Tumbleweed Optimization Algorithm and Its Application in Vehicle Path Planning in Smart City

Jeng-Shyang Pan[1,2], Qingyong Yang[1], Chin-Shiuh Shieh[3], Shu-Chuan Chu[1*]

[1] College of Computer Science and Engineering, Shandong University of Science and Technology, China
[2] Department of Information Management, Chaoyang University of Technology, Taiwan
[3] Department of Electronic Engineering, National Kaohsiung University of Science and Technology, Taiwan
jspan@cc.kuas.edu.tw, yang_qy@sdust.edu.cn, csshieh@nkust.edu.tw, scchu0803@gmail.com

Abstract

With the increasing complexity of optimization problems, the requirements for algorithm optimization capabilities are getting higher and higher. In order to better solve complex optimization problems, this paper proposes a new swarm intelligence optimization algorithm named Tumbleweed Optimization Algorithm (TOA). The TOA algorithm consists of two stages, which simulate the seedling growth phase and seed propagation phase of tumbleweed respectively. The TOA algorithm adopts a multi-group structure to improve the global searching ability of the algorithm. In order to verify the performance of the TOA algorithm in numerical optimization and solving practical application problems, this paper selects the CEC2013 benchmark function library and the vehicle path planning in the smart city for testing. Through the comparison of experimental results, the TOA algorithm can both show strong optimization capabilities. Compared with the other ten intelligent optimization algorithms, the TOA algorithm proposed in this paper can also show strong competitiveness.

Keywords: Tumbleweed optimization algorithm, Swarm intelligence, Multi-group structure, Vehicle path planning

1 Introduction

In the actual production and life process, various "optimal" problems are often encountered [1]. Such as maximizing the utilization of raw materials, the shortest driving distance, the highest production efficiency. Therefore, the "optimal" problem can be understood as selecting an optimal solution from many or even countless feasible solutions. Or it can also be understood as spending the least cost and achieving the best results. With the continuous development of human society, the scale and complexity of the "optimal" problem are becoming larger and more complex. When traditional optimization methods deal with these problems, the cost increases dramatically and becomes unbearable. Therefore, in the face of complex optimization problems, how to design an efficient solution algorithm has become an urgent problem to be solved.

Inspired by many biological, physical and chemical phenomena in nature, scholars began to explore the internal relationship between these natural evolution phenomena and solving optimization problems. Since 1980, a series of intelligent optimization algorithms have emerged to solve the "optimal" problem by simulating various biological phenomena in nature [2]. Representative methods include genetic algorithm (GA) [3], artificial neural network (ANN) [4], simulated annealing (SA) [5], tabu search (TS) [6], particle swarm optimization (PSO) [7] and ant colony optimization (ACO) [8]. When solving combinatorial optimization problems, these algorithms are often called meta-heuristic algorithms [9]. Compared with traditional optimization methods, intelligent optimization algorithms have the characteristics of self-learning, self-organization, self-adaptation and easy parallelization. Theoretically, the optimal solution or approximate optimal solution of the problem can be found in a reasonable time. In solving complex optimization problems, intelligent optimization algorithms have incomparable advantages over traditional optimization methods. At present, intelligent optimization algorithms are widely used to solve optimization problems in various fields, such as engineering optimization [10-11], intelligent scheduling [12-13], image processing [14-15], wireless sensor networks [16-18], path optimization [19-20], data prediction [21-22] and so on [23-24].

According to the number of agents included in the algorithm, intelligent optimization algorithm can be divided into individual based intelligent optimization algorithm and population-based intelligent optimization algorithm. The intelligent optimization algorithm based on population is also called swarm intelligence (SI) optimization algorithm. The PSO algorithm [7] and the ACO algorithm [8] mentioned above are typical examples. The SI algorithms simulate the group behavior of insects, herds, fish, birds and plants in nature [25]. These groups look for food or suitable living environment through cooperation. Each individual guides its activity behavior by learning the experience of itself, surrounding individuals or the whole population. Moreover, the SI algorithms have the characteristics of simple implementation, high scalability and strong adaptability. The research on swarm intelligence optimization algorithm has become a hot field in the research of intelligent optimization algorithm.

After years of research and development, more and more novel swarm intelligence optimization algorithms have been proposed. And they are applied to solve complex optimization problems in different fields [26-27]. Such as differential evolution (DE) algorithm [28], shuffled frog leaping algorithm (SFLA) [29-30], whale optimization algorithm (WOA) [31], grey wolf optimizer (GWO) [32], cat swarm optimization

*Corresponding Author: Shu-Chuan Chu; E-mail: scchu0803@gmail.com
DOI: 10.53106/160792642022092305002

(CSO) [33], fish migration optimization (FMO) [34], bat algorithm (BA) [35], quasi-affine transformation evolution (QUATRE) algorithm [36], butterfly optimization algorithm (BOA) [37], equilibrium optimizer (EO) [38], jellyfish search (JS) optimizer [39], sparrow search algorithm (SSA) [40], seagull optimization algorithm (SOA) [41], manta ray foraging optimization (MRFO) [42], etc.

Although there are already many swarm intelligence optimization algorithms, as explained in the NFL theorem [43], no algorithm is omnipotent. Each intelligent optimization algorithm has its shortcomings in some respects and cannot effectively solve all optimization problems. Therefore, more and more improved algorithms and new algorithms are proposed. Inspired by the habits of tumbleweed plants in nature, this paper proposes the tumbleweed optimization algorithm (TOA), a novel swarm intelligence optimization algorithm. The TOA algorithm includes two stages, which simulate the seedling growth stage and seed propagation stage of tumbleweed respectively. The seedling growth stage corresponds to the exploitation stage of the optimization algorithm, that is, the local search stage. The seed propagation stage corresponds to the exploration stage, that is, the global exploration stage. By judging the growth algebra of the individual, the switch of two stages is realized. The multi-group structure is adopted in the TOA algorithm. Each sub-population contains several tumbleweed individuals. In the growth process, according to the specific topology, the adjacent sub groups will affect each other. Since the spread seeds will be scattered in different areas, it is necessary to re-divide the sub groups. In the way of division, the TOA algorithm adopts k-means algorithm [44] based on individual position.

In order to verify the optimal performance of the TOA algorithm, the experiment in this paper consists of two parts. The first part is tested with the CEC2013 [45] benchmark library. In this part of the experiment, this paper selects three different dimensions of 30D, 50D and 100D to test. The comparison algorithm includes three classical optimization algorithms including the GA, DE and PSO algorithms, as well as seven new swarm intelligence optimization algorithms, which are proposed in recent years, including the WOA, GWO, JS, BOA, SSA, MRFO and SOA algorithms. The other part of the experiment is the vehicle path planning in smart city. With the continuous development of science and technology, the construction of smart city has gradually become a reality. The realization of smart city functions is inseparable from the association of various internet of thing (IOT) devices and the analysis of data [46-47]. Due to the increase of IOT devices, the pressure on base stations to store and forward data is also increasing. The pressure of base station can be greatly alleviated by using data collection vehicle to collect base station data regularly. Therefore, how to optimize the path of data collection vehicle and maximize the amount of path data collection will be a practical application problem to verify the effectiveness and feasibility of the proposed TOA algorithm in solving practical application problems. Finally, through the analysis of two parts of the experimental data, the TOA algorithm can show good optimization performance and strong competitiveness.

The remaining sections are arranged as follows: the Section 2 introduces the inspiration of the algorithm proposed in this paper. The optimization principle and process of the TOA algorithm will be introduced in the Section 3. The Section 4 is the numerical optimization test of the TOA algorithm. The mathematical model and test of the path planning of data collection vehicle will be stated in Section 5. Finally, the Section 6 is the summary of this paper and the future research work.

2 Inspiration

Tumbleweed, also known as prickly russian thistle, is a desert plant with strong vitality, which is commonly found in the Gobi and desert [48]. The growth process of tumbleweed at the seedling stage is similar to that of other plants. Depending on where the seeds are scattered, the seedlings tend to grow in small groups. The seedlings will be affected by the surrounding environment during the growth process. The influence of the environment comes from many aspects. On the one hand, it comes from the influence of other individuals in the same group or the individuals in another group. On the other hand, it comes from the influence of natural environmental factors, such as light, soil, and moisture. But unlike other plants, tumbleweeds are not static in adulthood. When the drought comes, the whole tumbleweed will retract its roots, move randomly with the wind, and spread its seeds in the process of moving. When a suitable environment is found, the roots will continue to penetrate deep into the soil to absorb water.

An intelligent optimization algorithm needs to include a certain number of agents. And the optimization process is divided into two stages: global exploration and local exploitation [49]. By analogy with the habits of tumbleweeds, each tumbleweed individual corresponds to a particle in an intelligent optimization algorithm. Tumbleweeds at the seedling stage are stationary and grow by absorbing nutrients from their surroundings. This stage can be considered as the development of the current environment, corresponding to the local exploitation stage in the intelligent optimization algorithm. Adult tumbleweed is a dynamic process of moving with the wind. This stage can be considered as the exploration of the whole search space, corresponding to the global exploration stage.

In order to solve the optimization problem, the mathematical modeling and formula expression of this biological phenomenon of tumbleweed are carried out in this paper. The specific content of tumbleweed optimization algorithm is introduced in Section 3.

3 Tumbleweed Optimization Algorithm

In this section, the optimization principle and specific process of tumbleweed optimization algorithm (TOA) are mainly described. In order to facilitate the subsequent description of the algorithm process, the relevant mathematical symbols and their meanings used in the TOA algorithm are listed in Table 1.

Table 1. Description of relevant symbols in the TOA algorithm

Symbol	Meaning
Ω	Solution space
G	Subpopulation structure
ps_g	The number of individuals in the subpopulation
X	Individual representation in subpopulation
fit	Individual fitness
pbest	The best individual in the subpopulation
pbest_val	Optimal individual fitness in subpopulation
gbest	Global optimal individual
gbest_val	Global optimal individual fitness
grow_iter	Growth algebra
grow_cycle	Growth cycle
K	Maximum number of subpopulations

3.1 Algorithm Structure

The TOA algorithm proposed in this paper uses a multi-group structure. The multi-group structure includes multiple sub-populations in the algorithm, and each sub-population contains several individuals and corresponds to a sub-region in the search space. The structure of each sub-population is as follows:

$$G = < X, fit(X), pbest, pbest\_val, ps\_g > \tag{1}$$

The optimization process of the TOA algorithm is divided into two stages, the first stage is the seedling growth stage, and the second stage is the seed propagation stage. The two stages correspond to the local exploitation and global exploration stages of the intelligent optimization algorithm, and each account for half of the tumbleweed growth cycle. In the process of seed dispersal, there is no guarantee that seeds from individuals in the same sub-population will remain in the same sub-population. Therefore, k-means clustering algorithm is adopted in this paper to achieve the re-division of population individuals after each growth cycle.

The overall architecture of the TOA algorithm is shown in Figure 1.

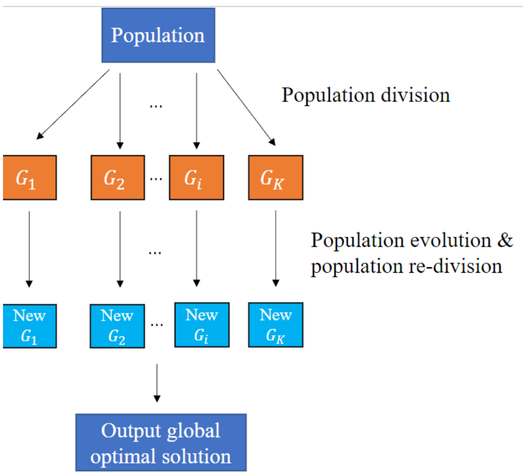


Figure 1. The algorithm architecture of the TOA algorithm

3.2 Seedling Growth Stage

Like other plants, tumbleweed is affected by the surrounding environment during the seedling growth stage. Influencing factors include many aspects, such as biological factors that compete with surrounding individuals, as well as natural factors such as soil, light, and moisture. In the TOA algorithm, the fitness of each individual is used as an index to evaluate the individual's adaptability to the environment. The individual's adaptability to the environment is represented by the symbol P_k^i, and the calculation formula is shown in Equation (2).

$$P_k^i = \frac{fit(X_k^i)}{sum(fit(X_k)) + \xi} \tag{2}$$

where, ξ is a very small number, which can be ignored. It should be noted that when solving the minimization problem, the fitness needs to be inverted. The greater the P value, the higher the individual's adaptability to the current environment. There will be more opportunities to learn and communicate with external groups, but they are also greatly affected by the outside world. Individuals with a lower P value are compared with the current environment based on their adaptability. If its adaptability can meet the current environment, it can communicate with the outside world. Otherwise, it can only communicate with individuals within the group. In the TOA algorithm, individuals who communicate outside or only within the group are divided by sorting the fitness of individuals in the group. The top half of the individuals have the ability to communicate with the outside world by default.

