2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD) | 978-1-7281-6597-4/21/$31.00 ©2021 IEEE | DOI: 10.1109/CSCWD49262.2021.9437657

Backtracking Search Algorithm based on Knowledge of Different Populations for Continuous Optimization Problems

Fuqing Zhao[1], Jinlong Zhao[1] , Xiaotong Hu[1]
[1]School of Computer and Communication Technology, Lanzhou University of Technology Lanzhou 730050, China
Fzhao2000@hotmail.com(Fuqing Zhao)
1409776617@qq.com(Jinlong Zhao)
1107223200@qq.com(Xiaotong Hu)

Yi Zhang[2],Weimin Ma[3]
[2]School of Mechnical Engineering, Xijin University Xi'an710123, China
1049440546@qq.com

[3]School of Economics and Management, Tongji University, Shanghai 200092, China
mawm@tongji.edu.cn

***Abstract*—Backtracking search algorithm (BSA) has been applied to solve the various optimization problems in recent years. However, BSA is difficult to solve non-separable problems due to its single search mechanism. In this paper, backtracking search algorithm based on knowledge of different populations, named DKBSA, is proposed to solve continuous optimization problems. In DKBSA, sub-population partitioning method is used to enhance the local search ability and alleviate the loss rate of the diversity of population. Afterwards, a mutation strategy with knowledge guidance and rotation invariance, which is based on the current sub-population information and historical information, is designed to improve the convergence speed of the DKBSA. Furthermore, a control parameter of adaptive search factor is embedded in the mutation strategy to balance the exploitation and exploration of the proposed algorithm. Finally, a probabilistic model-based strategy is proposed to generate dominant individuals to further improve the search ability of the proposed algorithm. The experimental results of the state-of-the-art algorithms in the CEC2017 benchmark test suit reveal that the DKBSA is effective for solving non-separable problems.**

Keywords—Backtracking search algorithm; sub-population; diversity; rotation invariance

I. INTRODUCTION

Artificial intelligence, a branch of computer science, has developed rapidly in several decades. As an important research area of artificial intelligence, evolutionary algorithms (EAs) are adopted to solve numerous problems in various domains due to the robustness of EAs, for instance, flow shop scheduling problems [1], image processing applications [2] and, speech recognition problems [3]. However, traditional mathematical methods are difficult and impossible to address various problems, such as non-linear, multi-extreme and non-differentiable.

The meta-heuristics based on swarm intelligence, including genetic algorithm (GA) [4], particle swarm optimization (PSO) [5], differential evolution algorithm (DE) [6], artificial bee colony algorithm (ABC) [7], biogeography-based optimization (BBO) [8], brain storming algorithm (BSO) [9], have been attracted attention in scholars and researchers owing to its desirable parallelism, strong generality and low dependence on solving certain optimization problems. Therefore, the meta-heuristics based on swarm intelligence are an effective method to solve single-objective continuous optimization problem.

Backtracking search optimization algorithm (BSA) is a novel swarm intelligence algorithm proposed by Civicioglu in 2013 [10]. BSA has simple structure and one parameter, which is easy to adapt to certain simple and complex problems. In the BSA, the historical information of the current population is stored in the historical population, which is equivalent to a historical memory bank. Afterwards, the historical information in the BSA is a reflection of the search path of the past population. The past experience is adequately utilized to guarantee that the algorithm have a relatively stable search performance and the sufficient diversity of population. Therefore, the historical population and current population are adopted to generate the trial population in mutation phase of the BSA, which is a considerable difference between the BSA and other swarm intelligence algorithms. In addition, the crossover operator is different from other swarm intelligence algorithms. As a result, BSA has been successfully applied to a variety of practical engineering problems in recent years, for instance, controller design of torque motor systems [11], power flow systems [12], distributed generator assigning [13] and so on. At the same time, BSA has been successfully used to solve multi-objective optimization problems [14], in scheduling problems, BSA is also widely used, for instance reactive power dispatch problems [15], the IEEE 14-bus test system [16].

