2021 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/IOP/SCI)

2021 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/IOP/SCI) | 978-1-6654-1236-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/SWC50871.2021.00034

A Route Optimization Scheme based on Improved Simulated Annealing Algorithm

1st Chenyan Sun
School of Computer Science
China University of Geosciences
Wuhan, China

2nd Xiaohan Hao
School of Computer Science
China University of Geosciences
Wuhan, China

3nd Wei Ren*
School of Computer Science
China University of Geosciences, Wuhan, China
Henan Key Laboratory of Network Cryptography Technology, Zhengzhou, China
Key Laboratory of Network Assessment Technology, CAS
(Institute of Information Engineering,
Chinese Academy of Sciences, Beijing, China 100093)
weirencs@cug.edu.cn

***Abstract*—The simulated annealing (SA) algorithm can be employed to solve the traveling salesman problem (TSP). When the scale of the problem increases, the convergence of SA will be extremely slow. To address this issue, we propose an improved SA (ISA) algorithm, which can adaptively search an optimal solution with multi-modal pathway variants, and introduces jump-cooling and quadratic variation mechanism during the search process to improve the convergence speed. A quadratic annealing is also introduced at the end of the algorithm to prevent from falling into a local optimum. A historical optimal solution memory is proposed to ensure that the optimal solution will not be lost due to probabilistic acceptance of poorer solutions. The results of comparative experiments on different data sets of TSP benchmarks, which are obtained from TSPLIB, justify that ISA outperforms SA as well as other heuristics in terms of faster convergence speed and higher search accuracy. We also evaluate and show the efficiency and optimization of ISA by real data, e.g., Hebei province tour data.**

***Index Terms*—Tourist route optimization, TSP, Simulated annealing algorithm, Optimization**

I. Introduction

The Traveling Traders Problem [1] was first proposed in 1930 and is one of the most important problems in combinatorial optimization. In real life, the TSP model is used in various fields such as circuit logistics management [2], production scheduling [3] and chip design [4]. Given the number of cites n, a travelling merchant needs to pass through all of the cities on his way, but only once, and then return to the city from which he started. How can the route be planned so that the route taken by the merchant is the shortest.

Conventional classical optimization algorithms were often used to solve the TSP problem. However, when the number of cities increases greatly, they became difficult to search the optimal solution quickly. With the development of intelligent algorithms, many problem-independent stand-alone algorithms have emerged, e.g., simulated annealing algorithm [5], ant colony algorithm [6], particle swarm algorithm [7], genetic algorithm [8], fish swarm algorithm [9], wolf swarm algorithm [10] and so on. These algorithms simulate certain phenomena in nature to derive new ideas and methods for solving complex problems. Although these methods may be used to solve TSP, it is still unsatisfactory with respect to either speed or quality of solution when the number of cities increases.

In this paper, we propose an improved simulated annealing algorithm to solve the problem of planning tourist routes to the main attractions in Hebei Province. The contributions of the paper are listed as follows:

1) An adaptive new solution generator and multiple path variants are proposed to improve the search efficiency for the optimal solution.
2) A jump-cooling and quadratic variation mechanism is proposed to improve the convergence speed of the algorithm, followed by a quadratic annealing to prevent from falling into a local optimum.
3) A historical optimal solution memory is proposed to guarantee the optimal solution will not be lost due to probabilistic acceptance of poorer solutions.

The rest of the paper is organized as follows. Section II presents research work in the relevant literature. Section III presents relevant background of simulated annealing algorithm. The TSP problem is stated formally in Section IV. Section V gives the details of our proposed scheme. Section VI provides some experimental results and evaluation analysis. Section VII applies the algorithm proposed in this paper to a practical problem. Finally, Section VIII concludes the paper.

II. Related Work

TSP, a well-known NP-hard problem, has two main types of solutions from the time it was proposed. One is the conventional stability algorithm, which can find the exact global optimal solution without considering time and space, e.g., dynamic programming method [11], which transforms a multi-stage

978-1-6654-1236-0/21/$31.00 ©2021 IEEE
DOI 10.1109/SWC50871.2021.00034

process into a series of single-stage problems and solves them one by one. However, as the size of the problem increases, the space required by this type of algorithm increases dramatically and therefore cannot solve large-scale TSP. The other are modern heuristic algorithms. Genetic algorithm [8] simulates the computational model of the biological evolutionary process of Darwinian biological evolution with natural selection and genetic mechanisms. Artificial neural network algorithm [12] stores information in a distributed manner and processes it collaboratively in parallel by simulating the structure of the human brain. Ant colony algorithm [6] generates new choices by simulating the collective path-finding behavior of ants in nature, adaptively searching for new paths based on a positive feedback mechanism with pheromone concentration. Although these methods cannot be solved exactly, they are able to control the error within a tolerable range and obtain the answer relatively quickly. During the past three years, the methods proposed to solve the TSP problem fall into two general directions. One is the optimization of a particular heuristic algorithm, the other is the hybrid of different heuristic algorithms.