Suppose the distribution of the groups sorted according to the optimal value of the sub-population in the space is shown in Figure 2. Each sub-population will have an impact on other sub-populations within a certain range. In the TOA algorithm, we assume that each sub-population can be affected by at most the other two sub-populations. Referring to Figure 2, the influence on the growth of individuals in group G_1 comes from two aspects. On the one hand, it comes from the influence of the individual $gbest$. On the other hand, it comes from the influence of the individual $pbest$ in group G_2. The influencing factors in the growth of individuals in the group G_2 come from group G_1 and G_3. The group G_3 is similar to the group G_2, and the influencing factors come from group G_3 and G_4. The group G_4 is only affected by the group G_3. Of course, in addition to external factors, the individual $pbest$ in the group will also affect the growth process of other individuals in the group. Equation (3) shows the mathematical expression of the influencing factors of this part.

$$\text{Factor} = \begin{cases} \dfrac{c1*\left(gbest - X_k^i\right) + c2*\left(pbest_k - X_k^i\right) + c3*(pbest_{k+1} - X_k^i)}{3}, if\ k == 1 \\ \dfrac{c1*\left(pbest_k - X_k^i\right) + c2*(pbest_{k-1} - X_k^i)}{2}, elif\ k == K \\ \dfrac{c1*\left(pbest_{k-1} - X_k^i\right) + c2*\left(pbest_k - X_k^i\right) + c3*(pbest_{k+1} - X_k^i)}{3}, else \end{cases} \tag{3}$$

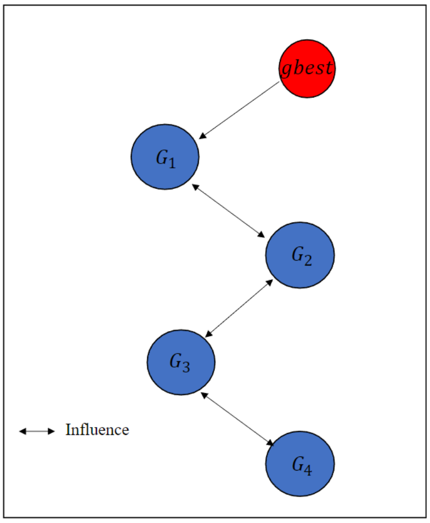


Figure 2. Spatial distribution map of sub-populations

The evolution formula for individuals who can communicate with the outside world is shown in Equation (4).

$$\mathrm{X}_{new,k}^{i} = X_k^i + r_1 * Factor \tag{4}$$

For individuals who only communicate within groups, that is, the adaptability of individuals cannot meet the current living environment. The influencing factors come from $pbest$ and other individuals in the current group. In the selection of other individuals, this paper uses roulette-wheel [50] to choose. The evolution formula is shown in Equation (5).

$$\mathrm{X}_{new,k}^{i} = \mathrm{X}_k^i + r_1 * (c4 * (pbest_k - X_k^i) + c5 * (X_{select,k}^i - X_k^i)) \tag{5}$$

In the above formula, $c1$, $c2$ and $c3$ are random numbers in interval [0, 2], $c4$ and $c5$ are random numbers in interval [0, 1]. The parameter r_1 is the influence of the current environment on the individual, and its value decreases linearly from the pre-set value t to 0 with the algorithm iteration process. Equation (6) gives the iterative formula of the parameter r_1. It indicates that with the continuous growth of tumbleweed seedlings, the influence of the surrounding environment on them is gradually reduced. In the TOA algorithm, t is set to 2 by default.

$$\mathrm{r}_1 = t * \left(1 - \frac{grow\_iter}{grow\_cycle}\right) \tag{6}$$

Finally, the evolution formula of seedling growth stage is shown in Equation (7).

$$\mathrm{X}_{new,k}^{i} = \begin{cases} Equation\ (4), if\ rank(i) < \frac{ps\_g}{2}\ and\ rand < P_k^i \\ Equation\ (5), else \end{cases} \tag{7}$$

3.3 Seed Propagation Stage

The adult tumbleweed will uproot its roots, move with the wind, and spread the seeds as it moves. According to the literature [51], the propagation distance of seeds can be calculated by Equation (8).

$$\mathrm{D} = \frac{H*U}{F} \tag{8}$$

where D is the propagation distance of the seed, H is the release height of the seed, F is the sedimentation rate of the seed, and U is the wind speed. For a specific plant, both H and F are constants.

This paper integrates Equation (8) with the iterative process of the optimization algorithm. That is, the parameter H corresponds to the difference between the growth cycle and the current growth generation. The parameter F corresponds to the growth cycle. Then the update formula at this stage is shown in Equation (9).

$$\mathrm{X}_{new,k}^{i} = pbest_k + V * \frac{grow\_cycle - grow\_iter}{grow\_cycle} \tag{9}$$

where, V is a uniformly distributed random number in the interval [lb, ub].

Algorithm A1 is the pseudo-code of the whole optimization process of the TOA algorithm.

4 Numerical Experimental Analysis

In order to verify the optimization ability of the proposed TOA algorithm in solving numerical optimization problems, this paper selects the CEC2013 benchmark function library for testing. It contains two parts in comparison with other intelligent optimization algorithms. One part is a comparison with classical algorithms such as the GA algorithm, the DE algorithm and the PSO algorithm. The other part is a comparison with some popular intelligent optimization algorithms proposed in recent years. All experiments are run on the same computer. The operating system is Windows 10, the CPU frequency is 3.0GHz, the memory is 8G, and the Matlab version is R2019b. In the process of testing, the maximum number of individuals of each algorithm is 30. The maximum number of evaluations for each individual is 5000 times. Each algorithm is tested continuously for 30 times, and the test results are recorded.

4.1 The Selection of *K* Value

Before comparing with other algorithms, this paper first makes an experimental comparison of TOA algorithm under different K values. Here, we constrain the value of K. According to the algorithm process described in the previous section, the minimum value of K needs to be 3 to ensure the normal operation of the algorithm. For the maximum value of K, we suggest that it is best not to exceed $floor(ps/5)$. This is because when the k-means algorithm is used to re-divide the population, the more groups, the more probability there is only one particle in each group will be increased, and some formulas in the TOA algorithm will lose their significance. It will also lead to the larger amount of memory occupied by the algorithm, resulting in slowing down the algorithm running speed. During the experiment, since the maximum number of particles ps is set to 30, the maximum value of K is 6. In order to verify the optimal performance of the TOA algorithm under different K values, this paper tests different dimensions of 30D, 50D and 100D on the CEC2013 benchmark functions. Table 2 to Table 4 records the test results of the TOA algorithm. Figure 3 to Figure 5 counts the number of wins on the 28 benchmark functions when K takes different values.

Algorithm 1. Tumbleweed optimization algorithm

Input: *f*(*x*): objective function; *ps*: population size; *K*: maximum number of subpopulations; *dim*: problem dimension; *gc*: growth cycle; *Max_gen*: maximum number of iterations

Output: optimal *gbest* and optimal value $f^*(gbest)$.

Initialize population *X*;
The population *X* is divided into *K* groups using the k-means method;
repeat
    grow_iter – mod(gen, *gc*);
    for *k* = 1 to *K* **do**
        if *grow_iter* < *gc*/2 **then**
            Calculate the adaptability P_k^i of each individual using Eq.2;
            Update r_1 using Eq.6;
            if rank(i) < $\frac{ps-g}{2}$ && rand < P_k^i **then**
                Calculate $X_{new,k}^i$ using Eq.4;
            else
                Use roulette-wheel to select an individual;
                Calculate $X_{new,k}^i$ using Eq.5;
            end if
        else
            Calculate $X_{new,k}$ using Eq.9;
        end if
        Boundary detection;
        Calculate the fitness of each individual in group G_k;
        Update *pbest* and *pbest_val*;
        Update *gbest* and *gbest_val*;
    end for
    if *grow_iter* == *gc* – 1 **then**
        Re-divide the population into *K* groups using the k-means method;
    end if
until (*gen* < *Max_gen* or Satisfy convergence constraints)

Table 2. The comparison of the TOA algorithm with different *K* values on 30D

Function	K = 3		K = 4		K = 5		K = 6	
	mean	std	mean	std	mean	std	mean	std
F1	0.096144	0.031638	0.276008	0.057915	0.404493	0.0883	0.41858	0.074997
F2	3067755	1293879	3100794	1564173	3119674	1570796	3015363	1195054
F3	1.38E+08	1.2E+08	1.47E+08	2.25E+08	1.09E+08	1.19E+08	1.66E+08	1.85E+08
F4	15959.21	4150.744	28207.08	8174.474	39105.38	12924.39	42377.83	9415.982
F5	0.134279	0.023901	0.281176	0.049889	0.365343	0.060176	0.3723	0.061835
F6	42.83187	27.56389	33.61487	23.28851	42.12815	28.59266	45.51614	27.12016
F7	71.07496	22.56875	60.62711	30.96051	42.71541	25.29023	31.23716	19.02553
F8	20.96944	0.050252	20.96191	0.057907	20.98514	0.054497	20.96192	0.059769
F9	19.91767	3.752046	21.80054	3.46938	19.38749	3.333257	19.88533	3.922631
F10	1.135956	0.07088	1.210449	0.108459	1.224013	0.076668	1.277206	0.121893
F11	80.60107	43.60489	62.02387	23.83	55.41564	17.32591	48.90504	13.96927
F12	92.14382	31.17321	71.23077	24.68128	58.13371	38.72918	83.84221	49.57836
F13	118.4556	41.44931	129.9995	38.26977	139.8033	39.41781	144.9187	27.53465
F14	1854.124	487.4749	1800.722	367.1469	1880.008	758.3335	2207.487	1249.349
F15	4416.567	1653.901	5718.174	1223.975	6133.255	391.928	6161.919	429.1614
F16	2.576708	0.312135	2.638239	0.324074	2.548155	0.39027	2.694517	0.283311
F17	191.3974	16.85131	199.4993	13.70573	201.2602	13.10772	198.137	12.21287
F18	217.9314	9.769594	211.8704	11.4511	210.2548	12.95397	213.3991	10.69973
F19	10.9281	1.515602	12.24005	1.099761	12.71881	1.341417	12.81897	0.950398
F20	12.64727	1.138753	12.27071	0.619029	12.24363	0.412427	12.17	0.38342
F21	313.4878	83.51404	319.342	79.63762	337.334	84.03659	320.9395	79.01058
F22	2123.634	654.0388	1816.455	455.8738	1730.149	786.0082	1366.033	465.016
F23	3814.234	1254.59	5405.479	1390.501	6097.164	796.1703	6360.263	302.8975
F24	273.8766	8.926126	264.6838	9.808036	262.1296	7.504067	261.2456	8.595267
F25	282.2473	9.524555	277.0566	6.69639	269.9496	8.294088	266.0621	8.018754
F26	347.2767	40.85647	337.9296	47.31488	334.5717	46.17868	302.271	68.12984
F27	892.9098	98.23928	857.8346	78.98519	838.4401	89.54661	815.682	79.37482
F28	389.1508	309.1235	543.9761	549.0281	421.3195	339.2462	392.048	279.4487

Table 3. The comparison of the TOA algorithm with different *K* values on 50D

Function	K = 3		K = 4		K = 5		K = 6	
	mean	std	mean	std	mean	std	mean	std
F1	0.63242	0.158301	1.844837	0.272823	2.419604	0.420692	2.545371	0.423594
F2	10713038	4201223	9574927	2454249	8405185	3371731	9045569	3866231
F3	1.46E+09	1.16E+09	1.41E+09	1.27E+09	1.12E+09	1.03E+09	1.23E+09	1.05E+09
F4	49061.03	9700.018	69348.99	19194.6	85947.18	18115.12	86188.71	15346.32
F5	0.622664	0.110201	1.100688	0.150874	1.312444	0.169935	1.392056	0.228668
F6	62.55962	25.01869	62.29403	28.67781	66.77283	34.50577	64.27811	30.94774
F7	119.5093	27.47312	99.36241	23.69478	79.52415	19.56601	72.35749	20.46601
F8	21.16217	0.035182	21.1741	0.034261	21.17288	0.026717	21.1694	0.034013
F9	44.49343	4.873374	42.46374	6.554176	44.83882	8.672894	44.68382	8.928748
F10	3.555221	0.877774	3.915795	1.102638	4.189038	1.065539	4.421568	1.132691
F11	158.1064	34.24508	180.5887	58.20192	156.5413	40.77313	139.8908	27.48833
F12	321.2626	139.7587	217.7524	71.16986	187.8925	90.79656	255.1405	111.8631
F13	354.6633	56.09104	357.0827	32.4481	358.2374	22.47033	354.8136	27.23791
F14	3744.468	843.3797	3622.209	774.8077	4050.258	2204.117	4738.309	2666.25
F15	11316.21	2530.887	12528.9	964.2468	12924.31	486.5203	12776.65	474.0312
F16	3.592678	0.349183	3.561117	0.433072	3.61176	0.444092	3.723119	0.340352
F17	403.3359	17.90593	403.6531	21.11961	403.212	21.1225	400.6012	21.02536
F18	442.2797	17.12376	433.3843	13.11045	418.6148	15.0859	424.5426	15.43264
F19	25.93426	2.700165	27.41984	1.971856	27.62178	1.663883	27.47743	1.741964
F20	22.15482	0.401716	22.02722	0.348817	22.30325	0.273764	22.14621	0.249159
F21	893.0278	361.4853	776.2925	387.6969	911.6565	368.2556	754.5459	402.9422
F22	4870.086	889.4075	4212.007	720.8297	3864.382	699.6814	4365.26	1578.757
F23	10331.79	2832.575	12493.69	1066.011	12682.51	459.2605	12852.77	494.529
F24	339.7667	11.91717	328.8694	10.56459	314.6382	11.61081	315.3647	10.62349
F25	362.6314	11.24134	351.9789	10.85247	338.4745	11.0536	334.1057	12.79486
F26	421.6564	12.28735	410.1403	12.16856	391.8521	37.66203	390.804	38.21663
F27	1545.562	141.5493	1454.324	107.2984	1339.917	139.307	1353.446	111.013
F28	1697.873	1611.931	1484.343	1549.585	1923.036	1651.005	1851.478	1685.664