However, BSA also has certain disadvantages. For example, random search step length is difficult to balance the exploration and exploitation. Due to the lack of optimal information in mutation strategy, it is difficult to improve the convergence speed of the algorithm. Meanwhile, the rotation invariance of mutation operator is not satisfactory. Therefore, it is difficult to find the optimal solution when certain CEC benchmark problems with shift and rotation are to be solved. In addition, the rapid decline for the diversity of population leads to the undesirable quality of solutions during the search process of

This work was financially supported by the National Natural Science Foundation of China under grant numbers 61663023. It was also supported by the Key Research Programs of Science and Technology Commission Foundation of Gansu Province (2017GS10817), Lanzhou Science Bureau project (2018-rc-98), Public Welfare Project of Zhejiang Natural Science Foundation (LGJ19E050001), Wenzhou Public Welfare Science and Technology project (G20170016), respectively.

978-1-7281-6597-4/21/$31.00 ©2021 IEEE

BSA and the optimal solution is difficult to find on account of the weakness ability of local search within a certain number of function evaluation. Owing to the above shortcomings of BSA, various scholars proposed certain mutation strategies to improve the performance of the BSA. The typical works are as follows.

Inspired by the simulated annealing algorithm, Wang, Hu et al. (2018) applied the design idea of Metropolis criterion to the search control factor F of the BSA and designed an adaptive F that could change the amplitude. In this algorithm, F is related to the difference of fitness between different individuals and the current iteration number of the algorithm, and F decreases with the increase of iteration number. Through the analysis of experimental results, the proposed algorithm is obviously faster than the BSA in convergence speed and better than the BSA in balanced exploration and exploitation ability. In addition, no new parameters are generated in the algorithm, and the structure of the algorithm is not complicated. However, the algorithm does not pay attention to the improvement of population diversity, it is poor performance in robustness.

Inspired by PSO and teaching-learning-based optimization, Chen, Zou [17] proposed a new variant of BSA named learning backtracking search algorithm (LBSA). In the LBSA, a new mutation strategy is proposed, which not only learns from the different individuals in the current population, but also from the historical population. In order to improve the convergence speed of the algorithm, it also learns from the best individual in the current population. Meanwhile, the worst individual in the current population was removed in the mutation process to improve the diversity of the population. The experimental results reveal that LBSA is obviously superior to the BSA in both convergence speed and population diversity. Since the three equations in the mutation strategy are randomly switched, the stability of the algorithm is not well.

In order to further improve the diversity of the population and the local search ability of the algorithm, Ahandani, Ghiasi [18] proposed a BSA variant based on multiple populations, named shuffled backtracking search optimization algorithm (SBSA). In the SBSA, the current population is divided into several sub-populations, and each sub-population makes use of the mutation, crossover and selection strategy of the BSA to conduct local search. Experiments reveal that SBSA improved the diversity of population and the local search ability. However, since each sub-population is searched independently, it is poor performance in solution accuracy.

In view of the shortcomings of the BSA algorithm analyzed above, a new BSA variant named backtracking search algorithm based on knowledge of different populations (DKBSA) is proposed in this paper. The contributions of the algorithm are as follows.

- A new mutation strategy is proposed to enhance rotation invariance. An adaptive control parameter of scale factor F introduced to balance the exploration and exploitation of the DKBSA.
- A sub-population classification method based on clustering instead of random partition is proposed. To improve the local search ability of the DKGBSA. Each sub-population has unique domain knowledge for other sub-populations to use, and the sub-population are regrouped before each iteration to improve the diversity of the population.
- A new method is proposed to generate the dominant individual based on the best individual in the current population and replace the worst individual in the current population. The goal is to improve the quality of the population.

The remainder of this paper is organized as follows. Section 2 gives a brief introduction to the BSA. Section 3 introduces DKBSA. Section 4 gives the experimental results. The conclusion and the study of the future work are presented in the Section 5.

II. BACKTRACKING SEARCH ALGORITHM (BSA)

BSA is a new population-based iterative EA. However, different from other algorithms, BSA uses both current and historical population to generate a search-direction matrix. BSA is divided into five main steps, i.e., initialization, selection-I, mutation, crossover and selection-II.