A self-adaptive ACO algorithm (DEACO) [13] is proposed in 2020, which first introduces a new pheromone matrix to intelligently select the initial nodes of each iteration in the search space. Thus, speeding up the evolution while avoiding jumping trapped in a local optimum. The parameters of the ant colony algorithm are afterward dynamically adjusted to improve the uncertain convergence time and stochastic decision making ability of the algorithm. Experiments on the TSPLIB public dataset show that DEACO has better performance in terms of convergence speed and search accuracy compared to the conventional ACO.

A hybrid algorithm named ant colony-partheno genetic algorithm (AC-PGA) [14] is proposed based on partheno genetic algorithm (PGA) and ACO. The method firstly divides the parameters into two parts - initial and intermediate, and uses PGA to determine the optimal values for initial locations of salesmen and number of cities each visits. ACO thereafter is used to determine the shortest route for each salesman, thus the global optimal solution to the problem can be obtained. By comparing this method with five other different heuristics under different number of cities, the results demonstrate that the algorithm outperforms the other algorithms regardless of the number of the cities.

MEATSP [15], a new heuristic algorithm proposed in 2020, is based on MEAF [15], which includes four operators derived from living cells: division, fusion, cytolysis and selection. This algorithm obtains the optimal solution of a problem through the evolution of membrane structure and objects within the membrane. After initialisation, each membrane is iterated with the population to obtain a new population through four operators respectively. The optimal membrane is then replaced by the membrane with the highest fitness in the population prior to iteration. The proposed algorithm performs well in both accuracy and average residual for problems of different sizes in TSPLIB, and is proved to be a stable algorithm.

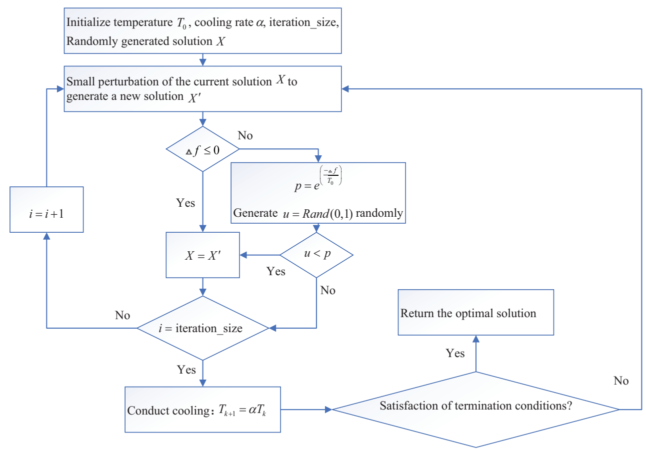


Fig. 1: conventional simulated annealing algorithm

In 2020, the article [16] implemented four improvement strategies on brain storm optimization algorithm(BSO) [17] and proposed an agglomerative greedy brain storm optimization algorithm (AG-BSO). The AG-BSO firstly introduced a greedy algorithm to ensure the diversity of the population, then replaced the k-means clustering algorithm in standard BSO with hierarchical clustering to eliminate the noise sensitivity of the original BSO algorithm when solving TSP, while introducing exchange rules for individuals within the population to improve the efficiency of the algorithm, and finally, a heuristic crossover operators was used to update individuals. The performance of AG-BSO was proved by comparing with other six algorithms on standard TSP data sets.

III. Preliminary

The simulated annealing (SA) algorithm draws on the mechanism of solid annealing. When a solid is hot, the internal particles are disordered and active. As it slowly cools down, the internal particles become progressively more ordered. Given an initial value as the current solution, and some of the elements contained in this solution are transformed to produce a large number of new solutions, among which the SA will select a solution as the current new one. In the process of selecting a new solution, the SA will search in the neighborhood of a poor solution, which provides a new direction for finding the global optimal solution and makes it more likely for SA to jump out of a local optimum. The probability of sudden jumpiness decreases as the temperature decreases. When the termination condition is reached, the solution obtained by the algorithm is the global optimal solution. Fig. 1 depicts the flow of SA.