Table 4. The comparison of the TOA algorithm with different K values on 100D

Function	K = 3		K = 4		K = 5		K = 6	
	mean	std	mean	std	mean	std	mean	std
F1	6.616276	1.076467	16.32805	2.312258	21.00996	3.04936	23.65251	3.424116
F2	50293628	15130891	51632909	13732257	52187198	11830649	45181707	10220826
F3	1.84E+10	6.94E+09	1.71E+10	7.37E+09	1.85E+10	9.72E+09	1.69E+10	7.82E+09
F4	143960.7	23557.11	193404.8	31619.13	210640.1	23223.85	243493.9	24350.6
F5	6.033462	1.391409	10.51364	1.853699	13.12416	2.18062	14.83327	2.707704
F6	312.2888	50.84528	317.8085	43.41032	344.2609	48.20632	349.8414	69.71744
F7	194.2171	23.578	176.4001	27.92209	152.0233	29.98462	140.2535	20.84657
F8	21.29226	0.040594	21.29205	0.036094	21.30712	0.033867	21.31208	0.029554
F9	110.7328	11.27031	115.1592	14.1134	115.5936	14.40197	115.6692	17.51954
F10	53.26273	14.15987	62.41185	16.75785	66.33938	18.16628	67.63841	16.36318
F11	787.6178	149.9097	728.9065	116.8229	658.37	134.4109	607.915	75.25374
F12	1052.646	333.0881	735.4677	162.8394	614.0463	218.588	621.529	202.238
F13	1025.473	59.71336	957.6716	51.69219	933.3907	47.90764	922.795	37.76413
F14	12222.25	1555.31	12290.15	1850.717	17140.31	7589.048	22540.06	7580.966
F15	28217.36	3263.074	29186.79	488.4754	29287.04	617.603	29336.69	537.1106
F16	4.350836	0.27027	4.431802	0.189435	4.392862	0.216579	4.375448	0.219281
F17	1035.532	52.4735	1018.092	36.52744	1006.721	38.20327	991.939	20.94016
F18	1100.239	33.16431	1077.23	34.0352	1045.594	27.5264	1033.107	28.69193
F19	75.50668	4.514959	73.39763	4.119254	72.99276	4.002279	71.77608	3.893688
F20	50	4.59E-10	50	1.42E-11	50	0	50	1.72E-12
F21	619.0414	132.8384	542.5915	149.6047	571.299	151.2571	599.8269	156.9915
F22	13793.08	1619.26	13280.18	1218.49	13916.92	3351.274	19987.17	7253.811
F23	28065.65	4445.921	29820.02	610.6007	29798.52	574.341	29948.13	472.3997
F24	541.8713	24.09574	524.5773	25.03967	495.6027	17.21219	489.8223	17.03644
F25	582.3262	20.97765	554.0448	27.18328	547.8547	22.45242	531.1209	22.17413
F26	584.6752	25.62407	571.7645	21.16896	558.5504	18.20557	547.7483	16.29979
F27	3374.463	217.8345	3201.656	222.7551	3055.537	197.6016	2997.57	254.5037
F28	5387.74	2495.625	5371.55	2382.901	5469.859	2204.019	5910.61	2918.885

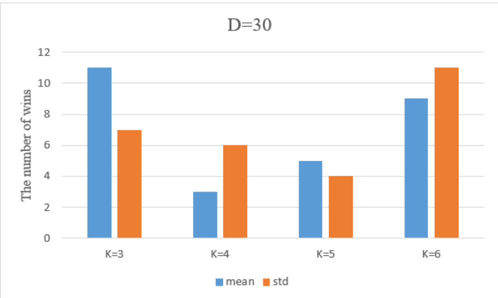


Figure 3. The number of wins under different K values on 30D

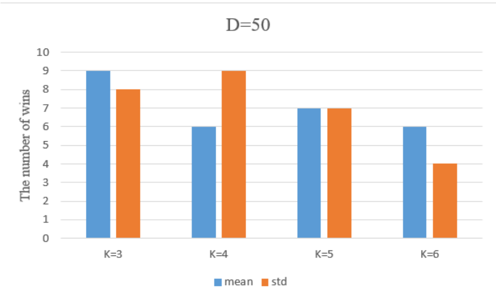


Figure 4. The number of wins under different K values on 50D

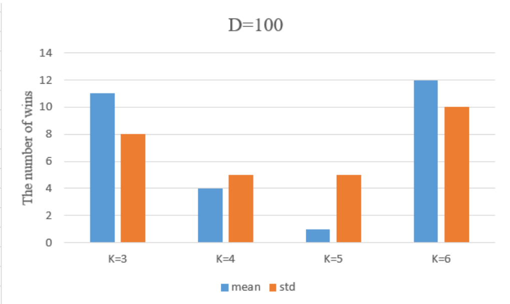


Figure 5. The number of wins under different K values on 100D

The average ranking is calculated by sorting the test results of the four values in each dimension. The following sorting results can be obtained:

mean: "$K = 3$", "$K = 6$", "$K = 5$", "$K = 4$".

std: "$K = 6$", "$K = 3$", "$K = 4$", $K = 5$".

In general, the optimization ability of the TOA algorithm with "$K = 3$" on the CEC2013 test set is the best among the four values. "$K = 6$" is weaker than "$K = 3$" in the mean, but better than "$K = 3$" in the standard deviation. The mean value reflects the accuracy of the algorithm, and the standard deviation reflects the stability of the algorithm. When the number of groups is 3, each group contains more particles, so the solution accuracy will be better than the other three values. But it is also very easy to fall into the local optimum, especially the problem to be solved is a multi-modal problem containing multiple local optimum solutions. Therefore, the results of each solution may vary greatly. Although the number of groups is 6, each group contains fewer particles, which may reduce the final solution accuracy of the algorithm. But the algorithm is easier to jump out of the local optimal solution, and stability will also bring certain benefits. Because the two values have different advantages. Therefore, in the subsequent comparison with other intelligent optimization algorithms, the value of the grouping number in the TOA algorithm will be set to "K =3" and "K =6", respectively. But by default, the TOA algorithm takes "K =3".

4.2 Comparison with Other Intelligent Optimization Algorithms

In this section, two TOA algorithms with different K values are compared with other ten intelligent optimization algorithms. The compared algorithms include three classical intelligent optimization algorithms including the GA, the DE and the PSO algorithms, as well as seven new swarm intelligent optimization algorithms proposed in recent years, including the WOA, the GWO, the JS, the BOA, the SSA, the MRFO and the SOA algorithms. Table 5 to Table 10 records the mean and standard deviation of these ten algorithms on 30D, 50D and 100D in the CEC2013 benchmark functions. The row win in the tables indicates the number of wins of the TOA algorithm compared with these ten algorithms.

The CEC2013 benchmark function set contains 28 test functions, which are divided into three function types. Among them, functions F1-F5 are unimodal functions, F6-F20 are multimodal functions, and F21-F28 are composite functions. In order to be able to visually observe the comparison of TOA algorithm with other algorithms in different function types, we make statistical processing on the results and records the relevant results in Table 11 to Table 14. The part of the TOA algorithm that performs poorly (the number of wins is less than half of the total number of test functions) is bolded.

It can be seen from the data in the table that the TOA algorithm with different K values performs better than other algorithms on the mean of three dimensions. And as the dimension increases, the TOA algorithm can still maintain its quantitative advantage. In the comparison of standard deviation, there is a clear gap between the two TOA algorithms and the DE algorithm. The two TOA algorithms are worse than the DE algorithm in three dimensions. However, compared with other algorithms, the TOA algorithm still has certain advantages.

By comparing each function type, the number of wins in the mean and standard deviation of the two TOA algorithms on the unimodal function type is better than that of the DE algorithm. But in the other two types of standard deviations, the TOA algorithm is weaker than the DE algorithm, which leads to weaker than the DE algorithm in the total number of standard deviation wins. In comparison with the JS algorithm, the total number of wins in the mean and standard deviation of the two TOA algorithms is better than that of the JS algorithm. In the comparison of 30D and 50D unimodal functions, the two TOA algorithms are worse than JS algorithm in mean and standard deviation. However, with the increase of dimension, the two TOA algorithms are better than the JS algorithm in the comparison of 100D unimodal functions. The two TOA algorithms are also weaker than the SSA algorithm and the MRFO algorithm in the performance of unimodal functions. Except for 30D, the TOA algorithm with "K = 3" is slightly better than the SSA algorithm in mean and standard deviation. The mean and standard deviation of other dimensions are worse than the SSA algorithm. Compared with the MRFO algorithm, the two TOA algorithms are worse than the MRFO algorithm in unimodal functions. In comparison with other intelligent optimization algorithms, except that the TOA algorithm with "K = 3" is weaker than the GWO algorithm in the mean value of 50D, the two TOA algorithms perform better in the comparison of the total number of wins and each function type.

Figure 6 shows the convergence curves of some test functions. It can be seen from the convergence curve that the TOA algorithm also has a better convergence performance. In the first 500 generations of the iterative process, except that the SOA algorithm has early stagnation on some test functions, the remaining algorithms are able to achieve the convergence optimization. In the later iterative optimization process, each algorithm has different evolution strategies, so different convergence accuracy will be achieved. Such as the WOA algorithm and the SOA algorithm, the convergence performance of the algorithm is greatly affected by the linearly decreasing parameter, so it is prone to premature convergence. Although the similar concept is also introduced in the TOA algorithm, the parameter is only related to the algebra of the growth cycle and is less affected by the iterative process. Therefore, even in the later stage of the algorithm, the TOA algorithm still has a strong global search capability, and can jump out of the local optimal solution in time, thereby achieving higher convergence accuracy. For example, in the function F23, the TOA algorithm has stagnated during the iteration process from 500 to 1000 generations. However, through the two-stage conversion, the TOA algorithm jumps out of the local optimal solution in time, and finally achieves better convergence accuracy.

4.3 The Exploration and Exploitation Conversion Process

According to the description of the TOA algorithm in the Section 3, the exploitation and exploration process of the algorithm is transformed and differentiated according to the growth cycle. Figure 7 shows the particles distribution and phase switching process in the solution process of the TOA algorithm. The test function used is the square function. In the TOA algorithm, the $grow\_cycle$ is set to 50. Therefore, the first 25 iterations are the seedling growth stage of the algorithm. At this stage, it needs to realize the convergence of the algorithm, so as to fully mine the information of the current region. From (a) to (b) of Figure 7, it can be seen that the algorithm has achieved convergence, and all particles are distributed near the optimal value. In the last 25 iterations, it is the seed propagation stage of the algorithm. The particles are updated in space according to Equation (9) to simulate the moving process of tumbleweeds. It can be observed that (c) to (d) in Figure 7, the distribution of particles is scattered. Through this stage, it can help the algorithm to mine new information and jump out of the local optimal solution in time. (e) to (f) in Figure 7 are the seedling growth stage of the new growth cycle. It can be seen that the particles gradually gather near the optimal value, that is, the algorithm realizes the mining of information in the previous stage. (g) in Figure 7 is the seed propagation stage of several growth cycles, and the distribution of particles is still scattered. It shows that the algorithm still has a strong global search capability and can still conduct new explorations in the current area.

