A Variable Neighborhood Differential Evolutionary Algorithm for solving Multiple Travelling Salesman Problem

Jinhua Bian[1], Xiaoxia Zhang[2]

[1]School of Computer Science and Software Engineering, University of Science and Technology LiaoNing, Anshan114051 , China,17816066868@163.com

[2]School of Computer Science and Software Engineering, University of Science and Technology LiaoNing, Anshan114051,China , zhangxiaoxia@ustl.edu.cn

Abstract
The Multiple Travelling Salesman problem (MTSP) is a variant of the Travelling Salesman problem,and it is also a typical combination optimization problem that needs to be approximated and solved using intelligent optimization algorithms represented by DE algorithms.However, typical DE algorithms are prone to local precocity and inaccurate arithmetic results. Therefore, we propose an improved differential algorithm using Variable Neighborhood Search to solve the multiple depot version of the MTSP problem, using variable neighborhood algorithms consisting of 2-opt, double bridge exchange and 3-opt to improve the local search ability of the differential evolution algorithm, and the experimental data proves the effectiveness of the improved differential algorithm in solving the multiple depot MTSP problem, and it can be widely used in the logistic transportation and other industry.
Keywords: **Multiple Travelling Salesman Problem, Differential Evolutionary Algorithm, Variable Neighborhood *Search***

1. Introduction

Among the MTSP problems, the multiple depot version of the MTSP problem[1] is a research problem with relatively few academic achievements, it can be described as follows: there are *m* travelers who need to visit *n* cities from *m* different cities, respectively, visit a certain number of cities and return to their respective starting points, in which each city needs to be and can only be visited by a certain salesman once, the goal of the multiple depot MTSP problem is when all the travelers have finished visiting the cities, the path sum of the cities visited by each traveler is minimized. The multiple depot MTSP problem is a typical NP-hard problem, and the differential evolution (DE) algorithm is a typical meta-heuristic algorithm that can effectively approximate the solution of the multiple depot MTSP problem. In previous academic research, there are many researchers who improve the DE algorithm for MTSP problems,Yun Bai[2] and others use parallel preprocessing of differential granularity and construction of cross-evolutionary operational hierarchies to realize multilevel, multi-objective bi-directional traveler computation; Huiren Zhou[3] and others achieve more accurate solution of MTSP problems by improving the crossover, mutation, and selection strategies of the DE algorithm;Banu Soylu[4] proposed a general-purpose variable neighborhood algorithm that uses different neighborhoods for searching, and uses its powerful local search ability to achieve the effect of optimizing the problem solution; Donald Sofge[5] and others used a variant of K-means neighborhood attraction model, compared various evolutionary algorithms and paradigms, and finally took the optimal solution to MTSP problem. The research of multiple depot MTSP problem is of great significance to the logistics and transportation industry.
In this paper, for the weakness that differential evolutionary algorithms are easy to converge locally and fail to obtain the optimal solution, an improved DE algorithm VNS&DE based on Variable Neighborhood Search(VNS) is proposed, which improves the shortcoming of DE algorithm that is easy to fall into the local optimum by using the Variable Neighborhood Search algorithm consisting of the Shaking algorithm, the 2-opt algorithm, the Kernighan-Lin and the 3-opt algorithm, so that the VNS&DE algorithm is able to solve the multiple depot MTSP problem more accurately and obtain the optimal solution.

2. Algorithm Design

2.1 Basic theory of Differential Evolutionary Algorithms

Like other evolutionary algorithms, Differential evolution algorithm is a model that simulates biological competition and evolution.The overall structure of DE algorithm is similar to genetic algorithm, which simulates the realization of biological evolution and iteration through the basic operations of crossover, mutation and selection. Different from the genetic algorithm, DE algorithm links the crossover, mutation and selection operations to form its unique memory search capability, which can adjust the search strategy in time and has a strong global search capability. For the mutation operation, DE algorithm starts from a randomly generated initial population, uses the difference vector of two individuals randomly selected from the population as the random change source of the third individual, weights the difference vector and sums it with the third individual in accordance with certain rules to generate the mutated individual; for the crossover operation, DE algorithm mixes the mutated individual with a predetermined target individual for parameters to generate the test individual. For the crossover operation, the DE algorithm mixes the variant individuals with a predetermined target individual to generate test individuals. For selection operation, if the fitness value of the test individual is better than that of the target individual, the test individual replaces the target individual in the next generation, otherwise the target individual is still preserved. The specific pseudo code of the difference algorithm is shown in Algorithm 1.