IV. Problem Formulation

Being a typical combinatorial optimization problem, the mathematical model of the TSP can be represented as follows:

Let $G = (V, E)$ be an assignment graph; $V = \{1, 2, \cdots, n\}$ is the set of vertices; E is the set of edges; d_{ij} $(d_{ij} > 0, d_{ii} = +\infty, i, j \in V)$ is the distance between vertices and

$$x_{ij} = \begin{cases} 1 & edge(i,j) \ on \ optimal \ route \\ 0 & else \end{cases}$$

The mathematical model of TSP can be written in the form of a linear program as follows:

$$\min \ Z = \textstyle\sum_{i \neq j} d_{ij} \cdot x_{ij}$$

$$\text{s. t} \quad \begin{cases} \sum\limits_{j \neq i} x_{ij} = 1 & i \in V \\ \sum\limits_{i \neq j} x_{ij} = 1 & j \in V \\ \sum\limits_{i,j \in S} x_{ij} \leq |S| - 1 & for \ all \ subsets \ S \ of \ V \\ x_{ij} \in \{0,1\} & i,j \in V \end{cases}$$

Here, $|S|$ is the number of vertices of the graph G contained in the set S. The first two constraints imply that for each vertex, there is only one edge in and one edge out, and the latter constraint guarantees that no subloop solution is generated. Thus, the solutions satisfying the above constraint form a Hamilton loop.

The notations will be used in the following presentations as follows:

1) T_0: Initial temperature to start the search;
2) T_f: Termination temperature for the end of the search;
3) α: Temperature update factor, i.e. $T_{i+1} = \alpha \times T_i$;
4) $iteration$: Number of searches at the same temperature;
5) N: Energy threshold for jump cooling and quadratic variation;
6) M: Current energy level;
7) x: A solution to the problem, i.e. a closed circuit connecting all cities;
8) x_{best}: The optimal solution path searched so far;
9) $f(x)$: A function returning a new solution path through multi-modal pathway variants based on x, which will be depicted in algorithm 1;
10) $value(x)$: The heuristic function returning the value of Z based on a solution path x.

Definition Given a specific location map, find a x with the minimum $value(x)$.

V. Proposed Scheme

In this section, we will describe our proposed schemes from multi-modal pathway variants, jump cooling and quadratic variation, quadratic annealing and historical optimal solution memory. Since the simulated annealing (SA) algorithm has a powerful global search capability for TSP problem, most of current schemes propose to add operators or combine SA with other heuristics to improve the accuracy of final solutions. However, when the scale of the problem increases, the efficiency of these schemes will be extremely slow. To address this issue, we discuss the implementation process of SA and propose the following schemes to improve the convergence speed.

A. Multi-modal pathway variants

In SA algorithm, new solutions are generated by randomly selecting two cities from the current path and swapping the positions of them, which will result in a low degree of solution updates. Besides, the new path will be generated in only one direction due to the single path variant, thus the optimal solution can only be applied to the specified path. To tackle above challenges, we propose two novel path variants and put them into the function $f(x)$.

1) Exchange: Generate three integers $X_1, X_2, X_3 (X_1 < X_2 < X_3)$ randomly, and swap path segments on the interval $[X_1, X_2]$ and $[X_2, X_3]$.
2) Invert: Generate two integers $X_1, X_2 (X_1 < X_2)$ randomly, and invert path segments on the interval $[X_1, X_2]$.

It is worth to note that these above path variants can be applied to an self-adaptive new solution generator, which is also included in the function $f(x)$. The detailed design of the function $f(x)$ is shown in algorithm 1. We assume that the current temperature is T and current solution path is x, and the local region where x is located will be searched repeatedly until the number of successes reaches $0.01 \times T$. Besides, as the temperature decreases, we observe that the threshold of successful searches will reduce. Thus, a smart search for the global solution should be proposed.

B. Jump cooling and quadratic variation

We observe that the degree of cooling is positively correlated with the frequency of state acceptance at a certain temperature, thus the speed of generating new paths should be improved especially in the high temperature environment. To avoid using the roundabout search for states, we propose a specific jump cooling and quadratic variation in a round of iterative search process, which is shown in algorithm 2. During the jump cooling and quadratic variation in a round of iterative search process, once a better solution is searched and accepted, the current energy value will be increased by one. After completing a round of iterative search, if the current energy value is higher than the set threshold, a jump cooling and second variation will be performed. Therefore, the convergence speed of SA will be increased significantly through this proposed process.