Table 5. The experimental results of the GA, DE, PSO, WOA and GWO algorithms on 30D

Function	GA		DE		PSO		WOA		GWO	
	mean	std	mean	std	mean	std	mean	std	mean	std
F1	8.661649	4.345551	60.46126	189.6953	2714.171	2065.21	4.090751	1.360965	1385.978	1172.106
F2	24595181	9052123	46828774	14769298	58108170	92596312	55845455	23832416	27522249	14254199
F3	3.35E+09	2.85E+09	3.77E+08	6.32E+08	2.17E+12	1.03E+13	2.16E+10	1.24E+10	5.82E+09	4.29E+09
F4	17826.57	5311.842	36419.21	7625.011	19830.61	16822.87	73589.2	22333.16	37494.3	7975.682
F5	3.207309	1.160075	5.524236	30.25749	4269.918	6561.401	132.8654	28.99177	1147.789	1160.091
F6	94.74387	33.161	36.1114	15.79434	407.7684	380.0673	146.4584	48.07945	161.5487	46.02031
F7	122.5543	31.21892	63.83607	15.13432	381.5227	501.0868	365.0493	407.8801	54.81723	16.71206
F8	21.01456	0.062922	20.97727	0.047266	20.97577	0.061351	20.963	0.053337	20.98347	0.045214
F9	31.57857	3.567546	32.39764	1.9602	32.53643	3.68242	37.54293	3.444963	20.5147	3.593446
F10	37.06136	12.80079	3.372953	9.320364	826.4471	660.6651	119.5265	49.89614	337.9304	176.2492
F11	12.46478	3.227798	12.25511	7.787907	253.2155	118.103	494.1376	98.91452	117.2497	38.36261
F12	166.7471	46.86798	179.2385	15.38812	342.4555	109.8611	517.1603	104.6119	160.1306	64.49017
F13	220.5353	44.3755	185.12	15.17705	363.5705	111.7234	494.7345	100.9042	196.1197	47.04376
F14	1095.452	277.4216	710.2058	250.2053	4276.665	1008.22	5324.428	922.3581	3061.444	655.2746
F15	5532.33	716.6023	7074.786	452.8437	4894.003	766.8807	5868.402	971.1377	4039.845	1672.409
F16	2.523974	0.388452	2.640978	0.302431	1.89517	0.531663	1.900875	0.503098	2.71892	0.312373
F17	52.22566	4.501254	44.65577	8.797319	321.8197	153.3608	622.049	82.69208	199.9262	44.10574
F18	269.0597	25.17444	222.8703	11.97936	366.4005	155.2992	588.3506	109.9185	287.1607	44.12155
F19	7.035856	2.162483	88.2282	448.8411	53496.45	206947.8	63.22957	22.45551	411.4745	887.3135
F20	13.79839	0.874531	12.8085	0.25925	13.22101	0.991915	14.78657	0.248251	13.15225	1.343945
F21	364.9776	67.52968	271.4515	57.58184	680.7613	501.7437	382.1256	67.99994	1309.703	383.6555
F22	1100.562	300.5451	511.3767	223.7623	4466.315	886.1114	6540.228	945.0371	3300.589	713.2825
F23	6290.525	943.0522	7149.653	442.7992	5719.571	999.8704	6598.121	1065.787	4404.052	1454.952
F24	279.6475	15.72454	279.5597	6.123687	307.4106	9.454427	313.5599	12.77925	257.2353	8.36071
F25	303.8829	11.45014	285.1288	5.489875	318.0505	14.38233	324.0143	11.60171	275.4095	8.101226
F26	312.7265	86.62918	257.6766	83.55917	352.6611	74.5788	367.3623	75.79114	325.7925	57.28825
F27	1085.431	103.4058	1096.417	56.83421	1227.7	99.40051	1323.541	69.48118	825.9263	65.72772
F28	824.0976	844.5101	534.305	410.7301	2836.366	975.1396	4516.61	569.0731	1398.433	443.5709
K=3, win	21	19	20	10	27	24	26	22	22	21
K=6, win	19	20	19	12	24	26	26	22	24	20

Table 6. The experimental results of the JS, BOA, SSA, MRFO and SOA algorithms on 30D

Function	JS		BOA		SSA		MRFO		SOA	
	mean	std	mean	std	mean	std	mean	std	mean	std
F1	6.73E-10	1.59E-09	57531.15	3448.22	1.08E-12	4.5E-13	7.27E-12	1.51E-11	5767.182	2127.976
F2	2466906	973506.9	1.54E+09	5.3E+08	3156405	1554435	719649.4	272431.7	36279968	14602409
F3	4.53E+08	6.2E+08	1.39E+21	4.29E+21	1.16E+09	8.89E+08	2.96E+08	3.17E+08	1.37E+10	6.32E+09
F4	16523.06	3746.482	66766.61	2228.394	39295.36	5656.328	6479.481	3620.848	36246.24	6953.384
F5	2.28E-08	9.89E-08	37819.9	10599.97	7.08E-12	1.01E-11	1.93E-12	2.03E-12	2005.619	1052.616
F6	71.28246	18.66047	13938.69	3490.159	36.61546	26.76641	37.94645	28.04896	264.9357	83.91498
F7	98.68171	23.97042	8692121	8761836	461.3639	1383.957	125.8213	30.97999	99.17569	16.19126
F8	20.96523	0.067653	20.96654	0.048356	20.9753	0.048827	20.98071	0.063598	20.99725	0.045557
F9	30.37451	3.164516	37.76753	1.760165	35.1354	3.072498	31.64337	3.7894	29.77655	3.601051
F10	1.136888	0.699928	9962.625	1393.658	0.266622	0.147702	0.193446	0.116717	702.2485	280.9877
F11	267.1178	58.13557	848.4123	60.1566	366.8806	103.0253	284.1898	99.63411	260.2921	37.08796
F12	233.731	57.97031	809.3627	94.53336	567.4477	186.9021	323.9546	89.55887	245.3688	53.77497
F13	290.1981	70.06344	802.9679	66.84484	561.2594	170.3178	355.9022	68.87073	310.6366	49.94417
F14	2127.966	588.8412	7784.21	457.9327	3512.519	572.1674	3279.588	720.5992	4778.072	774.2109
F15	6168.199	1395.788	7762.64	379.7931	5037.948	789.4932	4075.993	510.3017	4995.815	849.2191
F16	2.529235	0.305255	2.801721	0.248536	1.746564	0.479397	2.475954	0.433586	2.082306	0.400395
F17	290.8306	83.25273	855.8822	39.21048	705.2523	126.0239	402.8115	133.211	458.5184	57.04758
F18	309.083	37.05812	838.7355	56.06826	735.2734	125.5794	279.5979	131.9974	449.499	69.13246
F19	88.57673	75.12815	710644.3	190711.2	25.86628	10.34279	20.44952	8.441692	9102.513	41849.82
F20	12.15662	0.75601	14.99993	0.000359	14.80375	0.567777	13.15939	1.171283	12.80903	0.68778
F21	381.3417	72.34723	2572.389	60.86354	350.751	85.0941	368.0083	96.76306	1688.898	308.2996
F22	2932.12	662.0221	8385.241	315.4982	4426.713	938.3524	3722.385	970.6226	5239.346	751.8852
F23	5681.708	1431.926	8463.848	375.1362	5973.113	1063.452	4916.12	768.7731	5729.366	1079.06
F24	286.7253	11.52224	374.3448	39.78452	326.126	86.62957	295.158	11.95336	287.2667	10.86883
F25	311.015	9.306193	347.2868	23.11617	320.5632	11.00025	320.4388	13.85018	294.8671	10.68163
F26	291.6394	81.60908	281.1141	60.74828	362.1012	74.09312	200.0431	0.016256	224.0683	58.928
F27	1002.553	106.5557	1914.951	188.7125	1227.955	109.3907	1151.464	98.16136	1079.788	100.9288
F28	2757.278	713.7003	5493.255	499.2617	4268.823	1303.935	3218.544	718.0367	1878.418	198.2351
K=3, win	21	17	26	18	23	20	19	20	26	20
K=6, win	18	20	27	18	20	21	18	20	23	21

Table 7. The experimental results of the GA, DE, PSO, WOA and GWO algorithms on 50D

Function	GA		DE		PSO		WOA		GWO	
	mean	std	mean	std	mean	std	mean	std	mean	std
F1	138.4604	45.32537	31.26765	171.26	6951.315	9296.255	117.1347	61.26803	4769.89	2612.421
F2	77677480	19904935	2.01E+08	63004332	1.68E+08	3.13E+08	90660402	21109402	59601675	24624726
F3	1.94E+10	1.31E+10	1.66E+10	7.77E+09	3.01E+11	8.04E+11	4.55E+10	1.51E+10	2.41E+10	7.31E+09
F4	33811.94	7898.252	91997.23	11110.11	37800.16	23107.45	68923.24	10924.72	49957.55	7337.647
F5	20.26826	5.086442	51.14632	119.9528	6275.362	7099.478	261.9072	90.31555	1168.734	981.6189
F6	170.0405	80.44957	46.25593	11.56224	414.1749	329.0915	265.861	67.97517	397.0171	108.2144
F7	155.2019	24.83774	129.7206	18.67058	300.1518	199.4576	662.8723	734.4361	73.33709	11.04796
F8	21.19226	0.043877	21.16519	0.028805	21.16782	0.038569	21.15428	0.035305	21.16348	0.04461
F9	61.49513	6.103541	67.62077	2.127592	61.85371	4.550176	69.75953	4.125024	40.2868	3.520289
F10	297.0652	61.04447	5.058406	14.15881	1268.147	964.7758	379.9499	98.38282	960.0045	249.1535
F11	84.35533	12.17755	30.82997	9.290483	543.6186	149.3913	810.7925	90.05439	302.3784	55.29677
F12	465.5647	72.00788	418.1629	18.32047	651.5593	142.6071	955.4687	143.831	306.8703	86.97624
F13	515.802	79.05696	410.7213	18.32807	738.548	130.7393	978.7048	134.4973	419.2359	84.75965
F14	2996.24	483.7888	1073.449	387.5915	7582.233	1287.488	9860.203	1388.217	6258.712	1598.336
F15	12183.31	1078.137	14354.47	370.5647	9730.845	1031.935	11154.85	1429.48	8256.744	3160.583
F16	3.869842	0.616202	3.690207	0.322226	2.630718	0.639014	2.715584	0.638379	3.679927	0.274768
F17	168.1921	16.84471	188.2981	9.365485	842.6115	200.1628	1152.252	115.8864	387.0248	64.34234
F18	557.81	56.35016	448.2396	18.54875	863.6423	251.1883	1153.813	143.4793	586.1874	78.13226
F19	25.07318	6.437512	39.41424	112.357	476067.6	959660	180.917	44.9029	3282.201	6036.917
F20	23.63635	0.900632	22.76847	0.265106	23.61885	0.995594	24.66064	0.357453	21.41506	0.793517
F21	795.7041	382.7631	849.8604	381.379	1125.131	532.8774	1469.072	540.8025	2631.727	666.1197
F22	3552.827	555.9556	1669.622	1494.143	9526.982	1238.024	11777.07	1462.901	7714.162	2053.726
F23	13836.83	982.38	14399.89	449.4398	11264.49	2055.838	13417.28	1298.598	8811.322	2119.322
F24	350.3481	18.47074	361.1505	8.356525	402.6694	19.66707	412.3706	14.54932	313.9142	11.15468
F25	398.8186	16.42209	372.1048	7.249977	419.7917	14.5425	436.2235	17.69039	354.5284	11.92513
F26	409.9057	94.60607	338.0832	121.1317	433.2152	85.7065	466.1766	71.57856	381.7858	61.04963
F27	1842.515	187.5039	1932.262	91.28491	2069.93	96.18577	2272.513	109.1509	1346.654	105.9469
F28	1281.121	1552.038	1258.751	1435.903	4230.783	2023.156	8172.634	1686.339	2437.073	1148.367
K=3, win	19	18	20	11	25	24	25	23	17	19
K=6, win	20	19	19	9	23	24	24	23	17	19

Table 8. The experimental results of the JS, BOA, SSA, MRFO and SOA algorithms on 50D

Function	JS		BOA		SSA		MRFO		SOA	
	mean	std	mean	std	mean	std	mean	std	mean	std
F1	0.658438	3.597024	79614.83	4387.491	2.98E-12	1.71E-12	5.67E-07	3.1E-06	19483.91	4600.396
F2	7098967	1326959	3.14E+09	9.46E+08	6358213	2060433	2590323	1067854	97517434	43820073
F3	2.01E+09	9.71E+08	1.93E+20	5.53E+20	3.36E+09	2.88E+09	1.12E+09	1.32E+09	4.15E+10	1.27E+10
F4	31362.65	4952.535	94486.37	7257.858	102740.3	17571.6	28978.81	8723.476	59496.57	7151.036
F5	0.074729	0.28068	25243.55	4383.78	1.68E-05	1.04E-05	4.06E-10	1.51E-09	3372.928	1397.865
F6	98.62407	35.81018	11263.49	1678.882	75.40745	31.61675	73.9747	28.88176	1191.523	353.4238
F7	105.0368	20.89968	83789.81	79290.35	271.7981	164.4141	124.242	30.56187	112.6048	14.59396
F8	21.17242	0.054077	21.16285	0.038189	21.16796	0.031733	21.15879	0.057575	21.16598	0.037629
F9	55.35956	3.68367	71.31799	2.001563	66.2454	4.979508	61.9705	4.929116	57.03193	4.016654
F10	16.58311	9.961833	14576.12	1380.691	0.394054	0.327302	0.16967	0.074998	2175.525	460.8635
F11	557.1529	71.42898	1110.704	52.62632	638.4877	104.191	575.1105	136.3661	584.936	53.93134
F12	457.2435	88.29039	1198.373	58.34024	1061.026	57.29857	672.8795	136.0627	563.4211	66.22554
F13	588.6189	67.72029	1199.316	69.28912	1124.448	183.2224	733.0092	117.3841	634.396	70.49256
F14	4653.865	778.2087	14451.84	469.0507	5672.818	873.2752	5818.202	1210.473	9188.708	964.6076
F15	13894.76	830.3297	14882.97	520.3422	9070.244	975.6274	7864.908	1014.387	10970.33	1548.845
F16	3.711419	0.239728	3.849144	0.273853	2.545799	0.653556	3.316913	0.606217	3.286199	0.571076
F17	663.8371	125.4801	1376.463	42.25339	1017.454	129.7143	826.4854	141.3099	889.4302	114.8224
F18	651.5616	83.43523	1365.507	57.56635	1148.35	52.50265	688.9859	218.3507	913.9238	81.60569
F19	390.6677	399.8741	1358382	233141.1	76.84017	22.2787	55.68012	21.07047	30680.87	51459.57
F20	22.32491	0.877741	24.92593	0.110678	24.74687	0.244052	23.30974	1.082247	22.80691	1.046813
F21	939.2173	199.2073	4548.103	53.91577	950.7678	142.413	950.7622	142.406	3769.749	340.8457
F22	6420.441	1171.462	15813.86	398.0635	8510.626	1503.407	7426.211	1504.901	10532.24	1058.846
F23	13241.58	1798.668	15956.53	417.7359	11314.99	1564.085	9883.46	1175.026	11030.93	1515.45
F24	368.0008	13.03199	646.8198	130.3581	398.8679	14.40966	386.6209	14.04718	365.6246	13.35288
F25	413.2591	17.15056	491.1032	27.06354	425.4208	14.35237	439.4362	25.45498	386.9537	15.517
F26	410.5566	84.08666	443.3786	79.16116	445.6014	83.83203	361.7771	125.1741	450.7944	13.3342
F27	1864.147	117.5594	3582.503	491.7392	2159.57	167.6418	1975.933	139.1017	1845.072	144.1397
F28	4827.721	1490.715	9770.834	715.6657	5995.975	2994.601	4692.022	2298.082	4278.481	931.3917
K=3, win	23	15	28	17	22	18	17	18	25	20
K=6, win	22	18	27	18	20	17	17	18	23	19