Step1: Initialization. BSA initialize the current population P and historical population oldP using the following equations:

$$P_{i,j} \sim U(low_j, up_j) \tag{1}$$

$$oldP_{i,j} \sim U(low_j, up_j) \tag{2}$$

for $i = 1,2,3,\dots,N$ and $j = 1,2,3,\dots,D$,where N is the population size, D is the problem dimension, U is the uniform distribution.

Step2: Selection-I. the historical population is updated with a random probability in the selection-I process.

$$oldP_i = \begin{cases} P_i, & if\ a < b \\ oldP_i, & otherwise \end{cases} \tag{3}$$

where a and b are the uniformly distributed real numbers. After oldP is determined, the following equation is used to randomly change the order of the individuals in oldP:

$$oldP = permuting(oldP) \tag{4}$$

Step3: Mutation. The historical population and current population are used to generate the trial population Mutant using the following equation:

$$Mutant = P + F \times (oldP - P) \tag{5}$$

where, F is a search control factor that controls the amplitude of the search-direction matrix$(oldP - P)$. In the BSA uses the value $F = 3.rndn$, where $rndn \sim N(0,1)$ (N is the standard normal distribution).

Step4: Crossover. The final trial population T is generated in BSA's crossover process using the following equation:

$$T_{i,j} = \begin{cases} P_{i,j}, & if\ map = 1 \\ Mutant_{i,j}, & if\ map = 0 \end{cases} \tag{6}$$

the crossover process of BSA is divided into two steps. In the first step, map matrix is generated with two strategies. In the

second step, map matrix is used to generate the final experimental population.

Step5: selection-II. In this phase, a greedy strategy is used to generate individuals in the next generation.

$$P_i = \begin{cases} T_i, & if\ f(T_i) \le f(P_i) \\ P_i, & if\ f(T_i) > f(P_i) \end{cases} \quad (7)$$

where, f is the fitness function used to calculate the fitness value of an individual.

III. Backtracking Search Algorithm Based On Knowledge of Different Populations (DKBA)

Although the previous BSA variants have greatly improved the disadvantages of the BSA, there are still several problems. First, certain optimization problems are Non-separable, and researchers of BSA have not proposed a reasonable mutation operator to solve such problems. Second, BSA researchers did not propose an effective replenishment strategy for population diversity. For certain problems still existing in the BSA discussed above, this paper designs the following improved methods. The basic flow chart of DKBSA is shown in Figure 1.

A. Mutation strategy in the DKBSA

Certain historical knowledge has been used in the mutation operator of the BSA, but the source of knowledge should not only come from the historical population, but also from the current population. At the same time, it is necessary to learn different knowledge in different stages of algorithm iteration. Moreover, in order to enhance the rotation invariability of BSA, a new mutation strategy is proposed in this paper. The detailed mutation strategy of DKBSA described as follows.

The mutation strategy of DKBSA is divided into two phases. In the early stage of algorithm iteration, individuals make full use of the information of current population to improve their evolutionary ability and make full use of the information of historical population to maintain the diversity of population. The equation as follows:

$$Mutant_i = P_i + F \cdot (P_{r1} - P_i) + F \cdot (hP_{r2} - hP_{r3}) \quad (8)$$

where, $i = 1,2,3,\dots,N$. P_{r1} is an individual randomly selected from the current population . hP_{r2} , hP_{r3} are two different individuals in the historical population. The Eq.(8) is triggered when $nfes \le \alpha * Maxnfes$ is satisfied. Where, $nfes$ is the current evaluation number of the function, and $Maxnfes$ is the maximum evaluation number of the function. α is a random number between 0 and 1.

In the late stage of algorithm iteration, the algorithm mainly focuses on convergence speed, the best individual in all sub-populations is used to guide the population to converge to the optimal solution of algorithm quickly.

$$Mutant_i = P_i + F \cdot (P_{best} - P_i) \quad (9)$$

where, P_{best} is the best individual of all sub-populations. The Eq.(9) is triggered when $nfes > \alpha * Maxnfes$ is satisfied. F is a search control factor and adopts the same value as the BSA in DKBSA.

$$F = \rho.rndn \quad (10)$$

$$\rho = C_{max} - rand \cdot \left(-\left(\sin\left(\frac{\pi}{2 * n}\right) + C_{min}\right)\right) \quad (11)$$

where, ρ is designed to change the step length of the mutation. n is the current number of iterations, C_{min} and C_{max} are two real factors.