C. Quadratic annealing and historical optimal solution memory

Although jump cooling and quadratic variation can improve the convergence speed of SA, the possibility of falling into a local optimum will also increase. Therefore, we propose a mechanism of quadratic annealing. In this mechanism, after executing the whole annealing process, the obtained final solution path will be annealed again as an initial path, which can jump out the local optimum.

Besides, during the search process of SA, there is a possibility of losing the optimal solution due to the probabilistic acceptance of poorer solutions. To tackle this challenge, we set up a historical optimal solution memory to save the past

```
Data: Current solution path x, Current temperature T,
      City_Scale num
Result: New solution path x'
M ⇐ 0;
M' ⇐ 0.01 × T;
for (i = 0; M <= M'; i = i + 1) do
    p ⇐ random();
    if p <= 0.5 then
        X1 = randint(0, num − 3);
        X2 = randint(X1 + 1, num − 2);
        X3 = randint(X2 + 1, num − 1);
        X3 = randint(X2 + 1, num − 1);
        x'[0 : X1] ⇐ x[0 : X1];
        x'[X1+1 : X3−X2+X1] ⇐ x[X2+1 : X3+1];
        x'[X3 + 1 : num − 1] ⇐ x[X3 + 1 : num − 1];
        if value(x') < value(x) then
            x ⇐ x';
            M ⇐ M + 1;
        end
    end
    else
        X1 = randint(0, num − 2);
        X2 = randint(X1 + 1, num − 1);
        x'[0 : X1] ⇐ x[0 : X1];
        x'[X1 + 1 : X2] ⇐ x[X2 : X1 : −1];
        x'[X2 + 1 : num − 1] ⇐ x[X2 + 1 : num − 1];
        if value(x') < value(x) then
            x ⇐ x';
            M ⇐ M + 1;
        end
    end
end
x' ⇐ x;
return x';
```

Algorithm 1: The function $f(x)$ to generate a new solution path

```
Data: Current solution path x, Current temperature T
Result: New solution path x', Updated temperature T'
M ⇐ 0;
for (i = 0; i <= iteration; i = i + 1) do
    x' ⇐ f(x);
    if value(x') <= value(x) then
        x ⇐ x';
        M ⇐ M + 1;
    end
    if value(x') > value(x) then
        Δv ⇐ value(x') − value(x);
        p ⇐ random();
        if p < e^(−Δv/T) then
            x ⇐ x';
        end
    end
end
x' ⇐ x;
T' ⇐ α × T;
if M >= N then
    T' ⇐ α × T';
    x' ⇐ f(x);
    while value(x') > value(x) do
        x' ⇐ f(x);
    end
end
return x' and T';
```

Algorithm 2: Jump cooling and quadratic variation in a round of iterative search process

searched optimal solution. Each time a solution update is performed, i.e. $x \Leftarrow x'$, it will compare $vlaue(x_{best})$ with $value(x')$, if $vlaue(x_{best}) > value(x')$, then let $x_{best} \Leftarrow x'$, else do not update the value of x_{best}.

D. Proposed Algorithm

In this section, we will describe the improved simulated annealing algorithm (ISA), which is shown in Fig. 2. Compared with SA, our proposed ISA can iteratively search at each temperature more efficiently. The specific description is as follows:

- In the process of generating a new solution, multimodal searches are performed adaptively according to the current temperature.
- ISA decide whether to perform jump cooling and quadratic variation based on the number of successful merit selections.
- The solution obtained by the first annealing will be used as an initial path for the second annealing.
- A historical optimal solution memory is set up to save the past searched optimal solution.

VI. Analysis and Experiment Results

In this section, experiments conducted on TSP to test the improved simulated annealing (ISA) algorithm will be introduced in detail. For TSP datasets, the 8 instances used in this article come from the TSPLIB benchmark. The number of city nodes in these 8 instances ranges from 14 to 1748. We will analyze our experiment results to prove the feasibility of proposed scheme. The specific experimental process is described as follows. Firstly, SA and ISA are run 50 times respectively on the st70 public dataset, with the termination temperature $T_f = 0.001°C$ and temperature update factor $\alpha = 0.1$. The final solution paths they obtain are shown in Fig. 3. From Fig. 3 we can see that the distance of final path SA gets is 2803 while that of ISA is 705. Fig. 4 shows the specific variation of solution for both algorithms, with the number of current iterations as horizontal axis and the distance of current path as vertical axis. We use the distance of the final path to measure the quality of the solution, the shorter the distance, the higher the quality of the solution. The results show that when the city scale is 70, the convergence speed of ISA is 2