Table 9. The experimental results of the GA, DE, PSO, WOA and GWO algorithms on 100D

Function	GA		DE		PSO		WOA		GWO	
	mean	std	mean	std	mean	std	mean	std	mean	std
F1	1909.327	381.4485	117.5413	443.9738	20322.81	15070.91	2507.652	785.8997	29357.94	7602.8
F2	3.16E+08	52161982	9.72E+08	1.46E+08	8.14E+08	9.8E+08	3.13E+08	66176140	2.04E+08	74334905
F3	1.4E+11	3.49E+10	8.98E+10	3.17E+10	8.28E+13	4.23E+14	1.18E+14	3.81E+14	2.64E+12	4.66E+12
F4	98041.91	16741.1	257622.1	17968.63	123599.7	52803.15	480937.5	70094.94	126503	10649.23
F5	167.9382	33.42148	520.5181	1222.608	24479.2	22137.97	1369.543	206.7681	6735.436	2140.254
F6	1061.354	129.5758	242.8282	26.94896	2080.662	1335.255	1089.383	220.1019	3107.467	1134.132
F7	1020.958	1389.564	189.3974	13.6291	84606.22	183236.6	2193102	4361836	485.8619	264.6581
F8	21.34051	0.044407	21.33393	0.019357	21.31489	0.044329	21.3331	0.03411	21.3478	0.024128
F9	137.3006	10.67964	159.9213	3.051253	148.6034	5.343238	157.4193	5.711231	99.97048	5.801225
F10	1642.724	235.7736	49.0652	72.8573	4197.962	1947.641	1650.312	317.8302	3446.464	567.3849
F11	484.6748	52.88436	463.8083	38.75041	1903.606	335.2896	2340.769	159.0941	933.6864	112.9413
F12	1296.838	179.1738	1034.374	34.24023	2103.463	315.8261	2433.679	253.7415	928.0646	154.2877
F13	1384.332	103.7523	1036.462	33.58637	2134.813	352.1454	2616.658	219.6085	1201.362	166.4994
F14	12195.48	835.9804	15141.6	1152.272	20214.09	2219.777	23410.22	2811.295	16204.39	1182.912
F15	29452.4	1257.904	31666.37	545.1553	21855.38	1392.4	25767.42	1945.68	15713.5	4226.84
F16	4.604012	0.258509	4.389798	0.2488	3.645832	0.588476	3.625004	0.472722	4.359996	0.230861
F17	661.632	54.17324	656.1054	24.07945	2651.336	594.7682	2999.877	143.1233	1311.181	181.053
F18	1514.006	117.9767	1099.863	34.44211	2811.369	417.5795	3016.177	131.0211	1678.297	168.047
F19	128.343	44.37648	8399.177	45181.77	971991.3	1315773	1102.3	424.5641	86405.53	120404.8
F20	49.99995	0.000283	50	0	50	0	50	0	49.99427	0.022289
F21	1113.785	544.9645	424.5252	305.8433	2151.405	1855.473	1114.871	584.2577	5691.105	589.8719
F22	14314.03	2148.82	13338.46	2477.874	24022.5	2632.255	26883.29	2028.015	19537.7	2878.451
F23	31410.77	1688.875	32245.57	610.56	26692.57	1914.301	30181.42	1972.731	21671.18	5047.083
F24	569.6793	31.88158	586.2984	10.29793	745.9387	83.87324	685.5783	28.34847	488.2359	23.1066
F25	650.6229	32.4003	612.6162	15.10679	710.4656	40.87077	736.4244	41.36531	597.8205	24.44788
F26	638.8822	31.74385	695.5814	7.761697	652.9042	114.3959	710.1191	11.76008	542.4258	62.22963
F27	3791.994	255.5844	4257.914	90.28709	4523.522	204.9405	4737.575	223.7674	3061.916	209.9649
F28	9044.458	2156.747	4533.795	1122.073	19101.41	3435.119	23260.2	2115.844	9321.781	837.7965
K=3, win	23	19	18	9	24	22	26	20	19	18
K=6, win	22	21	20	12	23	23	26	21	18	20

Table 10. The experimental results of the JS, BOA, SSA, MRFO and SOA algorithms on 100D

Function	JS		BOA		SSA		MRFO		SOA	
	mean	std	mean	std	mean	std	mean	std	mean	std
F1	0.658438	3.597024	79614.83	4387.491	2.98E-12	1.71E-12	5.67E-07	3.1E-06	19483.91	4600.396
F2	7098967	1326959	3.14E+09	9.46E+08	6358213	2060433	2590323	1067854	97517434	43820073
F3	2.01E+09	9.71E+08	1.93E+20	5.53E+20	3.36E+09	2.88E+09	1.12E+09	1.32E+09	4.15E+10	1.27E+10
F4	31362.65	4952.535	94486.37	7257.858	102740.3	17571.6	28978.81	8723.476	59496.57	7151.036
F5	0.074729	0.28068	25243.55	4383.78	1.68E-05	1.04E-05	4.06E-10	1.51E-09	3372.928	1397.865
F6	98.62407	35.81018	11263.49	1678.882	75.40745	31.61675	73.9747	28.88176	1191.523	353.4238
F7	105.0368	20.89968	83789.81	79290.35	271.7981	164.4141	124.242	30.56187	112.6048	14.59396
F8	21.17242	0.054077	21.16285	0.038189	21.16796	0.031733	21.15879	0.057575	21.16598	0.037629
F9	55.35956	3.68367	71.31799	2.001563	66.2454	4.979508	61.9705	4.929116	57.03193	4.016654
F10	16.58311	9.961833	14576.12	1380.691	0.394054	0.327302	0.16967	0.074998	2175.525	460.8635
F11	557.1529	71.42898	1110.704	52.62632	638.4877	104.191	575.1105	136.3661	584.936	53.93134
F12	457.2435	88.29039	1198.373	58.34024	1061.026	57.29857	672.8795	136.0627	563.4211	66.22554
F13	588.6189	67.72029	1199.316	69.28912	1124.448	183.2224	733.0092	117.3841	634.396	70.49256
F14	4653.865	778.2087	14451.84	469.0507	6572.818	873.2752	5818.202	1210.473	9188.708	964.6076
F15	13894.76	830.3297	14882.97	520.3422	9070.244	975.6274	7864.908	1014.387	10970.33	1548.845
F16	3.711419	0.239728	3.849144	0.273853	2.545799	0.653556	3.316913	0.606217	3.286199	0.571076
F17	663.8371	125.4801	1376.463	42.25339	1017.454	129.7143	826.4854	141.3099	889.4302	114.8224
F18	651.5616	83.43523	1365.507	57.56635	1148.35	52.50265	688.9859	218.3507	913.9238	81.60569
F19	390.6677	399.8741	1358382	233141.1	76.84017	22.2787	55.68012	21.07047	30680.87	51459.57
F20	22.32491	0.877741	24.92593	0.110678	24.74687	0.244052	23.30974	1.082247	22.80691	1.046813
F21	939.2173	199.2073	4548.103	53.91577	950.7678	142.413	950.7622	142.406	3769.749	340.8457
F22	6420.441	1171.462	15813.86	398.0635	8510.626	1503.407	7426.211	1504.901	10532.24	1058.846
F23	13241.58	1798.668	15956.53	417.7359	11314.99	1564.085	9883.46	1175.026	11030.93	1515.45
F24	368.0008	13.03199	646.8198	130.3581	398.8679	14.40966	386.6209	14.04718	365.6246	13.35288
F25	413.2591	17.15056	491.1032	27.06354	425.4208	14.35237	439.4362	25.45498	386.9537	15.517
F26	410.5566	84.08666	443.3786	79.16116	445.6014	83.83203	361.7771	125.1741	450.7944	13.3342
F27	1864.147	117.5594	3582.503	491.7392	2159.57	167.6418	1975.933	139.1017	1845.072	144.1397
F28	4827.721	1490.715	9770.834	715.6657	5995.975	2994.601	4692.022	2298.082	4278.481	931.3917
K=3, win	23	15	28	17	22	18	17	18	25	20
K=6, win	22	18	27	18	20	17	17	18	23	19

Table 11. The mean of the TOA algorithm with K = 3

Dim	Type	GA	DE	PSO	WOA	GWO	JS	BOA	SSA	MRFO	SOA
D=30	Unimodal	5	5	5	5	5	**2**	5	3	**1**	5
	Multimodal	10	10	14	13	13	12	14	12	11	14
	Composition	6	5	8	8	4	7	7	8	7	7
	win	21	20	27	26	22	21	26	23	19	26
D=50	Unimodal	4	5	4	5	5	2	5	2	0	5
	Multimodal	11	11	13	12	9	14	15	12	11	12
	Composition	4	4	8	8	**3**	7	8	8	6	8
	win	19	20	25	25	17	23	28	22	17	25
D=100	Unimodal	4	5	4	5	4	4	5	**2**	**0**	5
	Multimodal	11	8	13	13	11	13	15	11	11	13
	Composition	8	5	7	8	4	7	8	6	6	7
	win	23	18	24	26	19	24	28	19	17	25