Algorithm 1: DE algorithm
Input: Population:P；Dimension:D；Generation:R；
Output: The Best solution path: BestPath

```
1:  r←1(initialization):
2:  for i = 1 to P do
3:      for j=1 to D do
4:          $x_{i,r}^{j} = x_{\min}^{j} + rand(0,1)\cdot(x_{\max}^{j} - x_{\min}^{j})$;
5:      end
6:  end
7:  while($|f(BestPath)| \ge \varepsilon$)or($r \le R$)do
8:      for i=1 to P do
9:          //Mutation and Crossover operation
10:         for j=1 to D do
11:             $v_{i,r}^{j} = Mutation(x_{i,r}^{j})$;
12:             $u_{i,r}^{j} = Crossover(x_{i,r}^{j}, v_{i,r}^{j})$;
13:         end
14:         //Selection operation
15:         if $f(u_{i,r}) < f(x_{i,r})$ then
16:             $x_{i,r} \leftarrow u_{i,r}$;
17:             if $f(x_{i,r}) < f(BestPath)$ then
18:                 BestPath $\leftarrow x_{i,r}$;
19:             end
20:         else
21:             $x_{i,r} \leftarrow x_{i,r}$;
22:         end
23:     end
24:     r←r+1;
25: end
26: return the Best solution path BestPath
```

2.2 Variable Neighborhood Search Algorithm Design

Variable Neighborhood Search algorithm is an improved local search algorithm, taking into account that the local optimal solution generated by using a certain neighborhood search is not necessarily the optimal solution of another neighborhood search, variable neighborhood search contains different local search algorithms, starting from the existing generated solution, using different local search algorithms to change the structure of the current solution, when the generated solution in a certain local search algorithm is better than the current solution, then immediately update the current optimal solution, and return to the first local search algorithm to start solving; conversely, enter the next local search algorithm. The algorithm loops until reaching the set upper limit of the number of iterations. The Variable Neighborhood Search algorithm is a

powerful local search algorithm that changes the structure of the solution through different local search algorithms to ensure the diversity of the structure of the solution, thus greatly increasing the probability of finding the optimal solution.
The performance of a Variable Neighborhood Search algorithm is largely determined by the choice of its local search algorithm. *n*-opt search algorithm is a typical local optimal search algorithm, its algorithmic idea is to change the structure of the solution by modifying the order of *n* edges to form a new solution, typical *n*-opt algorithms such as the 2-opt algorithm and the 3-opt algorithm are very effective approximate local search algorithms, which can effectively solving path optimization problems such as the traveler problem or the vehicle path problem; double bridge exchange is a classic local search algorithm in the Lin-Kernighan search algorithm[6], which modifies the structure of the solution by changing the connection of the four edges to produce a new solution, and is considered to be one of the most effective local search algorithms for solving the Travelling Salesman problem. The VNS search algorithm designed in this paper is shown in Fig. 1.

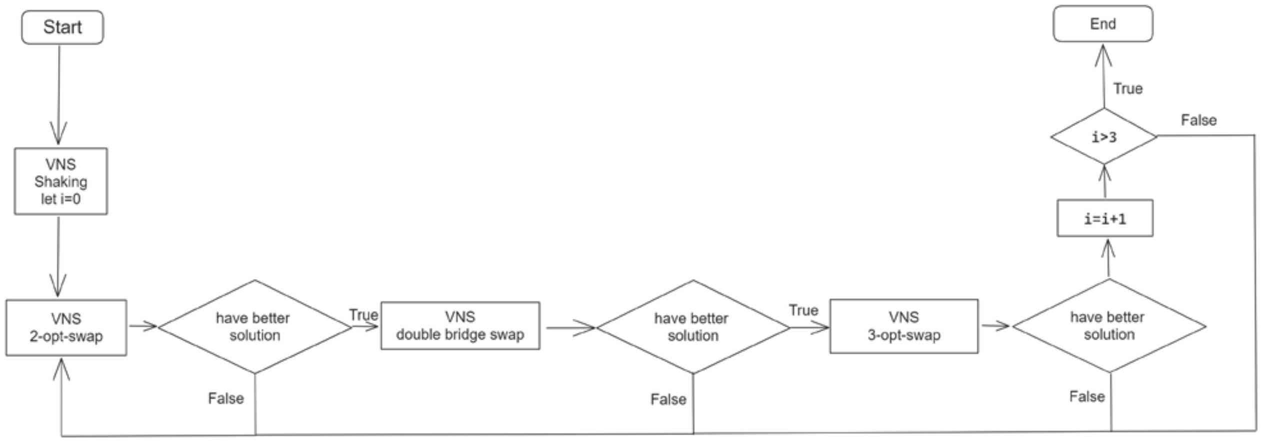

Fig. 1 Flowchart of VNS search algorithm

For the multiple depot MTSP problem, this paper utilizes the 2-opt, double bridge exchange and 3-opt algorithms which can effectively solve the TSP problem, as part of the Variable Neighborhood Search algorithm, and makes use of its powerful local search capability to compensate for the weakness of the DE algorithm which is prone to local optimum[7], so that it is easy to jump out of the local optimum and ensures the diversity of the solution structure in the population in order to generate better solution more quickly.