B. Population grouping strategy in the DKBSA

At present, there are certain methods for generating sub-populations, including random partition and cluster partition. Random partitioning maintains the diversity of the population better than cluster partitioning, but the main purpose of population grouping is to use different information of each sub-population, which makes cluster partitioning more reasonable.

The size of each sub-population is different, which is an inevitable phenomenon. However, the number of sub-populations has not yet been accurately classified, thus this paper does not focus on the discussion.

There is also an common problem to consider when plotting sub-populations. In the early iterations of the algorithm, individuals in the sub-population cannot be nearly identical, otherwise the population loses its ability to evolve. Therefore, a new approach to this problem is used in the DKBSA. First, the best and worst individuals in the sub-population are replaced before each iteration. Second, the sub-population is regrouped before each iteration.

C. Dominant individual generation strategy

In a population based on clustering, the vicinity of a dominant individual is generally the dominant individual. Therefore, using a probability distribution to generate certain individuals around the optimal individuals is an effective method. In this paper, the following equation is used to generate dominant individuals.

$$P_* = N(\mu, \sigma^2) \quad (12)$$

where, P_* is the newly generated dominant individual, N is a gaussian distribution, μ and σ are its mean and standard deviation, respectively. The values of μ determines the central location and the value of σ determines scope of the dominant individuals. In this paper, the values of μ and σ are shown as follows.

$$\mu = P_{best,j} \quad (13)$$

$$\sigma = \sqrt{meanP_j} \quad (14)$$

where, $j = 1,2,3,\dots,D$. $P_{best,j}$ is the jth gene of the best individual in the current population, and $meanP_j$ is the mean of the jth gene of the current population. At the end of each iteration, P_* is used to replace the worst individual in the current population.

D. Pseudo-code of the DKBSA

The pseudo-code for the DKBSA is as follows according to the previous analysis.

Step1: Initialize $N, D, n, mixrate, low, up, C_{max}, C_{min}$

Step2: Initialize $P, oldP$

Step3: Evaluate all individuals

Step4: generate initial evolution sub-populations and historical sub-populations

Step5: Calculate $meanP_j$

Step6: generate the new historical population and the initial trial population

Step7: Execute crossover operator to generate the final trial population

Step8: Execute all individuals to generate the next population

Step9: generate the dominant individual to replace the worst individual in the next generation

Step10: Regroup the population to form new sub-populations

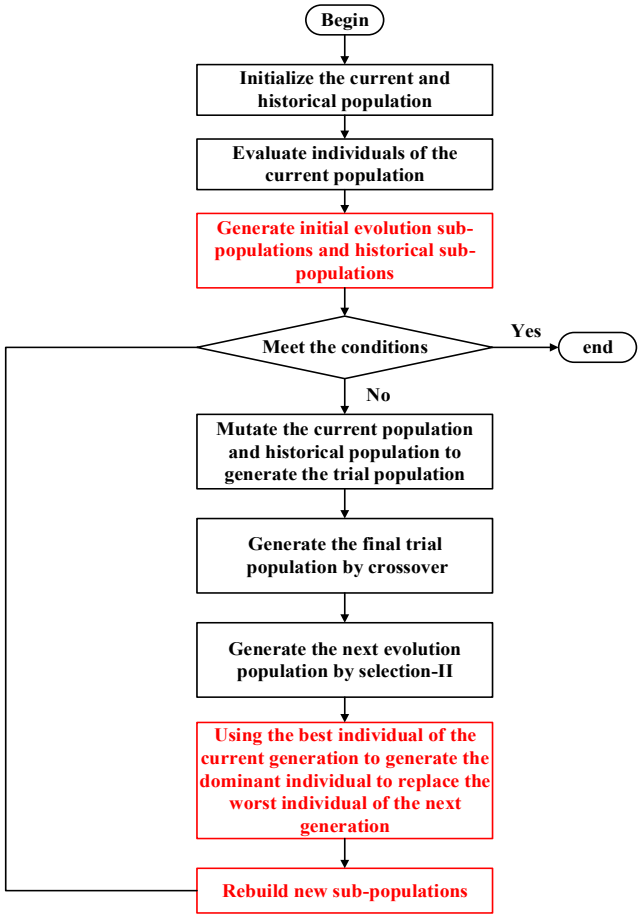


Fig.1 The flow chart of the DKBSA

IV. Simulation Results

In this paper, CEC 2017's standard test function is used to test the performance of DGKBSA. The five algorithms tested were the condition that problem dimension D=10, the population size N=50 and The maximum number of function evaluation $Maxnfes$ =10000*D.