Table 12. The std of the TOA algorithm with K = 3

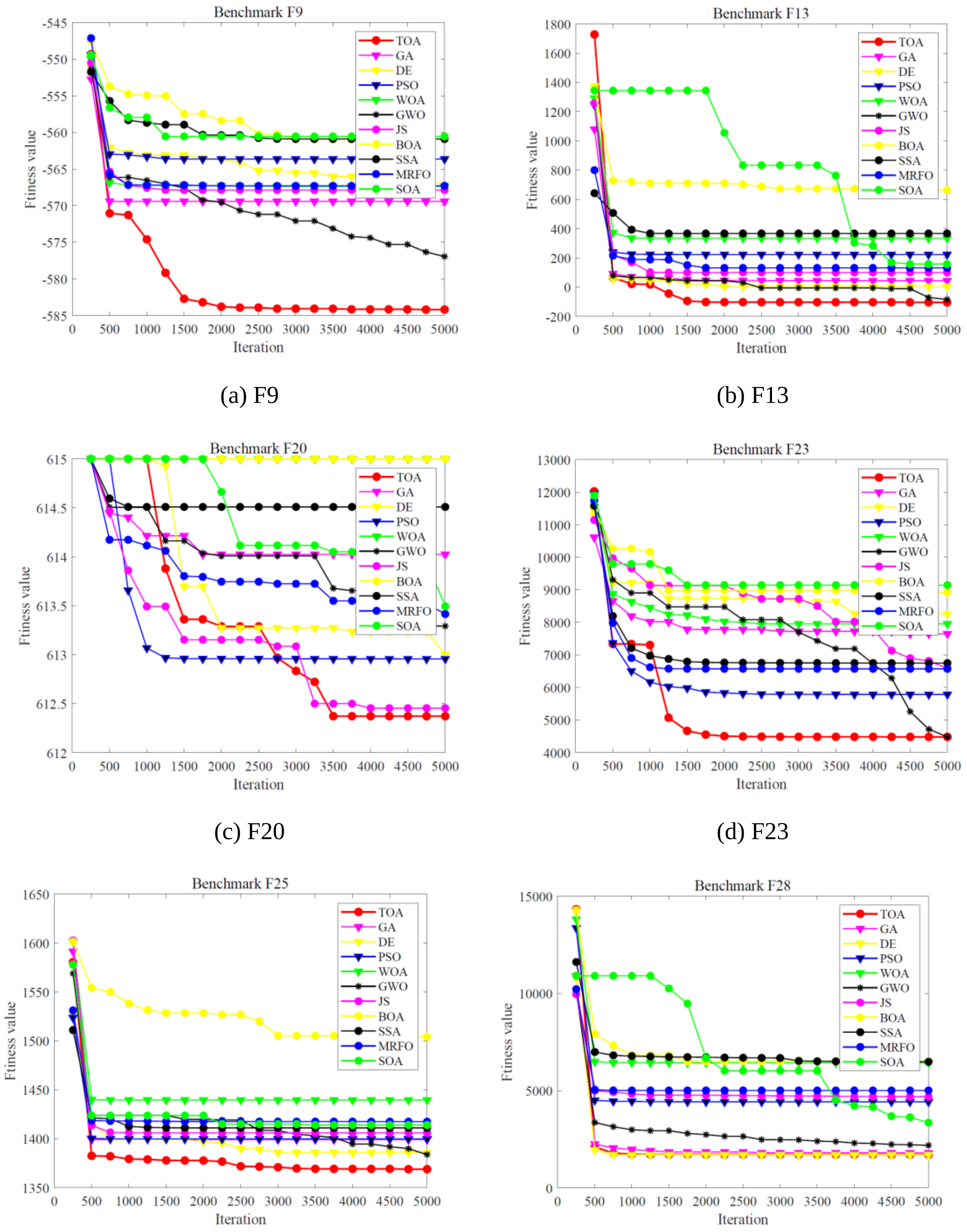
Dim	Type	GA	DE	PSO	WOA	GWO	JS	BOA	SSA	MRFO	SOA
D=30	Unimodal	5	5	5	5	5	**1**	4	3	**1**	5
	Multimodal	9	**3**	12	12	11	10	9	10	14	9
	Composition	5	**2**	7	5	5	6	5	7	5	6
	win	19	**10**	24	22	21	17	18	20	20	20
D=50	Unimodal	4	5	5	5	4	**2**	4	**2**	**1**	4
	Multimodal	9	**3**	13	12	11	9	9	10	12	11
	Composition	5	**3**	6	6	4	4	4	6	5	5
	win	18	**11**	24	23	19	15	17	18	18	20
D=100	Unimodal	4	4	5	5	4	4	5	**2**	**1**	4
	Multimodal	9	**3**	11	10	9	9	**7**	8	8	10
	Composition	6	**2**	6	5	5	4	5	5	4	3
	win	19	**9**	22	20	18	17	17	15	**13**	17

Table 13. The mean of the TOA algorithm with K = 6

Dim	Type	GA	DE	PSO	WOA	GWO	JS	BOA	SSA	MRFO	SOA
D=30	Unimodal	4	4	4	5	4	**1**	5	**2**	**1**	4
	Multimodal	9	10	13	13	14	11	15	11	11	13
	Composition	6	5	7	8	6	6	7	7	6	6
	win	19	19	24	26	24	18	27	20	18	23
D=50	Unimodal	4	5	4	4	4	**1**	5	**2**	**0**	4
	Multimodal	10	9	12	12	9	13	14	11	11	12
	Composition	6	5	7	8	4	8	8	7	6	7
	win	20	19	23	24	17	22	27	20	17	23
D=100	Unimodal	4	5	4	5	4	4	5	**2**	**1**	4
	Multimodal	11	10	12	13	10	11	15	10	10	13
	Composition	7	5	7	8	4	6	8	5	5	7
	win	22	20	23	26	18	21	28	17	16	24

Table 14. The std of the TOA algorithm with K = 6

Dim	Type	GA	DE	PSO	WOA	GWO	JS	BOA	SSA	MRFO	SOA
D=30	Unimodal	4	4	5	5	4	**1**	4	**2**	**1**	4
	Multimodal	10	**5**	13	11	11	12	9	11	12	11
	Composition	6	**3**	8	6	5	7	5	8	7	6
	win	20	**12**	26	22	20	20	18	21	20	21
D=50	Unimodal	4	4	5	4	4	2	4	**2**	**1**	4
	Multimodal	10	**4**	13	13	10	11	10	9	11	11
	Composition	5	**1**	6	6	5	5	4	6	6	4
	win	19	**9**	24	23	19	18	18	17	18	19
D=100	Unimodal	4	4	5	5	4	4	5	**2**	**0**	4
	Multimodal	11	**6**	12	12	11	12	10	8	9	11
	Composition	6	**2**	6	4	5	5	5	4	5	5
	win	21	**12**	23	21	20	21	20	14	14	20



(a) F9 (b) F13

(c) F20 (d) F23

(e) F25 (f) F28

Figure 6. Convergence curve of functions F9, F13, F20, F23, F25, F28

5 The Application of Path Planning Problem in Smart City

For numerical optimization problems, the TOA algorithm has better convergence performance and optimization effect. In order to further verify the effectiveness of the proposed algorithm in solving practical problems, the TOA algorithm is further tested in this paper. In this section, the application of TOA algorithm in path planning problem in smart city will be introduced.

5.1 System Model

Suppose there is a city with an area of $M \times M$. The city is divided into multiple areas, and each area contains a different number of sensor nodes and a base station. The sensor node is responsible for monitoring the current area and collecting data in the monitored area (such as air temperature and humidity, smoke, traffic flow, etc.). The sensor node sends the collected data to the base station in the area. Each base station has a fixed storage capacity, so the collected data will first be stored in the storage area of the base station. When the data in the storage area reaches the upper limit of the storage capacity, the base station organizes and packs the collected data. It is transmitted to the data center through wireless transmission or optical fiber transmission. The data center processes the data from the base station to form a visual report, thus realizing the monitoring of the entire city.

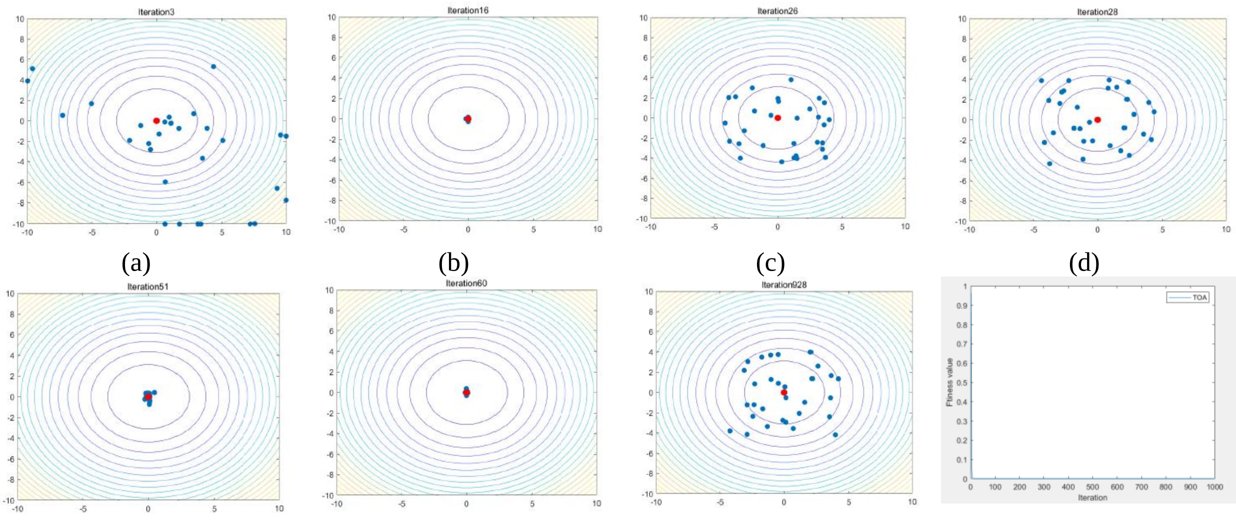


Figure 7. The distribution of particles in the solving process of the TOA algorithm

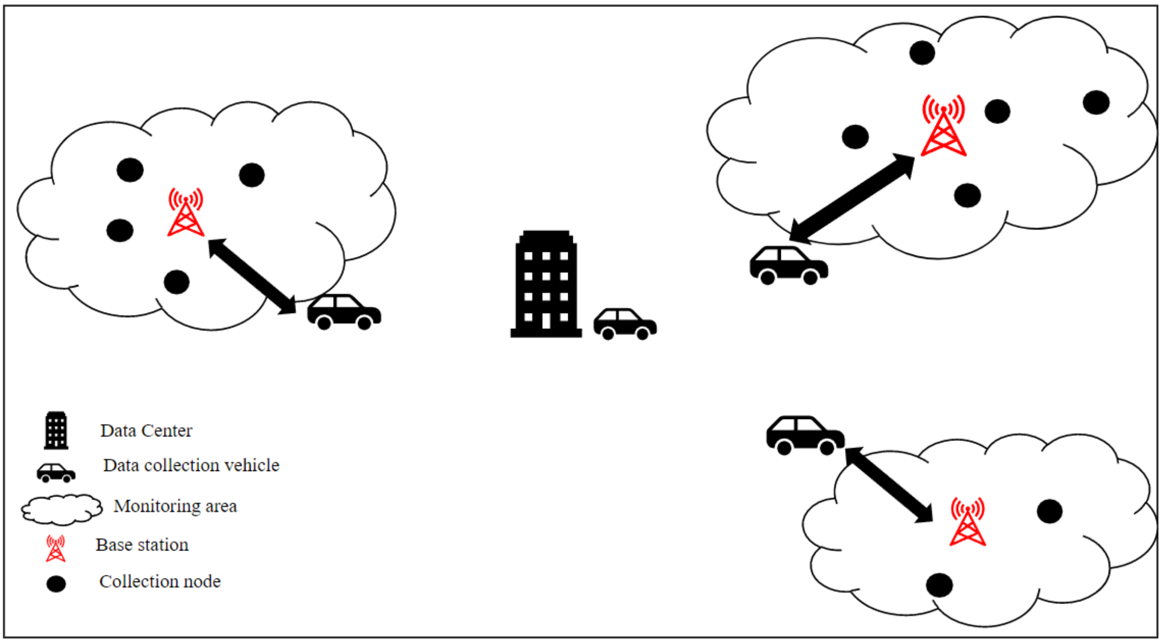


Figure 8. The system model

In order to relieve the pressure of base stations to transmit data, the data collection vehicles are introduced in the city. The collection vehicle starts from the data center, traverses each monitoring area at a certain speed, and collects the data stored in the storage area of the base station. The data collection vehicle finally returns to the data center and delivers the collected data to the processing program of the data center. Figure 8 shows the system model.

5.2 Objective Function Establishment

In this system, the main goal is to optimize the path length of the data collection vehicle, thereby reducing the fuel consumption of the collection vehicle. At the same time, it also needs to maximize the amount of path data collection to reduce the forwarding pressure of the base station. Equations (10) and (11) give mathematical models for these two goals. Table 15 gives the definition and meaning of related symbols in this section. The above two goals are based on the following assumptions:

(1) The data receiving rate and moving speed of the data acquisition vehicle are constant.

(2) Do not consider the failure of the data collection vehicle and the base station.

(3) The data storage capacity of the data collection vehicle is not considered.

$$f(1) = \min\left(\sum_{i=1}^{N+1} d_{s(i),s(i+1)}\right) \quad \textbf{(10)}$$

$$f(2) = \max\left(\sum_{i=1}^{N} c_{s(i)}\right) \quad \textbf{(11)}$$

Because the data collection vehicle needs to start from the data center and eventually return to the data center. Therefore, when encoding, the first element and the last element of the sequence s are the numbers corresponding to the data center. In this paper, the data center node is set to node 1 by default. Therefore, the sequence s actually contains N+2 elements. Equation (12) gives the calculation formula of the objective function $f(2)$. As the process of data collection by sensor nodes is continuous, data is added in the storage area of base stations all the time. When the data collection vehicle does not reach the base station and the data storage capacity has reached the maximum, the base station will directly forward the data to the data center. So we carry on the operation of mod to this process. The calculation of the time for the data collection vehicle to arrive at base station i includes two parts, one part is the total movement time of the data collection vehicle, and the other part is the sum of the data collection time of the previous i-1 nodes. Equation (13) is the calculation formula for this process.

$$c_{s(i)} = mod(R_{s(i)} * t_{s(i)} + C_{s(i)}, Max\_C) \quad \textbf{(12)}$$

$$t_{s(i)} = \frac{\sum_{k=1}^{i-1} d_{s(k),s(k+1)}}{v_m} + \frac{\sum_{k=1}^{i-1} c_{s(k)}}{Send} \quad \textbf{(13)}$$

Since the objective function $f(1)$ is a minimization problem, the objective function $f(2)$ is a maximization problem. Therefore, in order to simplify the solution, this paper takes the reciprocal of objective function $f(2)$ and unifies the two objectives into a minimization problem. By setting the priority of each objective, the optimization problem of the above two objectives is transformed into a single objective problem. Finally, the objective function and the fitness function in the algorithms are shown in Equation (14). α and β represent the priority of the two goals. $\alpha > \beta$, the priority is to minimize the path length; $\alpha < \beta$, the priority is to maximize the amount of path data collection; $\alpha = \beta$, the priority is the same, and the result is a compromise between the two goals. In this paper, both α and β are set to 0.5. The path planning problem proposed in this paper can be regarded as a TSP problem considering the time factor. The TSP problem itself is an NP-Hard problem, so the problem proposed in this paper is also an NP-Hard problem.

$$\min f = \alpha f(1) + \beta \frac{1}{f(2)} \quad \textbf{(14)}$$

Table 15. Description of relevant symbols in smart city path planning

Symbol	Meaning
N	Number of base stations
s	Sequential sequence of the data collection vehicle visiting the base stations
d	Distance matrix
$c_{s(i)}$	The amount of data stored in the i-th base station storage area
$R_{s(i)}$	The rate at which the i-th base station collects node data
$C_{s(i)}$	The initial data of the i-th base station storage area
Max_C	Maximum storage capacity of base station
v_m	The moving speed of the data collection vehicle
Send	The rate at which the base station transmits data to the data collection vehicle

5.3 Experiment Analysis

In order to verify the effectiveness of the proposed algorithm in solving this problem, simulation experiments are carried out in this subsection. The selected urban area is 1000km×1000km, and the data center is located in the center of the area, and the coordinate is (500, 500). Due to the different number of nodes connected to each base station, the amount of data collected per unit time is different. Initially, the amount of data stored in the storage area of each base station is also different. The maximum capacity Max_C of each base station is set to 2048MB. The data transmission rate *Send* to the collection vehicle is set to 20MB/s. The moving speed of the data collection vehicle is set to 36km/h (10m/s). During the experiment, the maximum number of iterations for each algorithm is 2000, and the number of consecutive tests is 20. The other parameter settings of the experiment are the same as those in the Section 4. Table 16 records the mean, standard deviation, and algorithm ranking when the city has different numbers of base stations.