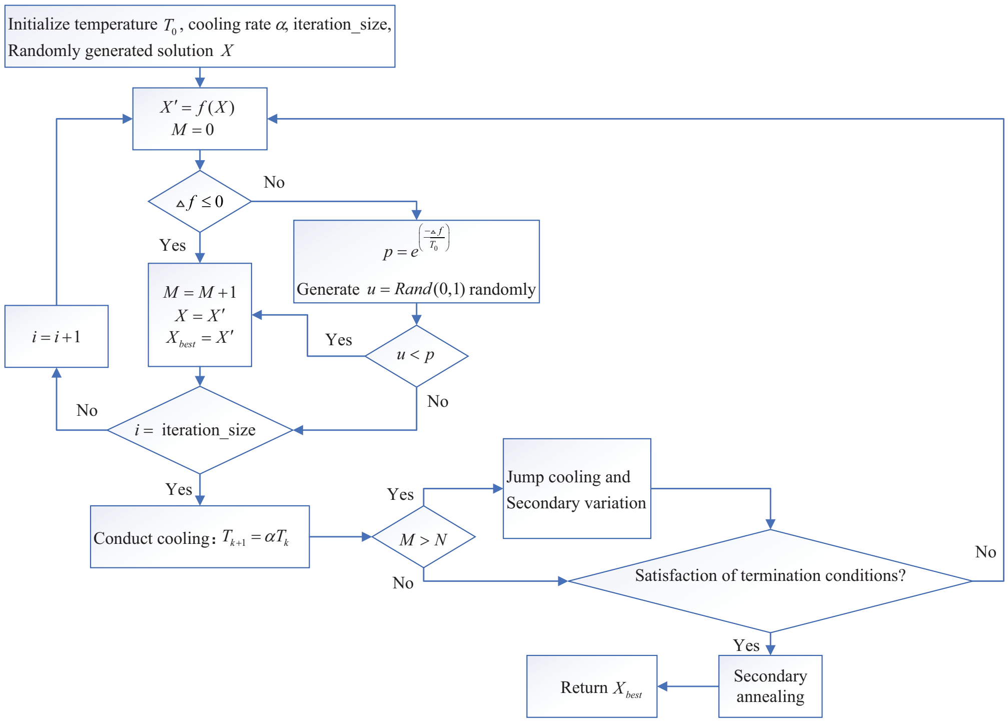
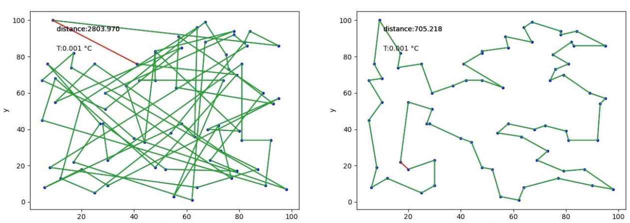


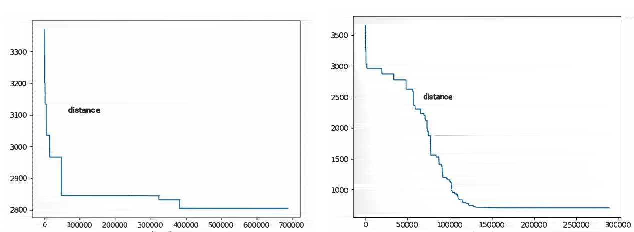
Fig. 2: Improved simulated annealing algorithm

times that of SA, and the quality of final solution is 4 times that of SA.



(a) conventional algorithmm (b) Improved algorithm

Fig. 3: Final path from both algorithms



(a) conventional algorithmm (b) Improved algorithm

Fig. 4: Variation of the solution of both algorithms

The results of two algorithms run on all the 8 datasets of TSPLIB with $T_f = 0.001°C$ and $\alpha = 0.1$ are shown in the Tab. I. The results show that when city scale is under 250, the quality of the final ISA solution is slightly better than that of the SA. However, in terms of the speed of convergence, ISA is significantly better than SA, up to 4 times faster than SA. When city scale is above 1000, we can see that the convergence speed of ISA is 3 times that of SA, and the quality of final solution is 2 times that of SA.