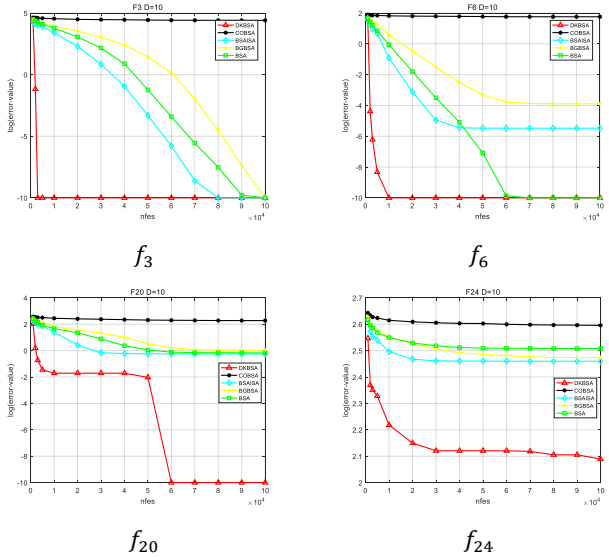
A. Parameter settings

In this paper, DKBSA is compared with the most BSA and the most advanced three variants of BSA, namely BGBSA [19], COBSA [20] and BSAISA [21]. The parameters of all algorithms are set as follows.

Algorithm	Parameters
BSA:	$F = 3 * randn, Mixrate = 1.0$
BGBSA:	$F = 3 * randn, Mixrate = 1.0, \alpha = rand$
COBSA:	$F = 3 * randn, Mixrate = 1.0\ m{\sim}N(1,0.3)$
BSAISA:	$Mixrate = 1.0$
DKBSA:	$C_{max} = 3, C_{min} = 1, Mixrate = 1.0$

B. Analysis and discussion of experimental results

Table.1 shows the mean error and standard deviation of the five algorithms with D = 10. In 29 test functions, DKBSA is superior to the other four comparison algorithms in results. More importantly, DKBSA finds the optimal solution stably on f1, f3, f4, f6, f9, f12, f20. The Wilcoxon's test is used to compare DKBSA with the other four comparison algorithms in pairs and detect the significant difference between them. The Wilcoxon's test analysis results of DKBSA and the other four comparison algorithms are shown in table 2. In table 2, R^+ is the sum of the rank that DKBSA is superior to the current comparison algorithm. R^- is the sum of the rank that the current comparison algorithm is superior to DKBSA. As can be seen from table 2, the value of R^+ in the comparison results of DKBSA in the four groups is higher than R^-.

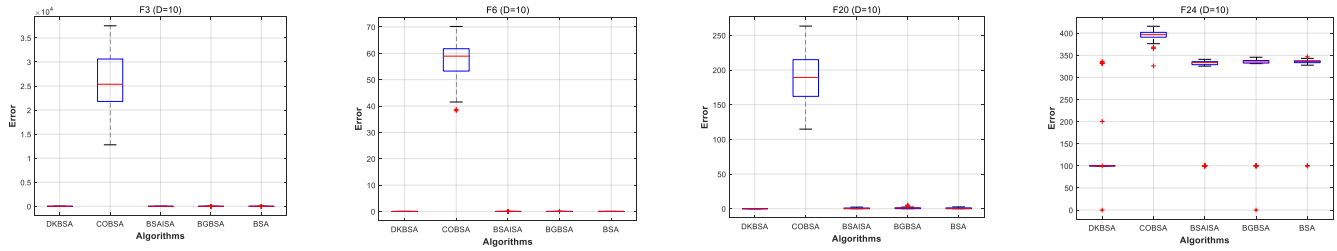


f_3 f_6 f_{20} f_{24}

Fig.2 Convergence curves of five algorithms for certain typical benchmark functions (10D)

