Bat Algorithm with Individual Local Search

Maoqing Zhang, Zhihua Cui, Yu Chang, Yeqing Ren, Xingjuan Cai, Hui Wang

Bat Algorithm with Individual Local Search

Maoqing Zhang*, Zhihua Cui*+ , Yu Chang*, Yeqing Ren*, Xingjuan Cai*, Hui Wang**

*Complex System and Computational Intelligence Laboratory,
Taiyuan University of Science and Technology, Taiyuan, Shanxi, 030024, China
maoqing_zhang@163.com
zhihua.cui@hotmail.com
18834174241@163.com
18435155956@163.com
xingjuancai@163.com
+Corresponding author
**School of Information Engineering, Nanchang Institute of Technology,
Nanchang 330099, China
huiwang@whu.edu.cn (Hui Wang)

Abstract. Bat algorithm (BA) is a well-known heuristic algorithm, and has been applied to many practical problems. However, the local search method employed in BA has the shortcoming of premature convergence, and does not perform well in early search stage. To avoid this issue, this paper proposes a new update method for local search. To verify the proposed method, this paper employs CEC2013 test suit to test it with PSO and standard BA as comparison algorithms. Experimental results demonstrate that the proposed method obviously outperforms other algorithms and exhibits better performance.

Keywords: Bat algorithm; Premature convergence; CEC 2013; Particle Swarm Optimization algorithm

1 Instruction

Nature-inspired computation is an umbrella for stochastic optimization algorithms by simulating the nature phenomenon. Up to now, many algorithms have been proposed, such as particle swarm optimization [1], ant colony optimization [2,3], bat algorithm [4], cuckoo search [5,6,7] and firefly algorithm [8,9].

adfa, p. 1, 2011.

Bat algorithm (BA) is a novel population-based swarm intelligent [4]. Due to its fast convergent speed, BA has been widely applied to many engineering problems, including tracking problems [10], economic load dispatch [11], Detection of Malicious Code [12], uninhabited combat aerial vehicle path planning[13], flow shop scheduling [14,15], and job shop scheduling problems [16].

There are numerous variants of BA, which have been greatly improved in term of performance. Gandomi [17] introduced chaos into BA so as to increase its global search capability for robust global optimization. To solve path planning which is a complicated high dimension optimization problem, Wang [18] proposed a new bat algorithm with mutation (BAM), and a modification was applied to mutate between bats during the process of the new solutions updating. Jr [19] hybridized BA using different DE strategies and applied them as a local search heuristic for improving the current best solution and directing the swarm towards the better regions within a search space. Cui [20] proposed three different centroid strategies and further combined them with BA. AI-Betar [21] studied six selection mechanisms to choose the best bat location: global-best, tournament, proportional, linear rank, exponential rank, and random. Cai proposed a triangle-flipping strategy to update the velocity of bats[22] and designed a optimal forage strategy to guide the search direction for each bat and employed a random disturbance strategy to extend the global search pattern[23]

In this paper, to enhance the convergence ability of BA, we proposed a modified BA called Bat Algorithm with Individual local search (IBA). In IBA, a new local search manner is introduced to enhance the convergence speed of each individual in early search stage. Then, to strength the convergence of the global best individual in later search stage, the standard local search method in BA is incorporated into IBA.

The rest of paper organized as follows: section 2 gives a brief description of Bat algorithm. After that, the newly modified bat algorithm is presented. In section 3, CEC2013 is employed to verify the proposed algorithm. Section 4 concludes the paper.

2 Bat Algorithm

In bat algorithm, there are many virtual bats in search space, while each bat flies to seek food according to the feedback of echoes. Suppose x_i^t and v_i^t are the position and velocity of bat i in generation t, then in the next generation, they are updated as follows:

$$v_i^{t+1} = v_i^t + (x_i^t - x^*) \times f_i \tag{1}$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{2}$$

where x^* is the historical best position found by entire swarm, and f_i is the frequency randomly generated as follows:

$$f_i = f_{\min} + (f_{\max} + f_{\min}) \times \beta \tag{3}$$

where $f_{\max}$ and $f_{\min}$ are the predefined maximum and minimum bounds of frequency, β is a random number uniformly generated from interval [0,1].

For some bats, they may move with the following local search manner:

$$x_i^{t+1} = x^* + \varepsilon \times \bar{A}' \tag{4}$$

where ε is random number uniformly generated from interval [-1,1], and $\bar{A}'$ is averaged loudness of all bats:

$$\bar{A}' = \frac{\sum_{i=1}^{n} A_i^t}{n} \tag{5}$$

If the position of bat *i* is updated, the loudness A_i^{t+1} and emission rate r_i^{t+1} are updated as follows:

$$\mathrm{A}_i^{t+1} = \sigma \mathrm{A}_i^t \tag{6}$$

$$r_i^{t+1} = r_i^0 (1 - e^{-\gamma t}) \tag{7}$$

where $\sigma > 0$ and $\gamma > 0$ are pre-defined parameters.

