An Improved Differential Evolution Algorithm for TSP Problem

Mei Mi, Xue Huifeng, Zhong Ming, Gu Yu
College of Automation , Northwestern Polytechnical University , Xian, Shaanxi, 710021, China
Littlemeimi@163.com

***Abstract*—TSP (Traveling Salesman Problem) is a kind of typical NP problems, mostly settled by genetic algorithm (GA). Differential Evolution Algorithm (DE) is a kind of new Evolution Algorithm which has many similarities with GA. We proposed to solve TSP problem by improved differential evolution algorithm. Added an auxiliary operator for regulating integer sequence to mutation process, and replaced the original crossover operator by Liuhai crossover operator. Experimental results show that this method can effectively improve the convergence speed and optimal quality, show good characteristic in the solution of TSP problem.**

Keywords-differential ecolution algorithm; tsp problem; genetic algorithm

I. INTRODUCTION

Traveling Salesman Problem (TSP) is a typical combinatorial optimization problem in which a saleman must seek a shortest tour to visit a list of cities and each city could be visited only once. This is a np-hard problem which is easy-to-describe but difficult to solve, its mathematical description as follows.

$$\min T = \sum_{i=1}^{n-1} d(c_i, c_{i+1})$$

Where: c_i is the i-th city, i =1, 2, 3⋯n; T is a total distance of a journey; $d(c_i, c_j)$ is the distance between city i and city j [1].

If the number of cities is n, there exists (n-1)! Closed paths. It is difficult to get the optimal solution precisely in a short time. So it makes sense to study approximate algorithm and find the optimal solution. Many researches on TSP have been done at home and abroad, common algorithms include genetic algorithm, simulated annealing algorithm, ant colony algorithm, neural networks and so on.

Differential evolution algorithm (DE) is a heuristic algorithm for global optimization proposed by Dainer Storn and Kenneth Price in 1995. In the first International IEEE Evolutionary Optimization Competition in 1996, DE was considered as the fastest evolutionary algorithm through on-site verification at present. For the simple control featuring less parameters and powerful capability of global optimization as well as other characteristics, it was rapidly applied in chemical industry, power electronics, robotics, bioinformatics and other realms. As a kind of evolutionary algorithms, DE has became a growing concern in discrete space optimization problem and been gradually used in multi-objective vehicle scheduling in recent years. However, with the differential evolution algorithm for path optimization problems is still relatively small [2].

II. TRADITIONAL DIFFERENTIAL EVOLUTION

The tradition DE based on real-code is similar to genetic algorithm in theory, and both include three main operations: crossover, mutation and selection factors. But the orders of the process are different. Compare with other evolution algorithms, DE produces a mutant vector in the mutation operation by adding the weighted difference between two randomly chosen vectors to a third vector [3]. Its pseudo-code form is given in Fig.1.

For the standard DE, the algorithm scheme is DE/rand/1/bin [4]. Where popsize is the number of populations, $x_{i,g}$ is the i-th individual in g-th generation.

Create an initial population
While (termination criterion is not satisfied)
Do Begin
For each individual $x_{i,g}$ in the population
Begin
Randomly generate three integer numbers
$r_1, r_2, r_3 \in [1 : popsize]$, where $r_1 \neq r_2 \neq r_3$
$v_{i,g+1} = x_{r1,g} + F \cdot (x_{r2,g} - x_{r3,g})$
Randomly generate one real number $rand_j \in [0;\ 1)$
If $rand_j < CR$ then $u_{i,g+1} = v_{i,g+1}$
Else $u_{i,g+1} = x_{i,g}$;
If $fitness(u_{i,g+1}) < fitness(x_{i,g})$
then $x_{i,g+1} = u_{i,g+1}$
Else $x_{i,g+1} = x_{i,g}$;
End;
End.;

Figure 1. Pseudo-code for a DE

The F parameter is scaling the values added to the particular decision vectors, and the CR parameter represents

978-0-7695-4077-1/10 $26.00 © 2010 IEEE
DOI 10.1109/ICICTA.2010.461

the crossover rate. The parameters F∈ [0;2]，and CR∈ [0:1] are determined by the user, and $v_{i,g+1}$ is the vector after mutation., $u_{i,g+1}$ is the vector after crossover, and fitness is the objective function. If the $fitness(u_{i,g+1})$ is better than the $fitness(x_{i,g})$, then put $x_{i,g}$ into next generation. The search space must be continuously.

III. IMPROVED DIFFERENTIAL EVOLUTION FOR TSP PROBLEM

DE is based on the real valued operators, so it does not suit combination optimization problems. To solve TSP, this paper introduced an improved DE which derived from traditional DE and genetic.

A. *Coding and fitness function*

The presented algorithm adopts integer-code which is simple and intuitive. And the fitness function f is the reciprocal of the length of the total path T, f=1/T. The larger the f is, the shorter the total path is and the corresponding individual is closer to the optimal solution [5].