3. VNS&DE algorithm

3.1 Overview of the VNS&DE algorithm

VNS&DE algorithm is a combination algorithm of differential evolution algorithm and Variable Neighborhood Search algorithm, he has the powerful global search performance of differential evolution algorithm, at the same time, through the variable domain search algorithm to make up for the shortcomings of DE algorithm's weak local search ability.VNS&DE algorithm has powerful search ability no matter in the global search or local search, which includes the operations of selecting, crossover, mutation, 2-opt, double bridge exchange, 3-opt and other operations.
First, initialize the population and randomly generate *N* paths that satisfy the requirements,then perform greedy initialization on these *N* paths, and each time when choosing the next target city to visit, we always choose the unselected closest city to the current city as the next city to visit, and in this way we finally generate the *N* path sets. After that, we set the crossover and mutation probability of DE algorithm, and start to calculate the iteration number of VNS&DE algorithm, perform the regular DE mutation, crossover and selection operations, update the current population solution and then select a city path solution set from the current population in the way of "roulette" algorithm, and enter into the Variable Neighborhood Search algorithm, perform the Shaking algorithm to divide the current optimal solution into four parts, and re-randomize the ordering to form a new solution ,after that we start counting the number of iterations of the VNS algorithm, after which the 2-opt algorithm is executed to change the structure of the solution to form a new solution, and if the new solution is less adaptive than the current optimal solution within the number of iterations stipulated by the algorithm, then it enters into the double bridge exchange algorithm. If the solution generated within the number of exchanges set by the double bridge

exchange algorithm cannot replace the current optimal solution, then enter the 3-opt algorithm, if it still cannot replace the optimal solution then the number of VNS iterations is increased by 1 and re-enter the 2-opt algorithm to solve the problem. If one of the solutions generated during the execution of the three local search algorithms is better than the current optimal solution, the current optimal solution is immediately updated and the search is restarted by jumping to the 2-opt algorithm. The algorithm loops until the number of iterations of the VNS algorithm reaches the set upper limit, and then enter the DE algorithm to update the population, until it reaches the set upper limit of iterations of the VNS&DE algorithm, the VNS & DE algorithm ends.

3.2 VNS&DE Algorithm Flow

Step 1:Initialize VNS&DE parameters: set DE mutation probability 0.05, crossover probability 0.7, upper limit of DE algorithm iteration number is 500, upper limit of Variable Neighborhood Search algorithm iteration number is 3, Population size *Pop_Size*=600, make the current VNS&DE iteration number g=0, and initialize the greedy initialization to generate the initial population *P*.
Step 2:Randomly select three solution sets from the current population, perform DE algorithm mutation operation to generate new solution paths, and calculate the fitness of the new solutions.
Step 3:Select solution paths from the current population and perform DE algorithm crossover operation with mutated individuals to generate new city solution paths and calculate the fitness of new solutions.
Step 4:Perform the DE algorithm selection operation, update the current population, and select an urban solution set by "roulette" algorithm.
Step 5:Enter the Variable Neighborhood Search algorithm, execute the shaking function operation, make the current Variable Neighborhood Search iteration number i=0.
Step 6:Perform 2-opt algorithm operation, randomly select two cities each time, exchange their order in the solution set, generate new solutions and calculate the fitness of the new solutions, if a better solution is generated within the number of operations specified by the 2-opt algorithm, repeat this step, otherwise go to Step 7.
Step 7:Perform double bridge exchange algorithm operation, randomly select four edges each time and exchange their order in the solution set, generate new solutions and calculate the new solution fitness, if a better solution is generated within the specified number of operations of the double bridge exchange algorithm,go to Step 6, otherwise go to Step 8.
Step 8:Perform 3-opt algorithm operation, randomly select three cities each time and exchange their order in the solution set, generate new solutions, calculate the new solution fitness, if a better solution is generated within the specified number of operations of the 3-opt algorithm, repeat this step, otherwise go to Step 9.
Step 9:Let $i=i+1$, if $i>3$ then it reaches the upper limit of iteration number of Variable Neighborhood Search algorithm, go to Step 10, otherwise go to Step 6 to continue searching.
Step 10:make $g=g+1$,if $g<500$ then go to Step 2,Otherwise it represents that VNS&DE algorithm reach the upper limit of iteration number, end VNS&DE algorithm.