TABLE I: Results of the TSP public dataset runs

Test dataset	Average solution		Average maximum number of iterations	
	Before improvements	After improvement	Before improvements	After improvement
burma14	3323	3323	2385	1625
att48	10976	10854	461040	196644
berlin52	7580	7577	762000	344540
st70	681	680	1309000	302430
kroA100	21476	21459	806420	347020
tsp225	4120	4014	1162678	841000
vm1084	559127	281639	29372831	9972831
vm1748	843429	412783	37679105	10253378

In addition to comparing with SA, we also compare ISA with other heuristic algorithms. Specifically, We run ISA, genetic algorithm (GA) [8], ant colony algorithm (ACO) [6] and particle swarm algorithm (PSO) [7] on TSPLIB datasets 50 times respectively, and the corresponding results of each algorithm are shown in Tab. II, Tab. III, Tab. IV, Tab. V, Tab. VI, Tab. VII, Tab. VIII, and Tab. IX.

The results show that with the number of cities increases, the quality of the final solution and the execution time of ISA is better than other heuristic algorithms.

TABLE II: Algorithms comparison for burma14

Algorithm	City size	Reference	Best	Mean	Variance	Average time
ISA	14	3323	3323	3323	0	**4.234 Sec**
GA [8]	14	3323	3323	3323	0	5.187 Sec
ACO [6]	14	3323	3323	3323	0	5.342 Sec
PSO [7]	14	3323	3323	3323	0	5.406 Sec

TABLE III: Algorithms comparison for att48

Algorithm	City size	Reference	Best	Mean	Variance	Average time
ISA	48	10628	10854.12	11032.87	**31684.12**	**14.913 Sec**
GA [8]	48	10628	**10832.58**	**10987.19**	34182.59	16.519 Sec
ACO [6]	48	10628	10968.93	11082.31	41362.27	16.418 Sec
PSO [7]	48	10628	11004.25	11132.28	38753.63	17.382 Sec

TABLE IV: Algorithms comparison for berlin52

Algorithm	City size	Reference	Best	Mean	Variance	Average time
ISA	52	7542	**7577.13**	**7589.28**	**88.72**	**15.236 Sec**
GA [8]	52	7542	7591.62	7601.31	113.64	18.812 Sec
ACO [6]	52	7542	7601.34	7614.79	216.68	19.251 Sec
PSO [7]	52	7542	7598.72	7612.43	167.98	18.195 Sec

TABLE V: Algorithms comparison for st70

Algorithm	City size	Reference	Best	Mean	Variance	Average time
ISA	70	675	**680.12**	**698.28**	196.87	**19.832 Sec**
GA [8]	70	675	712.28	734.31	213.34	23.183 Sec
ACO [6]	70	675	696.97	726.79	206.51	22.647 Sec
PSO [7]	70	675	693.54	708.43	**187.56**	23.196 Sec

TABLE VI: Algorithms comparison for kroA100

Algorithm	City size	Reference	Best	Mean	Variance	Average time
ISA	100	21282	**21459.83**	**21532.19**	**5329.13**	**30.137 Sec**
GA [8]	100	21282	21512.17	21653.13	6458.93	42.628 Sec
ACO [6]	100	21282	21559.62	21691.56	5819.28	39.192 Sec
PSO [7]	100	21282	21614.39	21758.37	6124.73	47.619 Sec

TABLE VII: Algorithms comparison for tsp225

Algorithm	City size	Reference	Best	Mean	Variance	Average time
ISA	225	3916	**4014.16**	**4097.34**	**6876.83**	**64.009 Sec**
GA [8]	225	3916	4062.58	4112.74	7058.17	85.172 Sec
ACO [6]	225	3916	4104.27	4187.18	6912.23	80.492 Sec
PSO [7]	225	3916	4096.85	4172.49	7824.73	97.105 Sec

TABLE VIII: Algorithms comparison for vm1084

Algorithm	City size	Reference	Best	Mean	Variance	Average time
ISA	1084	239297	**281639**	**303170**	**91027**	**329.138 Sec**
GA [8]	1084	239297	298621	335872	101389	614.297 Sec
ACO [6]	1084	239297	310268	339681	128023	537.841 Sec
PSO [7]	1084	239297	293831	320972	994738	727.268 Sec

TABLE IX: Algorithms comparison for vm1748

Algorithm	City size	Reference	Best	Mean	Variance	Average time
ISA	1748	336556	**399783**	**423697**	**109358**	**597.291 Sec**
GA [8]	1748	336556	413726	439618	125726	1483.108 Sec
ACO [6]	1748	336556	420819	441927	141203	1175.386 Sec
PSO [7]	1748	336556	409651	437204	119478	1230.914 Sec