B. *Create an initial population by greedy algorithms*

As the path of TSP problem is a circular loop, so any city can be selected as the starting one. In this paper, the initial population is created by greedy algorithm. To do as follows: randomly select a starting city C, according to the distance matrix, choose the second city D which is the nearest one from C, and then select the nearest city from D as the next city and so on. Always choose the nearest city from the current city as the next one, connect the last city to the starting one until all the cities were traveled [6].

C. *Mutation*

The mutation of DE algorithm is different from other evolutionary algorithms. After the amplication of individual vectors by the mutation parameter F, the obtained value is real number which must be changed into integer in order to get the integral fitness function. To solve this problem, a method based on regulating integer sequence can be adopted. The method is described as follows: put the real genes of the individual in ascending order according the value, and encode them. For example, put the smallest individual first, numbered 1. Then change the value of the individual into its corresponding coded number [2]. As follows:

The individual after mutation:

[3.4, 0, 0.5, 6.3, 2.1, 5.0];

Rank the genes in ascending order:

[0, 0.5, 2.1, 3.4, 5.0, 6.3];

Numbered: 1, 2, 3, 4, 5, 6;

So, the obtained individual: [4, 1, 2, 6, 3, 5].

D. *Liuhai Crossover*

The traditional crossover, such as order crossover, cycle crossover and other method, can not keep well the good gene of parents. To cover this shortcoming, the paper adopts Liuhai crossover [7]. It includes the best individual preservation strategy which puts the best individual into the next generation with probability 1 and cross the remaining individuals. The main idea of Liuhai crossover as follows. Suppose there are n cities, numbered 1-n, and the parents for crossing are:

$$x_1 = (r_{11}, r_{12}, r_{13}, \cdots r_{1n}),\ x_2 = (r_{21}, r_{22}, r_{23}, \cdots r_{2n})$$

The main purpose of the cross is to inherit excellent genes from parent. Take the above chromosome as a loop, that is, the next city of r_{1n} is $r_{1n+1} = r_{11}$.

- Choose the $r_{1c} = r_{11}$ as the present city, find $r_{2k} = r_{1c}$ from chromosome x_2, then add r_{1c} in offspring.
- If both r_{1c+1} and r_{2k+1} were not in offspring, then compare $d_{r_{1c}r_{1c+1}}$ with $d_{r_{2k}r_{2k+1}}$, if $d_{r_{1c}r_{1c+1}} \geqslant d_{r_{2k}r_{2k+1}}$, then keep $r_{1c} = r_{2k+1}$ as the present city, else keep $r_{1c} = r_{1c+1}$ as the present city.
- Find $r_{2k} = r_{1c}$ from chromosome x_2, then add r_{1c} in offspring.
- If r_{1c+1} was already in offspring while r_{2k+1} was not, then keep $r_{1c} = r_{2k+1}$ as the present city and add it in offspring; If r_{2k+1} was already in offspring while r_{1c+1} was not, then keep $r_{1c} = r_{1c+1}$ as the present city; If neither r_{1c+1} nor r_{1k+1} is in offspring, then compare $d_{r_{1c}r_{1c+2}}$ with $d_{r_{2k}r_{2k+2}}$.
- Repeat step 2 until the offspring is complete.

Choose r_{11} and r_{21} as the current starting city separately in the first step, then two offspring can be obtained by Liuhai Crossover.

IV. NUMERICAL EXPERIMENTS

To verify this new version of DE, symmetric and asymmetric TSP problems are tested by MATLAB2009.

A. *Symmetric TSP problem*

The proposed approach with popsize=20, Pc=0.8 and 100 generations have been implemented to TSP cases in [6] with 10 cities. The distance matrix is shown in Table I.

TABLE I. DISTANCE MATRIX OF SYMMETRIC TSP PROBLEM

Dist	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***
1	0	82	34	127	118	46	110	127	52	44
2	82	0	76	83	99	89	106	83	84	33
3	34	76	0	127	74	44	56	106	94	86
4	127	83	127	0	37	51	118	106	124	99
5	118	99	74	37	0	91	37	132	49	37
6	46	89	44	51	91	0	112	131	66	85
7	110	106	56	118	37	112	0	64	121	71
8	127	83	106	106	132	131	64	0	43	107
9	52	84	94	124	49	66	121	43	0	89
10	44	33	86	99	37	85	71	107	89	0

The numbers 1-10 represent cities

The optimal value 471 can be obtained just in the seventh generation, in the best cases, whose optimal path is （4 6 3 1 10 2 9 8 7 5）. The results show that the proposed approach is better and more effectiveness than the method in [6].

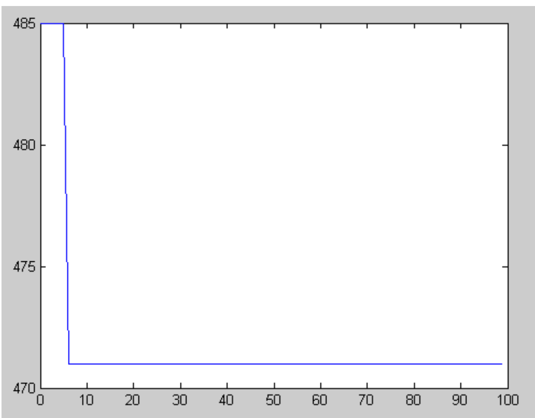

Figure 2. The sequence diagram of symmetric TSP cases.