Fig.2 and Fig.3 respectively shows the Convergence curves and Boxplots of five algorithms on f3, f6, f20 and f24, with D=10. These four functions represent four different types of functions in CEC2017, which are Unimodal, Simple Multimodal, Hybrid and Composition respectively. It can be seen from the convergence curve that DKBSA is faster than other comparison algorithms in convergence speed. Analysis

Boxplots shows that DKBSA is smaller than other comparison algorithms in standard deviation. In conclusion, DKBSA is superior to other comparison algorithms in global performance.



f_3 f_6 f_{20} f_{24}

Fig.3 Boxplots of five algorithms for certain typical benchmark functions (10D)

TABLE I. THE RESULT OF FIVE ALGORITHMS (10-DIMENSIONAL BENCHMARK FUNCTION

Fun	BSA		COBSA		BGBSA		BSAISA		DKBSA	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F1	0.00E+00	2.52E-09	5.62E+09	1.20E+09	3.05E-05	2.09E-04	7.53E-10	5.28E-09	**0.00E+00**	0.00E+00
F3	0.00E+00	0.00E+00	2.56E+04	6.03E+03	0.00E+00	1.19E-10	0.00E+00	0.00E+00	**0.00E+00**	0.00E+00
F4	6.85E-01	3.99E-01	3.76E+02	8.47E+01	5.68E-01	4.89E-01	3.16E+00	7.82E-01	**0.00E+00**	0.00E+00
F5	5.08E+00	1.62E+00	8.20E+01	8.43E+00	5.56E+00	2.17E+00	4.20E+00	1.61E+00	**3.32E-01**	6.19E-01
F6	0.00E+00	0.00E+00	5.72E+01	6.87E+00	1.28E-04	4.99E-04	3.32E-06	2.09E-05	**0.00E+00**	0.00E+00
F7	1.52E+01	1.71E+00	3.03E+02	8.01E+01	1.55E+01	2.30E+00	1.39E+01	1.63E+00	**6.89E+00**	3.66E+00
F8	4.84E+00	2.18E+00	9.87E+01	9.29E+00	5.48E+00	2.62E+00	3.66E+00	1.71E+00	**3.71E-01**	5.96E-01
F9	3.87E-02	1.61E-01	2.20E+03	5.48E+02	0.00E+00	0.00E+00	4.81E-02	1.65E-01	**0.00E+00**	0.00E+00
F10	3.28E+02	1.43E+02	1.33E+03	1.81E+02	2.92E+02	1.69E+02	1.08E+02	9.18E+01	**1.51E+01**	1.48E+01
F11	2.10E+00	1.76E+00	8.57E+02	3.15E+02	1.66E+00	1.48E+00	2.65E+00	2.70E+00	**3.43E-02**	2.86E-01
F12	4.34E+02	6.72E+02	1.69E+08	6.51E+07	5.18E+02	1.09E+03	2.28E+02	2.68E+02	**0.00E+00**	0.00E+00
F13	5.93E+00	3.05E+00	5.88E+05	5.08E+05	6.08E+00	3.00E+00	4.86E+00	2.87E+00	**1.01E+00**	1.87E+00
F14	1.56E+00	1.03E+00	6.62E+02	5.45E+02	1.50E+00	1.21E+00	1.89E+00	1.53E+00	**3.98E-02**	1.95E-01
F15	1.25E+00	1.11E+00	5.23E+03	2.85E+03	8.91E-01	7.11E-01	1.03E+00	8.15E-01	**7.98E-05**	2.24E-04
F16	2.23E+00	6.04E+00	2.84E+02	7.59E+01	8.73E-01	1.55E+00	6.20E-01	2.03E-01	**3.03E-02**	8.30E-02
F17	2.64E+00	2.56E+00	2.16E+02	4.24E+01	3.37E+00	3.71E+00	1.87E+00	3.23E+00	**1.22E-02**	6.12E-02
F18	9.73E-01	9.60E-01	7.32E+05	8.76E+05	1.37E+00	9.04E-01	1.13E+00	1.20E+00	**1.50E-03**	6.63E-03
F19	2.27E-01	4.24E-01	2.41E+04	2.03E+04	1.94E-01	3.03E-01	2.83E-01	5.12E-01	**9.56E-04**	4.02E-03
F20	6.73E-01	7.32E-01	1.88E+02	3.50E+01	1.07E+00	1.20E+00	5.54E-01	6.43E-01	**0.00E+00**	0.00E+00
F21	1.37E+02	5.01E+01	1.42E+02	1.50E+01	1.19E+02	4.11E+01	1.38E+02	5.11E+01	**1.00E+02**	0.00E+00
F22	9.34E+01	1.93E+01	7.17E+02	2.10E+02	9.01E+01	2.24E+01	9.36E+01	2.07E+01	**3.92E+01**	3.86E+01
F23	3.08E+02	2.45E+00	3.65E+02	8.53E+00	3.07E+02	2.53E+00	3.07E+02	3.26E+00	**2.97E+02**	4.24E+01
F24	3.22E+02	5.61E+01	3.94E+02	1.45E+01	2.98E+02	9.25E+01	2.88E+02	9.39E+01	**1.23E+02**	7.27E+01
F25	4.19E+02	2.30E+01	7.01E+02	6.99E+01	4.09E+02	1.88E+01	4.17E+02	2.23E+01	**3.76E+02**	8.19E+01
F26	3.00E+02	2.00E+01	7.74E+02	9.84E+01	3.00E+02	0.00E+00	3.00E+02	3.12E+01	**2.84E+02**	6.12E+01
F27	3.92E+02	2.71E+00	4.07E+02	2.09E+00	3.91E+02	1.71E+00	3.94E+02	2.28E+00	**3.89E+02**	8.07E-01
F28	3.16E+02	3.64E+01	5.30E+02	3.53E+01	3.04E+02	2.09E+01	3.30E+02	1.02E+02	**3.00E+02**	0.00E+00
F29	2.50E+02	6.95E+00	3.89E+02	4.32E+01	2.44E+02	7.42E+00	2.49E+02	1.02E+01	**2.31E+02**	1.82E+00
F30	1.20E+03	9.55E+02	9.02E+05	3.10E+05	1.46E+03	1.33E+03	1.09E+03	9.10E+02	**4.08E+02**	2.74E+01