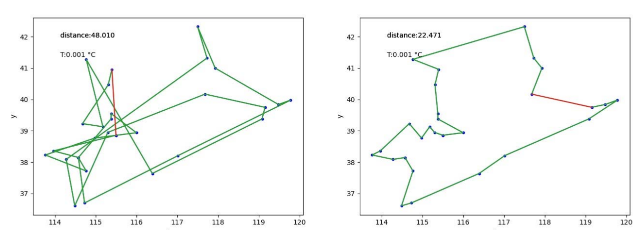
VII. CASE STUDY

After proving that ISA is superior to other heuristic algorithms in terms of convergence speed and solution quality, especially when the city scale is large. We consider to apply it to solving some practical problems. Therefore, the ISA is applied to planning tourist routes in Hebei Province. We collect 30 major attractions in Hebei province and use them to construct a city map of the TSP problem. Tab. X lists the locations of 30 major attractions in Hebei Province.

TABLE X: Coordinates of 30 tourist attractions in Hebei Province

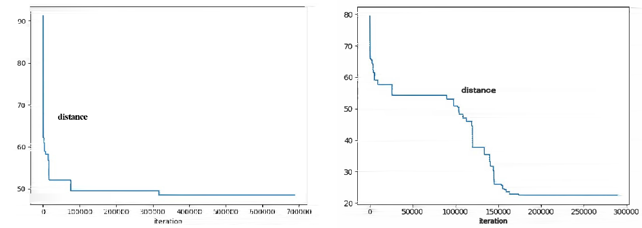
Number	Name of attraction	latitude	longitude
1	Shanhaiguan	119.7761	39.9789
2	Beidaihe	119.4846	39.8351
3	Wolfsbane	115.1779	39.1293
4	White Rock Mountain	114.6835	39.2328
5	Nansanpo	115.3798	39.5491
6	Baiyangdian	115.9953	38.9402
7	Dong Martyrs'Cemetery	117.727	41.3213
8	Chengde Summer Resort	117.9303	40.9975
9	Sehamba Forest Park	117.4964	42.319
10	Former Leting County	119.0849	39.385
11	Jieshi Mountain Scenic	119.1488	39.7515
12	Jieming Mountain	115.305	40.4668
13	Chongli Wanlong Ski	115.3955	40.9603
14	Zhangbei Grassland	114.768	41.2848
15	Qing Dongling	117.6787	40.1684
16	Handan Guangfu City	114.7326	36.699
17	Zhaozhou Bridge	114.7687	37.7219
18	Wuqiao Acrobatic World	116.3878	37.6398
19	Cangzhou Iron Lion	117.0223	38.2064
20	Handan Congtai Park	114.491	36.6146
21	Xibaipo Scenic Area	113.9718	38.3598
22	Longxing Temple	114.5827	38.1441
23	Rongguo Mansion	114.5794	38.1476
24	Hugh Calf	114.2757	38.0877
25	Tiangui Mountain	113.7662	38.2322
26	Baoding Park	115.4974	38.8595
27	Ancient Lotus Pond	115.4982	38.8574
28	Mancheng Han Tomb	115.2977	38.9466
29	Qingsi Ling	115.3846	39.38
30	Baikouen Ko Memorial Hall	114.9847	38.7699

Specifically, we use the latitude of these sites as their x-value and the longitude as their y-value to construct the city location distribution map for TSP problem. Next, we run SA and ISA 50 times with $T_f = 0.001°C$ and $\alpha = 0.1$ on this map respectively. Fig. 5 shows the final paths obtained by these two algorithms, and the variation of the solution for both algorithms are shown in Fig. 6. From Fig. 5 we can see that the distance of final path SA gets is 48.01 while that of ISA is 22.471. From Fig. 6 we can see that the iteration of SA is approximately 330,000 while that of ISA is 160,000.



(a) conventional algorithmm (b) Improved algorithm

Fig. 5: Final path from both algorithms



(a) conventional algorithmm (b) Improved algorithm

Fig. 6: Variation of the solution of both algorithms

Experimental results show that when dealing with the problem of planning tourist routes to the main attractions in Hebei Province, the convergence speed of ISA is 2 times that of SA, and the quality of final solution is 2 times that of SA. Thus, we can conclude that ISA can optimize travel routes more efficiently.