4. Computational Results

The VNS&DE algorithm proposed in this paper is implemented using Matlab software, in order to test the performance of the VNS&DE algorithm in solving the multiple depot MTSP problem and to compare the difference in performance between the ordinary DE algorithm and the VNS&DE algorithm, in this paper, we use the TSPLIB test datasets, and we choose three city datasets that have large differences in the number of cities, for each test set we set the number of salesman to 4, 8, 12, and we set a fixed starting point for each of them, and after that, we start testing the algorithm.
In order to set the optimal VNS&DE algorithm parameters for the multiple depot MTSP problem, we utilized the Bayg29 data set, which has fewer cities and is relatively easy to compute the optimal solution for parameter testing, and finally determined that the optimal crossover probability parameter of the DE algorithm is 0.7, the variance probability is 0.05, the population size is 600, and the upper limit of the number of iterations of the VNS&DE algorithm is 500.

Table 1: Comparison of DE algorithm, GA algorithm and VNS&DE algorithm in terms of operational performance

Number of cities	Number of Salesman	DE		GA		VNS&DE	
		Best	Average	Best	Average	Best	Average
29	4	3804	4608	3749	4801	2962	3017
	8	5349	6119	4982	5182	4593	4609
	12	6832	7082	6437	6924	5429	5498
100	4	25637	28479	25349	27638	23457	24078
	8	28039	30018	27043	30154	25049	25333
	12	28447	31109	27471	30879	26396	26608
150	4	28736	30982	30103	32274	28035	28107
	8	29880	33098	29795	34087	28576	28691
	12	32147	34192	31076	35099	29302	29789

Table 1 gives the optimal paths generated by DE algorithm, GA algorithm and VNS&DE algorithm under the test of the datasets with city size of 29,100,150. The data show that under the same parameter settings, VNS&DE can not only find a better path to the city solution set, but also the solution differences are more stable than the ordinary DE algorithm and GA algorithm, which proves that VNS&DE can jump out of the local optimum timely when the algorithm falls into the local optimum, ensure the diversity of the population solution, and find the optimal solution with a greater probability.

5. conclusion

In this paper, for the shortcomings of DE algorithm which is easy to fall into local optimization, Variable Neighborhood Search algorithm is introduced to make up for the shortcomings of ordinary DE algorithm which is easy to fall into local optimization. For the multiple depot MTSP problem, the solution set of the VNS&DE algorithm proposed in this paper is proved to be better than the ordinary DE algorithm, and the probability of finding the optimal solution is higher, which proves the feasibility and effectiveness of the VNS&DE algorithm.However, the VNS&DE is a kind of algorithm that sacrifices the time for the accuracy of the solution, so when the size of the required set of cities is larger, the time complexity of the VNS&DE algorithm needs to be further improved.

Acknowledgments

This work it was supported by National College Students' innovation and entrepreneurship training program project(2024)

References

[1] Yanan Cheng , Xiaofeng Wang, Jingzo Liu et al. K-means clustering information propagation algorithm for solving the multi-start-multi-traveler problem[J]. Science, Technology and Engineering,2022,22(23):10146-10154.
[2] Yun Bai,Yuyuan Gao.Application of differential evolutionary algorithm to the traveler problem[J]. Science and Technology Innovation,2022,(23):23-26.
[3] Huiren ZHOU,Wansheng TANG,Hailong WANG. Optimization of multi-traveler problem based on differential evolutionary algorithm[J]. Systems Engineering Theory and Practice,2010,30(08):1471-1476.
[4] Banu Soylu,A general variable neighborhood search heuristic for multiple traveling salesmen problem,Computers & Industrial Engineering,Volume 90,2015,Pages 390-401.
[5] Sofge, D., Schultz, A., De Jong, K. (2002). Evolutionary Computational Approaches to Solving the Multiple Traveling Salesman Problem Using a Neighborhood Attractor Schema. In: Cagnoni, S., Gottlieb, J., Hart, E., Middendorf, M., Raidl, G.R. (eds) Applications of Evolutionary Computing. EvoWorkshops 2002. Lecture Notes in Computer Science, vol 2279. Springer, Berlin, Heidelberg.
[6] Keld Helsgaun "An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic" European Journal of Operational Research.126(1),106-130(2000)
[7] Xingjun Lai, Xin Tang, Xin Lin et al.Multi-UAV cooperative area search strategy based on differential evolutionary particle swarm hybrid-algorithm[J/OL]Journal of Ballistic and Guidance:1-11[2024-02-13].

Jinhua Bian is currently reading in School of Computer Science and Software Engineering, University of Science and Technology LiaoNing, Anshan 114051, China. He is an undergraduate in the third year.
Xiaoxia Zhang is currently working as a Professor at the School of Computer Science and Software Engineering, University of Science and Technology LiaoNing, Anshan 114051, China.