The pseudocode of standard bat algorithm is described as follows:

Algorithm1: Standard bat algorithm

```
Begin
  Initialize position, velocity and other parameters for each bat
  While (Stop criteria is met ?)
    Randomly generates the frequency for each with Eq. (3)
    Update the velocity with Eq. (1)
    Update the position with Eq. (2)
    If rand> r_i^t
      Update the position with Eq. (4)
    End
    Calculate the fitness;
    If (rand< A_i^t )&&( f(x_i^t) < f(x^*) )
      Replace the position with the new one
      Update r_i^t and A_i^t with Eq.(7) and Eq.(6)
    End
    Select the current global best position
  End
```

Output the best position
End

3 Bat algorithm with individual local search

In standard BA, local search is performed with Equation (4), and it can be expressed in Figure 1, where x^* is the current global best position, x_{i-1}^t and x_i^t are population members, and the circles represent the potential better positions. From figure 1, we can find that equation (4) is mainly used to search potential better individuals around the current global best position. However, as we all know that, in the initial search stage, any position (even the current best position) has potential to be the best one. In other words, the individuals around the current global best position may not be better than those around regular positions. However, equation (4) may be effective in later search stage because the individuals around current global best position are likely to be the best ones compared with other regular positions.

To overcome the drawback in the early search stage, this paper proposes an improved version of equation (4), and it can be expressed with equation (8):

$$x_i^{t+1} = x_i^t + \varepsilon \times \bar{A}' \tag{8}$$

In the later search stage, position x_i^t is updated with equation [4]:

$$x_i^{t+1} = x^* + \varepsilon \times \bar{A}' \tag{4}$$

Figure 2 illustrates the mechanism of improved local search. From figure 2, we can obviously see that every position can be exploited sufficiently, and new potential better position can be found in early search stage. The newly modified BA can be named as Bat algorithm with individual local search (IBA for short). As for how to balance equation (4) and (8) in the search process, we employ a parameter r, which is a percentage of the largest generation. We will verify it in later experimental section.

The pseudocode of IBA can be described as follows:

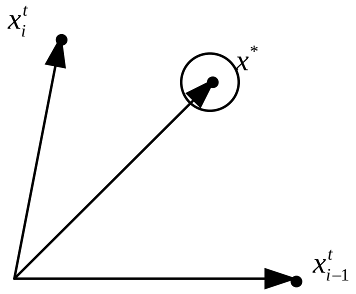

Figure 1. Illustration of local search

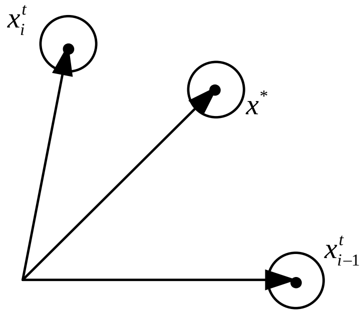

Figure 2. Illustration of improved local search

Algorithm2: Bat algorithm with individual local Search

Begin
  Initialize position, velocity and other parameters for each bat
  While (Stop criteria is met?)
    Randomly generates the frequency for each with Eq. (3)
    Update the velocity with Eq. (1)
    Update the position with Eq. (2)
    If (rand> r_i^t)&&(rand<***r)***
      Update the position with Eq. (8)
    Else if (rand> r_i^t)&&(rand>=***r)***
      Update the position with Eq. (4)
    End
    Calculate the fitness;
    If (rand< A_i^t)&&($f(x_i^t) < f(x^*)$)
      Replace the position with the new one
      Update r_i^t and A_i^t with Eq. (7) and Eq. (6)
    End
    Select the current global best position
  End
  Output the best position
End

4 Experiments

To investigate the performance of IBA, we compare it with particle swarm optimization (PSO) [1] and standard bat algorithm (BA) [4]. By the way, CEC2013 test suit [24] is employed. CEC2013 contains of twenty-eight functions, including five unimodal functions, fifteen basic multi-modal functions and eight composition functions. The simulation is conducted in the machine with Intel Core *i*5-2400 3.10GHz CPU, 6GB memory, and windows 7 operating system with Matlab7.9.