TABLE II. RANKINGS OBTAINED THROUGH WILCOXON'S TEST

DKBSA vs	R+	R-	+	−	≈	Z	p-value	α=0. 05	α=0. 1
BSA	3.51E+02	0.00E+00	26	0	3	-4.458	8.29E-06	yes	yes
COBSA	4.35E+02	0.00E+00	29	0	0	-4.703	2.56E-06	yes	yes
BGBSA	3.78E+02	0.00E+00	27	0	2	-4.541	5.61E-06	yes	yes
BSAISA	4.06E+02	0.00E+00	28	0	1	-4.623	3.79E-06	yes	yes

V. CONCLUSION AND FUTURE WORK

In this paper，a backtracking search algorithm based on knowledge of different populations (DKBSA) is proposed. In DKBSA, the sub-population strategy based on decision space classification is applied to improve the local search ability of the DKBSA and maintain the diversity of population. A mutation operator with rotation invariance and knowledge is designed based on the historical and optimal information of sub-population, which effectively solves the problem of indivisibility of variables in CEC2017. The generation strategy of dominant individuals improves the evolutionary ability of the next generation. It will increase the complexity of the algorithm to divide a new sub-population at the completion of each iteration. Meanwhile, in certain complex functions, the evaluation times of the function do not decrease significantly. Using the surrogate models to predict the fitness function value to reduce the number of evaluations of the function is the focus of our future research.

ACKNOWLEDGMENT

This work was financially supported by the National Natural Science Foundation of China under grant numbers 62063021 and 61873328. It was also supported by the Key Research Programs of Science and Technology Commission Foundation of Gansu Province (2017GS10817), Lanzhou Science Bureau project (2018-rc-98), Public Welfare Project of Zhejiang Natural Science Foundation (LGJ19E050001), Wenzhou Public Welfare Science and Technology project (G20170016), respectively.

