis, and applications [M]//Foundations of Computational Intelligence Volume 3. Springer, Berlin, Heidelberg, 2009: 23-55.
[9] Dasgupta S, Das S, Abraham A, et al. Adaptive computational chemotaxis in bacterial foraging optimization: an analysis [J]. IEEE Transactions on Evolutionary Computation, 2009, 13(4): 919-941.
[10] Biswas A, Dasgupta S, Das S, et al. Synergy of PSO and bacterial foraging optimization—a comparative study on numerical benchmarks [M]//Innovations in hybrid intelligent systems. Springer, Berlin, Heidelberg, 2007: 255-263.
[11] Zhao W, Wang L. An effective bacterial foraging optimizer for global optimization [J]. Information Sciences, 2016, 329: 719-735.