For each algorithm, the size of population is set to 100, and the dimension of search space is 30. Each algorithm is run 51 times. For each algorithm, the following mean error is considered:

$$MeanError = \frac{\sum_{i=1}^{51}(f_{best}^{j} - f^{*})}{51} \tag{9}$$

where f_{best}^{j} represents the fitness value of obtained best solution, f^{*} is the fitness value of true global optimum.

4.1 Investigation of parameter r

Table 1 and Table 2 present the experiment results with fixed parameter r ranging from 0.1 to 1.0. It is obviously that IBA with r =0.1, 0.2 and 0.4 performs similar performance. To further investigate the difference, Friedman test is also employed, and the results are listed in Tab.3. From Tab.3, we can that BA with parameter r =0.4 performs the best performance.

Table 1. Comparison results with different parameter ***r*** from 0.1 to 0.5

Function	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**
F1	1.55E+00	1.22E+00	1.15E+00	1.10E+00	**8.60E-01**
F2	**4.08E+06**	4.32E+06	5.85E+06	8.33E+06	9.10E+06
F3	4.46E+08	**4.03E+08**	1.24E+09	1.45E+09	5.37E+09
F4	**3.43E+04**	3.50E+04	4.28E+04	4.06E+04	4.60E+04
F5	4.74E-01	**4.74E-01**	2.34E+01	6.54E+01	1.59E+02
F6	7.80E+01	**6.50E+01**	9.40E+01	1.10E+02	9.01E+01
F7	3.31E+02	2.17E+02	2.17E+02	2.78E+02	2.01E+02
F8	2.10E+01	2.10E+01	2.10E+01	2.10E+01	**2.10E+01**
F9	3.50E+01	3.33E+01	3.22E+01	3.31E+01	3.34E+01
F10	**1.30E+00**	1.37E+00	1.44E+00	2.03E+00	7.24E+00
F11	4.34E+02	4.08E+02	4.04E+02	**3.48E+02**	3.97E+02
F12	4.59E+02	4.34E+02	4.09E+02	3.73E+02	4.07E+02
F13	4.56E+02	**4.13E+02**	4.50E+02	4.42E+02	4.33E+02
F14	4.23E+03	4.34E+03	3.81E+03	3.92E+03	3.70E+03
F15	3.93E+03	4.14E+03	3.63E+03	4.02E+03	4.05E+03
F16	2.26E+00	2.22E+00	2.45E+00	2.15E+00	2.14E+00
F17	9.09E+02	**9.04E+02**	9.84E+02	9.69E+02	9.92E+02
F18	9.65E+02	9.35E+02	9.74E+02	**9.33E+02**	9.60E+02
F19	**6.70E+01**	7.10E+01	8.54E+01	1.13E+02	1.14E+02
F20	1.48E+01	**1.45E+01**	1.46E+01	1.46E+01	1.49E+01
F21	**3.04E+02**	3.94E+02	3.68E+02	3.41E+02	3.67E+02

F22	5.19E+03	4.73E+03	4.20E+03	4.41E+03	4.34E+03
F23	5.08E+03	4.90E+03	4.33E+03	4.48E+03	4.83E+03
F24	3.26E+02	3.25E+02	3.20E+02	**3.18E+02**	3.25E+02
F25	**3.39E+02**	3.44E+02	3.46E+02	3.48E+02	3.55E+02
F26	**2.00E+02**	2.00E+02	2.00E+02	2.00E+02	2.00E+02
F27	1.33E+03	1.27E+03	1.30E+03	**1.23E+03**	1.32E+03
F28	3.78E+03	**3.17E+03**	3.30E+03	3.25E+03	3.63E+03