It can be seen from the table that the TOA algorithm also has certain competitiveness when solving this problem. By calculating the average ranking, the following ranking results are obtained:

TOA with “K = 6” (2.625) < DE (3.625) < JS (3.875) < TOA with “K = 3” (4.875) < GA (5.625) = GWO (5.625) < WOA (6.875) < MRFO (8) < BOA (8.5) < SSA (9.125) < SOA (9.5) < PSO (9.75)

The TOA algorithm with “K = 6” has the best overall performance, followed by the DE algorithm. It can also be seen that the performance of the TOA algorithm with “K = 3” is weaker than that of the TOA algorithm with “K = 3”. This is because in this problem, the optimization algorithm not only optimizes each dimension, but also considers the final generated path order. It means that each dimension needs to have strong mutation, that is, the algorithm needs to have strong global search ability. According to the analysis in Section 4.1, when K value is small, the algorithm has strong local search ability. For larger K value, the algorithm will have stronger global search ability. Therefore, for the solution of this problem, the TOA algorithm with “K = 3” is slightly weaker than the TOA algorithm with “K = 6”.

Figure 9 shows the convergence curves of all algorithms in solving the above examples. Due to the large number of groups, the TOA algorithm with “K = 6”, the convergence speed in the early stage will be weaker than that of “K = 3”. However, in the later stage, the TOA algorithm with “K = 3” relies on its strong global search ability, its final convergence accuracy will be better than “K = 3”. This is also consistent with the comparison of the experimental results of the two algorithms in Table 16. Figure 10 shows the path planning diagrams of the TOA algorithm with “K = 6”.

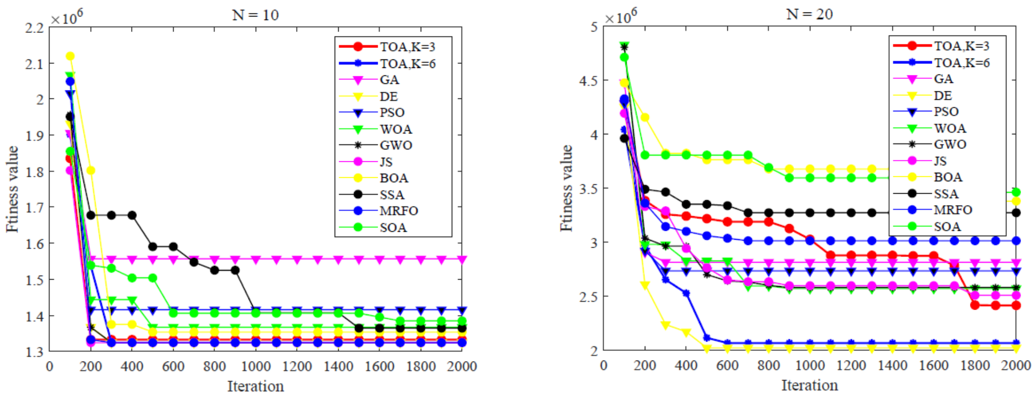
On the whole, the TOA algorithm can also achieve a better solution in the solution of the practical application problem. Compared with other algorithms, it also has strong competitiveness, especially when the number of base stations is large. This verifies the feasibility of the TOA algorithm in solving practical application problems.

6 Conclusion

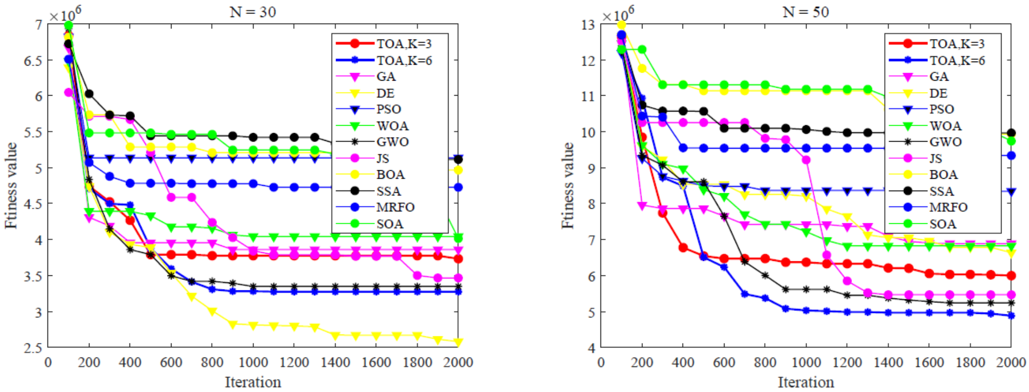
Inspired by the habits of tumbleweed plants in nature, this paper proposes a new swarm intelligence optimization algorithm called the tumbleweed optimization algorithm (TOA). The TOA algorithm simulates the two stages of tumbleweed seedling growth and seed propagation. The two stages correspond to the exploitation and exploration of algorithm respectively. And through the growth cycle, the two-stage switching is realized. The TOA algorithm is a multi-group structure algorithm. The introduction of multi-group structure can ensure that the algorithm can have a large search space, and then ensure the global search ability. In order to verify the optimization performance of the proposed algorithm, relevant experimental tests are carried out in this paper. On the test of CEC2013 benchmark functions, the TOA algorithm shows good optimization performance. Compared with the other ten intelligent optimization algorithms, the TOA algorithm also has strong competitiveness. At the same time, in order to verify the effectiveness and feasibility of the TOA algorithm in solving practical application problems, this paper establishes the data collection vehicle path planning model for testing. Through experimental comparison, the TOA algorithm can also better optimize this problem, which shows the feasibility in solving practical application problems. Like most algorithms, the TOA algorithm also has the problem of difficult parameter selection. Therefore, in the following work, we will focus on solving the problem of setting K value and further improve the optimization performance of TOA algorithm. In the future, we will also expand the application fields of the TOA algorithm to solve more practical application problems [52-54].

Table 16. The experimental results of the algorithms in solving cases with different *N*

Algorithm	N=10				N=20				N=30				N=50			
	mean	rank	std	rank	mean	rank	std	rank	mean	rank	std	rank	mean	rank	std	rank
TOA, K=3	1.37E+06	6	4.38E+04	4	2.43E+06	4	2.43E+05	7	3.40E+06	5	2.95E+05	5	5.96E+06	2	6.12E+05	6
TOA, K=6	1.34E+06	3	2.13E+04	3	2.25E+06	2	2.03E+05	5	3.01E+06	2	2.62E+05	3	5.17E+06	1	5.07E+05	2
GA	1.42E+06	10	1.11E+05	11	2.49E+06	6	2.01E+05	4	3.39E+06	4	2.86E+05	4	6.29E+06	5	4.08E+05	1
DE	1.33E+06	2	6.82E+03	2	2.10E+06	1	6.88E+04	1	2.73E+06	1	3.18E+05	7	6.25E+06	4	1.06E+06	11
PSO	1.45E+06	12	1.28E+05	12	2.87E+06	10	4.76E+05	12	4.20E+06	8	5.09E+05	11	8.15E+06	8	5.44E+05	5
WOA	1.36E+06	5	5.49E+04	5	2.72E+06	7	2.82E+05	8	4.00E+06	7	4.06E+05	9	8.00E+06	7	7.24E+05	7
GWO	1.39E+06	8	9.91E+04	9	2.48E+06	5	2.00E+05	3	3.47E+06	6	2.17E+05	1	6.05E+06	3	9.86E+05	10
JS	1.32E+06	1	2.12E-06	1	2.36E+06	3	1.92E+05	2	3.38E+06	3	3.06E+05	6	7.22E+06	6	9.65E+05	9
BOA	1.44E+06	11	9.91E+04	10	3.31E+06	12	2.26E+05	6	5.10E+06	12	2.39E+05	2	1.02E+07	12	5.18E+05	3
SSA	1.40E+06	9	7.27E+04	8	3.17E+06	11	3.31E+05	9	4.83E+06	11	4.22E+05	10	9.78E+06	11	5.26E+05	4
MRFO	1.36E+06	4	7.18E+04	7	2.75E+06	8	3.32E+05	10	4.33E+06	9	3.97E+05	8	8.84E+06	10	8.48E+05	8
SOA	1.39E+06	7	6.66E+04	6	2.80E+06	9	4.16E+05	11	4.40E+06	10	5.48E+05	12	8.49E+06	9	1.06E+06	12

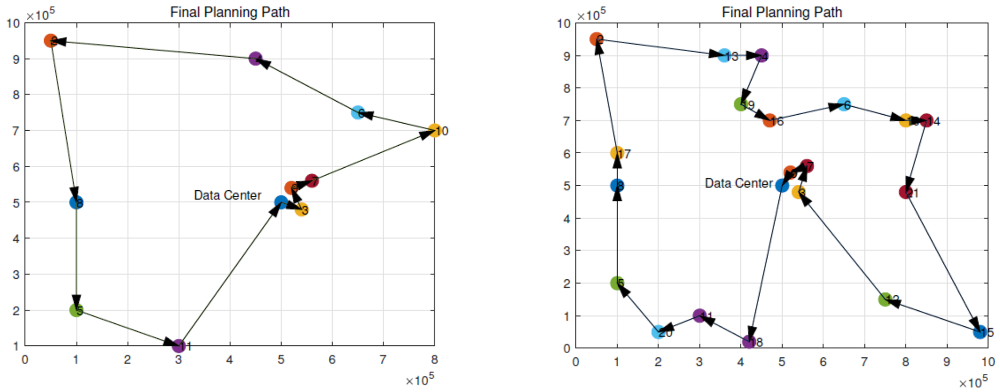


(a) N=10 (b) N=20

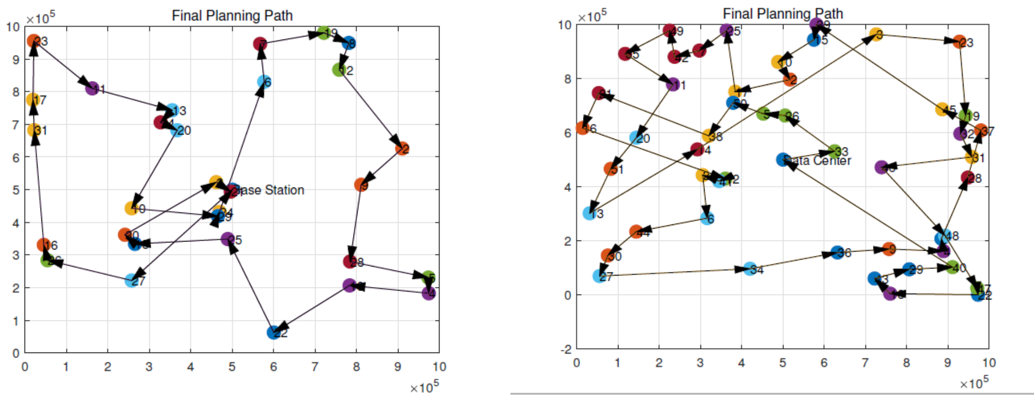


(c) N=30 (d) N=50

Figure 9. The convergence curves of the algorithms in solving cases with different N



(a) N=10 (b) N=20



(c) N=30 (d) N=50

Figure 10. The path planning diagrams of the TOA algorithm with "K= 6"

References

[1] A. Antoniou, W.-S. Lu, The optimization problem, in: A. Antoniou, W.-S. Lu (Eds.), *Practical Optimization*, Springer, New York, NY, 2021, pp. 1-26.