B. *Other experiments*

In order to prove the performance of the method further, a number of TSP problem instances have been selected with which to test the presented approach. These problems are from TSPLIB [9] and the results are outlined in Table II.

TABLE II. EXPERIMENT RESULTS COMPARED WITH THE BEST-KNOWN VALUE

Name	**Cities**	**Obtained best value**	**Best-known value**
Oliver30	30	424	424
eil51	51	429	426
eil76	76	545	538
KroA150	150	26744	26524

Compare with TSPLIB

In the widely accepted eil51 problem, set popsize=300, and the generations 50. By experiment, the best length obtained is 429.7271 while the optimal length of TSPLIB is 426. The chart of optimum path is as follows in Fig.3.

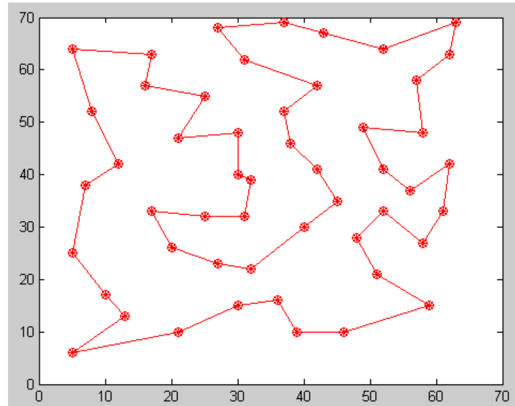

Figure 3. The optimum path of eil51.

C. *Asymmetric TSP problem*

To validate the proposed method is effective not only in symmetric TSP problem but also in asymmetric problems, choose the symmetric case in [7] with 8 cities to test. Set the parameters according [7], popsize=10, Pc=0.2. The optimal value 52 can be got in the eighth generation, the results show that the proposed approach is better and more effectiveness than the method in [7].

V. CONCLUSION

This paper introduced a novel differential evolution to solve traveling salesman problems, which compensated the defect of traditional method by generating initial population by greedy algorithm, changed mutated real vector into integer based on regulating integer sequence, and inherited excellent genes from parents through Laihai Crossover. The experiments indicated that the proposed method was faster in convergence in TSP problems and can get optimal solution or approximate optimal solution. The method was proved to be effective and feasibility.

ACKNOWLEDGMENT

The authors are grateful to the anonymous referees for their valuable comments, which have improved the presentation of this contribution.

REFERENCES.

[1] Ji Huaimeng, Cai Haibin, “A Novel Genetic Algorithm for Traveling Salesman Problem,” Computer & Digital Engineering, Vol.37, No.4, pp. 10-12, 2004.

[2] Cao Erbao, Lai Mingyong, “Research on VRP-SDPTW Based on Improved Differential Evolution,” journal of industrial engineering and engineering management, Vol.23, No.2, pp. 80-84, 2009, doi: 1004-6062 (2009) 02-0080-06

[3] Zhou Yanping, Gu Xingsheng, “Development of Differential Evolution Algorithm,” Control and Instruments in Chemical Industry, Vol.34, No.3, pp. 1-5, 2007, doi:1000-3932 (2007) 03-0001-05.

[4] Adam Slowik, Michal Bialko, “Training of artificial neural networks using differential evolution algorithm,” Krakow, Poland, pp. 25-27, May. 2008, doi:1-4244-1543-8/08/$25.00 ©2008 IEEE.

[5] Huang Yongjun, Wu Youxin and Liu Huabin, “Based on improved genetic algorithm for TSP,” Computer Engineering and Design, vol. 128, Aug. 2007, pp. 5909-5911, Dec.2007, doi:1000-7024(2007)24-5909-03

[6] Hu Xiaohui, Li Xiaoyang, and Chen Junlian, “Hybrid Genetic Algorithm Based on Strategy of Greedy for TSP,” journal of Lanzhou Jiaotong University, Vol.28, No.3, pp. 58-61, Jun. 2009, doi:1001-4373 (2009) 03-0058-04

[7] Liu Hai, Hao Zhifeng, and Lin Zhiyong, “Improving Genetic Cross Operator to Solve TSP Problem,” Journal of South China University of Technology, pp. 71-73, Dec.2002, doi:1000-565X(2002) 12-0071-03

[8] H. K. Tsai, J. M. Yang, Y. F. Tsai, and C. Y. Kao, “An Evolutionary Algorithm for Large Traveling Salesman Problems,” IEEE Transactions on SMC -Part B, vol. 34, Aug. 2004, pp. 1718-1729, doi:1083-4419/04$20.00©2004IEEE.

[9] TSPLIB, Http://www.Iwr.uni_heidelberg.de