Table 2. Comparison results with different parameter ***r*** from 0.6 to 1.0

Function	**0.6**	**0.7**	**0.8**	**0.9**	**1.0**
F1	1.05E+00	1.03E+02	3.53E+03	1.49E+04	3.16E+04
F2	1.32E+07	2.31E+07	2.85E+07	5.54E+07	2.15E+08
F3	8.15E+09	1.49E+10	1.77E+10	4.97E+10	7.65E+11
F4	4.35E+04	5.34E+04	5.56E+04	5.57E+04	6.87E+04
F5	2.59E+02	4.68E+02	1.43E+03	2.66E+03	7.25E+03
F6	1.25E+02	1.64E+02	4.19E+02	9.58E+02	3.72E+03
F7	2.45E+02	5.12E+02	3.03E+02	4.65E+02	9.09E+02
F8	2.09E+01	**2.09E+01**	2.09E+01	2.10E+01	2.09E+01
F9	**3.13E+01**	3.37E+01	3.17E+01	3.47E+01	3.60E+01
F10	4.73E+01	1.44E+02	5.61E+02	1.52E+03	3.65E+03
F11	4.07E+02	4.39E+02	3.97E+02	3.78E+02	4.76E+02
F12	4.37E+02	3.69E+02	**3.66E+02**	3.85E+02	4.30E+02
F13	4.71E+02	4.22E+02	4.53E+02	4.59E+02	5.77E+02
F14	3.42E+03	3.78E+03	3.89E+03	**3.41E+03**	4.21E+03
F15	3.74E+03	**3.45E+03**	3.54E+03	3.95E+03	4.48E+03
F16	**2.12E+00**	2.51E+00	2.13E+00	2.39E+00	2.49E+00
F17	9.40E+02	9.80E+02	9.90E+02	1.09E+03	1.16E+03
F18	9.60E+02	9.60E+02	1.04E+03	9.70E+02	1.11E+03
F19	1.38E+02	1.48E+02	3.41E+02	6.74E+03	1.35E+05
F20	1.48E+01	1.49E+01	1.48E+01	1.50E+01	1.49E+01
F21	4.33E+02	3.40E+02	1.09E+03	1.93E+03	2.70E+03
F22	4.13E+03	4.39E+03	4.26E+03	**4.07E+03**	4.87E+03
F23	4.43E+03	4.27E+03	**4.16E+03**	4.50E+03	4.98E+03
F24	3.20E+02	3.25E+02	3.25E+02	3.33E+02	3.36E+02
F25	3.47E+02	3.44E+02	3.53E+02	3.45E+02	3.53E+02
F26	2.00E+02	2.00E+02	2.01E+02	2.03E+02	2.27E+02
F27	1.29E+03	1.26E+03	1.27E+03	1.24E+03	1.35E+03
F28	3.37E+03	3.53E+03	3.46E+03	3.53E+03	4.45E+03

Table 3. Friedman test for parameter ***r***

r	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Rankings	5.86	4.71	5.00	**4.46**	6.00	5.36	6.23	6.50	7.61	10.46

4.2 Comparison with state-of-art algorithms

To further verify the performance of IBA with ***r***=0.4, IBA is further compared with PSO and standard BA, and the simulation results are listed in Tab.4, while *w/l/t* means that IBA win in *w* functions, lose in *l* functions and tie in *t* functions. Friedman test is also employed to verify their performance (please refer to Table 5). In one word, our modification achieves the best performance when compared with PSO and BA.

Table 4. Comparison results

Function	PSO	BA	IBA
F1	2.9600E+04	1.8300E+00	**1.1000E+00**
F2	4.0200E+08	**3.2600E+06**	8.3300E+06
F3	1.6500E+14	**2.2600E+08**	1.4500E+09
F4	5.2500E+04	**3.4400E+04**	4.0600E+04
F5	1.0400E+04	**4.7400E-01**	6.5400E+01
F6	4.5400E+03	**5.9900E+01**	1.1000E+02
F7	1.6000E+04	**1.7500E+02**	2.7800E+02
F8	**2.1000E+01**	**2.1000E+01**	**2.1000E+01**
F9	4.0500E+01	3.5700E+01	**3.3100E+01**
F10	3.9800E+03	**1.3200E+00**	2.0300E+00
F11	5.7200E+02	3.8100E+02	**3.4800E+02**
F12	5.1800E+02	4.0400E+02	**3.7300E+02**
F13	5.4300E+02	4.7300E+02	**4.4200E+02**
F14	8.2100E+03	4.6100E+03	**3.9200E+03**
F15	7.1900E+03	5.1700E+03	**4.0200E+03**
F16	2.5900E+00	2.3000E+00	**2.1500E+00**
F17	**7.2500E+02**	9.3100E+02	9.6900E+02
F18	**7.1200E+02**	9.4900E+02	9.3300E+02
F19	9.5200E+04	**6.1500E+01**	1.1300E+02
F20	1.5000E+01	**1.4600E+01**	**1.4600E+01**
F21	2.2800E+03	3.7000E+02	**3.4100E+02**
F22	8.5500E+03	5.6600E+03	**4.4100E+03**
F23	8.3800E+03	5.7300E+03	**4.4800E+03**
F24	3.7600E+02	**3.1200E+02**	3.1800E+02
F25	3.9200E+02	**3.4300E+02**	3.4800E+02
F26	2.4400E+02	**2.0000E+02**	**2.0000E+02**
F27	1.4900E+03	1.2600E+03	**1.2300E+03**
F28	4.4300E+03	3.7900E+03	**3.2500E+03**
w\l\t	25\2\1	14\11\3	

Table 5. Friedman test on comparison algorithms

Algorithm	Rankings
PSO	2.86
BA	1.63
IBA	**1.52**

5 Conclusion

In this paper, an individual local search manner is designed to enhance the exploitation. With this manner, the bats can make a deep search within their neighbours during the early search period. Simulation results show its effectiveness.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grant No.61663028, Natural Science Foundation of Shanxi Province under Grant No.201601D011045 and Graduate Educational Innovation Project of Shanxi Province under Grant No.2017SY075.