[2] F. Tao, Y. Laili, L. Zhang, Brief history and overview of intelligent optimization algorithms, in: F. Tao, L. Zhang, Y. Laili (Eds.), *Configurable Intelligent Optimization Algorithm*, Springer, Cham, 2015, pp. 3-33.

[3] D. Whitley, A genetic algorithm tutorial, *Statistics and computing*, Vol. 4, No. 2, pp. 65-85, June, 1994.

[4] D. F. Specht, A general regression neural network, *IEEE transactions on neural networks*, Vol. 2, No. 6, pp. 568-576, November, 1991.

[5] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, Optimization by simulated annealing, *Science*, Vol. 220, No. 4598, pp. 671-680, May, 1983.

[6] F. Glover, M. Laguna, Tabu search, in: D. Z. Du, P. M. Pardalos (Eds.), *Handbook of combinatorial optimization*, Springer, Boston, MA, 1998, pp. 2093-2229.

[7] R. Poli, J. Kennedy, T. Blackwell, Particle swarm optimization, *Swarm intelligence*, Vol. 1, No. 1, pp. 33-57, June, 2007.

[8] M. Dorigo, M. Birattari, T. Stutzle, Ant colony optimization, *IEEE computational intelligence magazine*, Vol. 1, No. 4, pp. 28-39, November, 2006.

[9] S. Desale, A. Rasool, S. Andhale, P. Rane, Heuristic and meta-heuristic algorithms and their relevance to the real

world: a survey, *International Journal of Computer Engineering in Research Trends*, Vol. 2, No. 5, pp. 296-304, May, 2015.

[10] D. E. Goldberg, M. P. Samtani, Engineering optimization via genetic algorithm, *Electronic computation*, ASCE, 1986.

[11] N. B. Guedria, Improved accelerated PSO algorithm for mechanical engineering optimization problems, *Applied Soft Computing*, Vol. 40, pp. 455-467, March, 2016.

[12] D. Gao, G.-G. Wang, W. Pedrycz, Solving fuzzy job-shop scheduling problem using DE algorithm improved by a selection mechanism, *IEEE Transactions on Fuzzy Systems*, Vol. 28, No. 12, pp. 3265-3275, December, 2020.

[13] M. H. F. Zarandi, A. A. S. Asl, S. Sotudian, O. Castillo, A state of the art review of intelligent scheduling, *Artificial Intelligence Review*, Vol. 53, No. 1, pp. 501-593, January, 2020.

[14] G. A. Samra, F. Khalefah, Localization of license plate number using dynamic image processing techniques and genetic algorithms, *IEEE transactions on evolutionary computation*, Vol. 18, No. 2, pp. 244-257, April. 2014.

[15] X. Wang, J.-S. Pan, S.-C. Chu, A parallel multi-verse optimizer for application in multilevel image segmentation, *IEEE Access*, Vol. 8, pp. 32018-32030, February, 2020.

[16] A. Gopakumar, L. Jacob, Performance of some metaheuristic algorithms for localization in wireless sensor networks, *International journal of network management*, Vol. 19, No. 5, pp. 355-373, September, 2009.

[17] T. T. Nguyen, J.-S. Pan, T.-K. Dao, A compact bat algorithm for unequal clustering in wireless sensor networks, *Applied Sciences*, Vol. 9, No. 10, Article No. 1973, May, 2019.

[18] X. Wang, S.-C. Chu, H.-C. Chao, J.-S. Pan, A multi-group dragonfly algorithm for application in wireless sensor network deployment problem, *International Journal of Ad Hoc and Ubiquitous Computing*, Vol. 37. No. 4, pp. 227-239, August, 2021.

[19] T. Arora, Y. Gigras, A survey of comparison between various metaheuristic techniques for path planning problem, *International Journal of Computer Engineering & Science*, Vol. 3, No. 2, pp. 62-66, November, 2013.

[20] P.-C. Song, J.-S. Pan, S.-C. Chu, A parallel compact cuckoo search algorithm for three-dimensional path planning, *Applied Soft Computing*, Vol. 94, Article No. 106443, September, 2020.

[21] F. Zhang, T.-Y. Wu, Y. Wang, R. Xiong, G. Ding, P. Mei, L. Liu, Application of quantum genetic optimization of LVQ neural network in smart city traffic network prediction, *IEEE Access*, Vol. 8, pp. 104555-104564, June, 2020.

[22] J. M.-T. Wu, L. Sun, G. Srivastava, J. C.-W. Lin, A novel synergetic lstm-ga stock trading suggestion system in internet of things, *Mobile Information Systems*, Vol. 2021, pp. 1-15, July, 2021.

[23] G.-G. Wang, Y. Tan, Improving metaheuristic algorithms with information feedback models, *IEEE transactions on cybernetics*, Vol. 49, No. 2, pp. 542-555, February, 2019.

[24] B. Yang, J. Wang, X. Zhang, T. Yu, W. Yao, H. Shu, F. Zeng, L. Sun, Comprehensive overview of meta-heuristic algorithm applications on pv cell parameter identification, *Energy Conversion and Management*, Vol. 208, Article No. 112595, March, 2020.

[25] R. S. Parpinelli, H. S. Lopes, New inspirations in swarm intelligence: a survey, *International Journal of Bio-Inspired Computation*, Vol. 3, No. 1, pp. 1-16, February, 2011.

[26] A. Slowik, H. Kwasnicka, Nature inspired methods and their industry applications- swarm intelligence algorithms, *IEEE Transactions on Industrial Informatics*, Vol. 14, No. 3, pp. 1004-1015, March, 2018.

[27] J. Tang, G. Liu, Q. Pan, A review on representative swarm intelligence algorithms for solving optimization problems: Applications and trends, *IEEE/CAA Journal of Automatica Sinica*, Vol. 8, No. 10, pp. 1627-1643, October, 2021.

[28] S. Das, P. N. Suganthan, Differential evolution: A survey of the state-of-the-art, *IEEE transactions on evolutionary computation*, Vol. 15, No. 1, pp. 4-31, February, 2011.

[29] M. M. Eusuff, K. E. Lansey, Optimization of water distribution network design using the shuffled frog leaping algorithm, *Journal of Water Resources planning and management*, Vol. 129, No. 3, pp. 210-225, May, 2003.

[30] M. Eusuff, K. Lansey, F. Pasha, Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization, *Engineering optimization*, Vol. 38, No. 2, pp. 129-154, January, 2006.

[31] S. Mirjalili, A. Lewis, The whale optimization algorithm, *Advances in engineering software*, Vol. 95, pp. 51-67, May, 2016.

[32] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, *Advances in engineering software*, Vol. 69, pp. 46-61, March, 2014.

[33] S.-C. Chu, P.-W. Tsai, J.-S. Pan, Cat swarm optimization, *Pacific Rim international conference on artificial intelligence*, Guilin, China, 2006, pp. 854-858.

[34] J.-S. Pan, P. Hu, S.-C. Chu, Binary fish migration optimization for solving unit commitment, *Energy*, Vol. 226, pp. 120329, July, 2021.

[35] X.-S. Yang, X. He, Bat algorithm: literature review and applications, *International Journal of Bio-inspired computation*, Vol. 5, No. 3, pp. 141-149, July, 2013.

[36] Z. Meng, J.-S. Pan, H. Xu, Quasi-affine transformation evolutionary (quatre) algorithm: A cooperative swarm based algorithm for global optimization, *Knowledge-Based Systems*, Vol. 109, pp. 104-121, October, 2016.

[37] S. Arora, S. Singh, Butterfly optimization algorithm: a novel approach for global optimization, *Soft Computing*, Vol. 23, No. 3, pp. 715-734, February, 2019.

[38] A. Faramarzi, M. Heidarinejad, B. Stephens, S. Mirjalili, Equilibrium optimizer: A novel optimization algorithm, *Knowledge-Based Systems*, Vol. 191, Article No. 105190, March, 2020.

[39] J.-S. Chou, D.-N. Truong, A novel metaheuristic optimizer inspired by behavior of jellyfish in ocean, *Applied Mathematics and Computation*, Vol. 389, Article No. 125535, January, 2021.

[40] J. Xue, B. Shen, A novel swarm intelligence optimization approach: sparrow search algorithm,

Systems Science & Control Engineering, Vol. 8, No. 1, pp. 22-34, January, 2020.

[41] G. Dhiman, V. Kumar, Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems, *Knowledge-Based Systems*, Vol. 165, pp. 169-196, February, 2019.

[42] W. Zhao, Z. Zhang, L. Wang, Manta ray foraging optimization: An effective bio-inspired optimizer for engineering applications, *Engineering Applications of Artificial Intelligence*, Vol. 87, Article No. 103300, January, 2020.

[43] D. H. Wolpert, W. G. Macready, No free lunch theorems for optimization, *IEEE transactions on evolutionary computation*, Vol. 1, No. 1, pp. 67-82, April, 1997.

[44] J. A. Hartigan, M. A. Wong, Algorithm as 136: A k-means clustering algorithm, *Journal of the royal statistical society. series c (applied statistics)*, Vol. 28, No. 1, pp. 100-108, 1979.

[45] J. Liang, B. Qu, P. Suganthan, A. G. Hernández-Díaz, Problem definitions and evaluation criteria for the cec 2013 special session on real-parameter optimization, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical Report 201212, January, 2013.

[46] T. H. Kim, C. Ramos, S. Mohammed, Smart city and IOT, *Future Generation Computer Systems*, Vol. 76, pp. 159-162, November, 2017.

[47] A. Gaur, B. Scotney, G. Parr, S. McClean, Smart city architecture and its applications based on IOT, *Procedia computer science*, Vol. 52, pp. 1089-1094, 2015.

[48] L. Pammel, Botany of russian thistle, *Bulletin*, Vol. 3, No. 26, pp. 3, July, 2017.

[49] M. Crepinsek, S.-H. Liu, M. Mernik, Exploration and exploitation in evolutionary algorithms: A survey, *ACM computing surveys (CSUR)*, Vol. 45, No. 3, pp. 1-33, June, 2013.

[50] A. Lipowski, D. Lipowska, Roulette-wheel selection via stochastic acceptance, *Physica A: Statistical Mechanics and its Applications*, Vol. 391, No. 6, pp. 2193-2196, March, 2012.

[51] R. Nathan, G. G. Katul, H. S. Horn, S. M. Thomas, R. Oren, R. Avissar, S. W. Pacala, S. A. Levin, Mechanisms of long-distance dispersal of seeds by wind, *Nature*, Vol. 418, No. 6896, pp. 409-413, July, 2002.

[52] J. Wu, M. Xu, F.-F. Liu, M. Huang, L. Ma, Z.-M. Lu, Solar wireless sensor network routing algorithm based on multi-objective particle swarm optimization, *Journal of Information Hiding and Multimedia Signal Processing*, Vol. 12, No. 1, pp. 1-11, March, 2021.

[53] T. N. Tu, A fuzzy approach of large size remote sensing image clustering, *Journal of Information Hiding and Multimedia Signal Processing*, Vol. 11, No. 4, pp. 187-198, December, 2020.

[54] X. Kou, J. Feng, Matching ontologies through compact monarch butterfly algorithm, *Journal of Network Intelligence*, Vol. 5, No. 4, pp. 191-197, November, 2020.

Biographies

Jeng-Shyang Pan received the B.S. degree in electronic engineering from the National Taiwan University of Science and Technology in 1986, the M.S. degree in communication engineering from National Chiao Tung University, Taiwan, in 1988, and the Ph.D. degree in electrical engineering from the University of Edinburgh, U.K., in 1996. He is currently the Professor of Shandong University of Science and Technology. He is the IET Fellow, U.K., and has been the Vice Chair of the IEEE Tainan Section and Tainan Chapter Chair of IEEE Signal Processing Society. His current research interest includes the information hiding, artificial intelligence and wireless sensor networks.

Qingyong Yang received the B.S. degree from the Shandong University of Science and Technology, Qingdao, China, in 2020. He is currently pursuing the master's degree with the Shandong University of Science and Technology, Qingdao, China. His recent research interests include swarm intelligence, intelligent scheduling and transportation problems.

Chin-Shiuh Shieh received the M.S. degree in electrical engineering from the National Taiwan University, Taipei, Taiwan, in 1991 and the Ph.D. degree from the Department of Computer Science and Engineering, National Sun Yat-Sen University, Kaohsiung, Taiwan, in 2009. In August 1991, he joined the faculty of the Department of Electronic Engineering, National Kaohsiung University of Science and Technology, Kaohsiung, where he is currently an Associate Professor. His research interests include computer networking, embedding system, computational intelligence and information security.

Shu-Chuan Chu received the Ph.D. degree from the School of Computer Science, Engineering and Mathematics, Flinders University, Australia, in 2004. She joined Flinders University, in December 2009, after nine years at Cheng Shiu University, Taiwan. She has been a Research Fellow with the College of Science and Engineering, Flinders University, since December 2009. She has been a Research Fellow, with PhD advisor, at the College of Computer Science and Engineering, Shandong University of Science and Technology, since September 2019. Her research interests mainly include swarm intelligence, intelligent computing, and data mining.