VIII. Conclusion

In this paper, we propose an improved simulated annealing algorithm called ISA to solve the travel route optimization problem. Compared to conventional simulated annealing (SA) algorithm, ISA selects an adaptive new solution generator with multi-modal pathway variants and a historical optimal solution memory to improve the search efficiency for the optimal solution. ISA also introduces jump-cooling and quadratic variation mechanism to improve the convergence speed of the algorithm, followed by a quadratic annealing at the end of the algorithm to prevent from falling into a local optimum. The performance of ISA is evaluated extensively on different sets of TSP benchmarks obtained from TSPLIB. The experiment results illustrates that compared to SA, ISA has faster convergence and better accuracy. Also, ISA outperforms on large-scale TSP problems than other heuristics. ISA is also evaluated in the real data on optimizing travel routes of 30 major attractions in Hebei Province of China, which further demonstrates the ability of the algorithm on solving realworld problems.

IX. Data statement

The source code for this paper can be open accessed and downloaded from https://github.com/SCYan-CUG/A-Route-Optimization-Scheme-based-on-Improved-Simulated-Annealing-Algorithm.

Acknowledgement

The research was financially supported by National Natural Science Foundation of China (No. 61972366), the Provincial Key Research and Development Program of Hubei (No. 2020BAB105), the Foundation of Henan Key Laboratory of Network Cryptography Technology (No. LNCT2020-A01), and the Foundation of Key Laboratory of Network Assessment Technology, Chinese Academy of Sciences (No. KFKT2019-003).

References

[1] Bahareh Bahrami, Mohammad Ali Jabraeil Jamali, and Shahram Saeidi. A hierarchical architecture based on traveling salesman problem for hybrid wireless network-on-chip. *Wireless Networks*, 25(5):2187–2200, 2019.

[2] Jose B Escario, Juan F Jimenez, and Jose M Giron-Sierra. Ant colony extended: experiments on the travelling salesman problem. *Expert Systems with Applications*, 42(1):390–410, 2015.

[3] Georgios P Georgiadis, Georgios M Kopanos, Antonis Karkaris, Harris Ksafopoulos, and Michael C Georgiadis. Optimal production scheduling in the dairy industries. *Industrial & Engineering Chemistry Research*, 58(16):6537–6550, 2019.

[4] Pouya Baniasadi, Mehdi Foumani, Kate Smith-Miles, and Vladimir Ejov. A transformation technique for the clustered generalized traveling salesman problem with applications to logistics. *European Journal of Operational Research*, 285(2):444–457, 2020.

[5] Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[6] Marco Dorigo and Luca Maria Gambardella. Ant colonies for the travelling salesman problem. *biosystems*, 43(2):73–81, 1997.

[7] Xiaohu H Shi, Yanchun Chun Liang, Heow Pueh Lee, C Lu, and QX Wang. Particle swarm optimization-based algorithms for tsp and generalized tsp. *Information processing letters*, 103(5):169–176, 2007.

[8] David S Johnson and Lyle A McGeoch. The traveling salesman problem: A case study in local optimization. *Local search in combinatorial optimization*, 1(1):215–310, 1997.

[9] Mehdi Neshat, Ghodrat Sepidnam, Mehdi Sargolzaei, and Adel Najaran Toosi. Artificial fish swarm algorithm: a survey of the state-of-the-art, hybridization, combinatorial and indicative applications. *Artificial intelligence review*, 42(4):965–997, 2014.

[10] Seyedali Mirjalili, Seyed Mohammad Mirjalili, and Andrew Lewis. Grey wolf optimizer. *Advances in engineering software*, 69:46–61, 2014.

[11] Chryssi Malandraki and Robert B Dial. A restricted dynamic programming heuristic algorithm for the time dependent traveling salesman problem. *European Journal of Operational Research*, 90(1):45–55, 1996.

[12] Hassan Ghaziri and Ibrahim H Osman. A neural network algorithm for the traveling salesman problem with backhauls. *Computers & Industrial Engineering*, 44(2):267–281, 2003.

[13] Chao Jiang, Zhongping Wan, and Zhenhua Peng. A new efficient hybrid algorithm for large scale multiple traveling salesman problems. *Expert Systems with Applications*, 139:112867, 2020.

[14] Sahar Ebadinezhad. Deaco: Adopting dynamic evaporation strategy to enhance aco algorithm for the traveling salesman problem. *Engineering Applications of Artificial Intelligence*, 92:103649, 2020.

[15] Ping Guo, Mengliang Hou, and Lian Ye. Meatsp: A membrane evolutionary algorithm for solving tsp. *IEEE Access*, 8:199081–199096, 2020.

[16] Changyou Wu and Xisong Fu. An agglomerative greedy brain storm optimization algorithm for solving the tsp. *IEEE Access*, 8:201606–201621, 2020.

[17] Yuhui Shi. Brain storm optimization algorithm. In *International conference in swarm intelligence*, pages 303–309. Springer, 2011.