References

1. Eberhart R, Kennedy J. A new optimizer using particle swarm theory, International Symposium on MICRO Machine and Human Science. IEEE, 39-43 (2002).
2. Dorigo M, Birattari M, Stutzle T. Ant colony optimization. IEEE Computational Intelligence Magazine, 1(4): 28-39 (2007).
3. Xu B, Zhu J, Chen Q. Ant Colony Optimization, New Advances in Machine Learning. InTech, p. 1155-1173 (2010).
4. Yang X S. A New Metaheuristic Bat-Inspired Algorithm. Computer Knowledge & Technology, 284:65-74 (2010).
5. Yang X S, Deb S. Cuckoo Search via Lévy flights, Nature & Biologically Inspired Computing, NaBIC 2009. 2010:210-214 (2009).
6. Zhang M, Wang H, Cui Z, et al. Hybrid multi-objective cuckoo search with dynamical local search. Memetic Computing, 10(2):199-208 (2018).
7. Cui Z, Sun B, et al, A novel oriented cuckoo search algorithm to improve DV-Hop performance for cyber-physical systems, Journal of Parallel and Distributed Computing, 103:42-52(2017).
8. Yang X S. Firefly Algorithms for Multimodal Optimization. Mathematics, 5792:169-178 (2009).
9. Babak Nasiri, Mohammad Reza Meybodi, History-driven firefly algorithm for optimisation in dynamic and uncertain environments, International Journal of Bio-Inspired Computation, 8(5): 326-339(2016).
10. Gao M L, Shen J, Yin L J, et al. A novel visual tracking method using bat algorithm. Neurocomputing, 177©: 612-619 (2016).
11. Pham L H, Ho T H, Nguyen T T, et al. Modified Bat Algorithm for Combined Economic and Emission Dispatch Problem., Proceedings of the International Conference on Advances in Electrical Engineering and Related Sciences (2016).
12. Cui Z, Xue F, et al, Detection of malicious code variants based on deep learning, IEEE Transactions on Industrial Informatics,14(7): 3187-3196(2018).
13. Luo Q, Li L, Zhou Y, A quantum encoding bat algorithm for uninhabited combat aerial vehicle path planning, International Journal of Innovative Computing and Applications ,8(3):182-193(2017).

14. Marichelvam M K, Prabaharan T, Yang X S, et al. Solving hybrid flow shop scheduling problems using bat algorithm. International Journal of Logistics Economics & Globalisation, 5(1):15-29(2013).
15. Ömür Tosun, M.K.M, Hybrid bat algorithm for flow shop scheduling problems, International Journal of Mathematics in Operational Research, 9(1):125-138(2016) .
16. Dao T K, Pan T S, Nguyen T T, et al. Parallel bat algorithm for optimizing makespan in job shop scheduling problems. Journal of Intelligent Manufacturing, 29(2):1-12 (2015).
17. Gandomi A H, Yang X S. Chaotic bat algorithm. Journal of Computational Science, 5(2): 224-232 (2014).
18. Wang G, Guo L, Hong D, et al. A Bat Algorithm with Mutation for UCAV Path Planning. The Scientific World Journal, 2012(6):418946(2012).
19. Jr F I, Fong S, Brest J, et al. A novel hybrid self-adaptive bat algorithm. The scientific world journal, 2014 (1-2):709-738(2014).
20. Cui Z, Cao Y, Cai X, et al. Optimal LEACH protocol with modified bat algorithm for big data sensing systems in Internet of Things. Journal of Parallel & Distributed Computing, DOI:10.1016/j.jpdc.2017.12.014(2017).
21. Al-Betar M A, Aw adallah M A, Faris H, et al. Bat-inspired Algorithms with Natural Selection mechanisms for Global optimization. Neurocomputing, (2017).
22. Cai X, Wang H, et al, Bat algorithm with triangle-flipping strategy for numerical optimization, International Journal of Machine Learning and Cybernetics, 9(2):199-215(2018).
23. Cai X, Gao X , Xue Y, Improved bat algorithm with optimal forage strategy and random disturbance strategy, International Journal of Bio-inspired Computation, 8(4):205-214(2016).
24. Liang J J, Runarsson T P, Mezura-Montes E, et al. Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization. International Journal of Computer Assisted Radiology & Surgery (2005).