REFERENCES

1. Zhao, F.Q., et al., A discrete Water Wave Optimization algorithm for no-wait flow shop scheduling problem. Expert Systems with Applications, 2018. 91: p. 347-363.
2. De Falco, I., et al., Differential Evolution as a viable tool for satellite image registration. Applied Soft Computing, 2008. 8(4): p. 1453-1462.
3. Najkar, N., F. Razzazi, and H. Sameti, A novel approach to HMM-based speech recognition systems using particle swarm optimization. Mathematical and Computer Modelling, 2010. 52(11-12): p. 1910-1920.
4. Deep, K., et al., A real coded genetic algorithm for solving integer and mixed integer optimization problems. Applied Mathematics and Computation, 2009. 212(2): p. 505-518.
5. Zhan, Z.H., et al., Adaptive Particle Swarm Optimization. Ieee Transactions on Systems Man and Cybernetics Part B-Cybernetics, 2009. 39(6): p. 1362-1381.
6. Storn, R. and K. Price, Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. Journal of Global Optimization. 11(4): p. 341-359.
7. Akay, B. and D. Karaboga, Artificial bee colony algorithm for large-scale problems and engineering design optimization. Journal of Intelligent Manufacturing, 2012. 23(4): p. 1001-1014.
8. Bhattacharya, A. and P.K. Chattopadhyay, Biogeography-Based Optimization for Different Economic Load Dispatch Problems. Ieee Transactions on Power Systems, 2010. 25(2): p. 1064-1077.
9. Shi, Y.H., Brain Storm Optimization Algorithm, in Advances in Swarm Intelligence, Pt I, Y. Tan, et al., Editors. 2011. p. 303-309.
10. Civicioglu, P., Backtracking Search Optimization Algorithm for numerical optimization problems. Applied Mathematics and Computation, 2013. 219(15): p. 8121-8144.
11. Precup, R.E., et al., Backtracking Search Optimization Algorithm-Based Approach to PID Controller Tuning for Torque Motor Systems, in 2015 9th Annual Ieee International Systems Conference. 2015. p. 127-132.
12. Ayan, K. and U. Kilic, Optimal power flow of two-terminal HVDC systems using backtracking search algorithm. International Journal of Electrical Power & Energy Systems, 2016. 78: p. 326-335.
13. El-Fergany, A., Optimal allocation of multi-type distributed generators using backtracking search optimization algorithm. International Journal of Electrical Power & Energy Systems, 2015. 64: p. 1197-1205.
14. El Maani, R., B. Radi, and A. El Hami, Multiobjective backtracking search algorithm: application to FSI. Structural and Multidisciplinary Optimization, 2019. 59(1): p. 131-151.
15. Shaheen, A.M., R.A. El-Sehiemy, and S.M. Farrag, Integrated Strategies of Backtracking Search Optimizer for Solving Reactive Power Dispatch Problem. Ieee Systems Journal, 2018. 12(1): p. 424-433.
16. Abdolrasol, M.G.M., et al., An Optimal Scheduling Controller for Virtual Power Plant and Microgrid Integration Using the Binary Backtracking Search Algorithm. Ieee Transactions on Industry Applications, 2018. 54(3): p. 2834-2844.
17. Chen, D., et al., Learning backtracking search optimisation algorithm and its application. Information Sciences, 2017. 376: p. 71-94.
18. Ahandani, M.A., A.R. Ghiasi, and H. Kharrati, Parameter identification of chaotic systems using a shuffled backtracking search optimization algorithm. Soft Computing, 2018. 22(24): p. 8317-8339.
19. Zhao, W., et al., Best Guided Backtracking Search Algorithm for Numerical Optimization Problems, in Knowledge Science, Engineering and Management, Ksem 2016, F. Lehner and N. Fteimi, Editors. 2016. p. 414-425.
20. Zhao, L., et al., Improved Backtracking Search Algorithm Based on Population Control Factor and Optimal Learning Strategy. Mathematical Problems in Engineering, 2017. 2017.
21. Wang, H., et al., Modified Backtracking Search Optimization Algorithm Inspired by Simulated Annealing for Constrained Engineering Optimization Problems. Computational Intelligence and Neuroscience. 2018: p. 1-27.