Optimized Crossover Genetic Algorithm for TSP based on Path Representation

Jianzhong Xu, Fu Yan [*] Fengshu Li
(School of Economics and Management, Harbin Engineering University, Harbin 15001, China)

Abstract

The genetic algorithm is a population based meta-heuristic global optimization technique for dealing with complex problem with very large search space. In order to improve its performance, the optimized crossover genetic algorithm based on path representation is proposed in this paper. Firstly, greedy choice method of improving the generation of initial population is applied to improve the quality of initial population. Secondly, an improved crossover operation is raised to improve the convergence speed. Finally, an improved evolutionary strategies of genetic algorithm is given, which preserves elite individuals produced by crossover. The results of eight simulation experiments show that, compared with basic genetic algorithm, the optimized crossover genetic algorithm is more effective, which significantly improves computing speed and reduces iteration number.

Keywords: Path representation; Greedy choice method; Optimized crossover; Evolutionary Strategy

1. Introduction

Traveling salesman problem (TSP) is a typical NP-hard problem in the field of modern combinatorial optimization, it can be described briefly as follows: *m* cities and the distances between each them are given, the traveling salesman problem (TSP) is to find a complete, minimal-cost tour path visiting each city once[1].The goal of TSP is to find a access sequences, which means to find a circuit path $C=(c_1, c_2, \ldots, c_m)$, to minimize the objective function.

$$min\ f\left(d\right)=\sum_{i=2}^{m-1}\sum_{j=2}^{m-1}d(\mathrm{c}_i,c_j)+d(c_m,c_1) \tag{1}$$

where c_i is represent the city number *i*, $d(c_i,c_j)$ is the distance between city *i* and city *j*.

Although the statement of the problem is very simple, it is a well-known NP-hard problem and unable to find a polynomial time complexity when solving complex problems[2]. For decades, academia put forward many algorithms to solve TSP problem, these algorithms are primarily classified into two categories. The first category is the traditional optimization algorithm, such as exhaustive method[3], Linear Programming Approach[4], Branch and Bound Algorithm[5], etc. It can obtain the optimal loop by searching the whole solution space when the numbers of the city is small, but when the numbers of the city is large, it can't solve effectively. The second category is approximation algorithms, such as Insertion Algorithm[6], Nearest Neighbor[7], λ-opt Algorithm[8], Neural Networks[9], Genetic Algorithm[10] etc. The second method can't search the whole solution space, but it can find the approximate solution for the problem and the operation time is much less than the first method.

*This paper is funded by the International Exchange Program of Harbin Engineering University for Innovation—originated Talents Cultivation. We also acknowledge anonymous IAMOT reviewers whose constructive feedback helped to improve the quality of this paper.

The Genetic algorithm (GA) derived from the principle of natural selection and the concept of the GA is a global search method, and it performance to solve complex problems is effective and parallel. Although GA has many advantages of global search, it easy premature and convergence to the local optimum, and many scholars provide lots of improved methods to overcome its drawbacks. Yong Deng presented an improved genetic algorithm which based on k-means algorithm in initialization of population to increase the diversity of population [11]. Ao Haldun Sürral, et al. raised an improved algorithm to prevent premature and maintain population diversity by employing exchange heuristic crossover and exchange mutation operators[12]. Shubhra Sankar Ray gave a new "nearest fragment operator" and a modified version of order crossover operators of GA for solving TSP[13]. However, those improve theory, although remedies the problem of easy convergence to the local optimum and the diversity of population based algorithms to some extent, it is far from being an ideal solution towards addressing it completely. Hence, an improvement over GA theory, named optimized crossover genetic algorithm is introduced in this paper.

2. The proposed of an improvement Genetic Algorithm

2.1. The method of producing initial population

Haupt points out that for the intelligent optimization algorithm based on group iteration, the initial population can affect the accuracy and convergence speed of the algorithm[14]. The initial population with better diversity can lay the foundation for GA algorithm. However, the initial population generated by the random method can't guarantee its diversity, and couldn't extract the useful information in the search space effectively, and maybe affect the search efficiency of the algorithm to a certain extent. In this paper, an improved method based greedy choice is used to produce the initial population. Hypothesis, there are *m* cities and their pair wise distance, the method of producing initial population can be described as follows.

(1) Generating a city randomly and denoting it as "a", which is regarded as the starting city of the route and marked it. (2) Selecting a city which is the nearest to the city "a" among all unmarked cities and denoting it as "b", the route between the city "a" and the city "b" is described as "a-b", the city "b" is then marked. (3) Following the rule that produce the route "a-b", selecting a city which is nearest to the city "b" among all unmarked cities and denoting it as "c", the route among the city "a", "b" and "c" are described as "a-b-c" and the city "c" is then marked. (4) Repeating (2) and (3) until all cities have been selected and then get a route, those selected "city" are the individuals of population.

2.2. The method of optimized crossover operation

The thought of improvement genetic algorithm can be described as follows. First of all, we use classic roulette wheel method to select individuals from the population as parent individuals, denote them as $(c_{i1}, c_{i2}, \cdots, c_{im})$, $c_{ij}\left(j=1,2,\cdots,m\right)$ represent the *j*th city connect with the *i*th path.

Second, one of the cities is chosen randomly as intersection point. For the city "c_i", according to the path distance between the city "c_{i-1}" and the city "c_{i+1}", we can compare the distances of $d\left(c_i, c_{i-1}\right)$ and $d\left(c_i, c_{i+1}\right)$, if $d\left(c_i, c_{i-1}\right) \geq d\left(c_i, c_{i+1}\right)$, then visiting c_{i+1}; otherwise visiting c_{i-1}.

The selected cities must be deleted from the parents. Finally, repeating the above step until all cities had been selected once.

As an improvement genetic algorithm adopts the principle of minimum distance, makes each crossover operation directional, so it is an optimization process in each crossover operation. As the next joint city is selected from two directions, so it increasing the possibility of offspring's inheriting excellent gene from the parents', thus it is beneficial to improving the convergence speed and finding out optimum solution.

In order to introduce the optimized crossover genetic algorithm based on path representation clearly, the nine cities are introduced into this paper as an example to introduce the method. The distances of the nine cities are listed in table 1.

Table 1 The distance between nine cities

$d(c_i, c_j)$	1	2	3	4	5	6	7	8	9
1	0	18	13	7	21	4	15	29	17
2	18	0	9	25	16	23	5	20	11
3	13	9	0	22	38	29	34	19	38
4	7	25	22	0	21	69	31	28	31
5	21	16	38	21	0	41	39	16	37
6	4	23	29	69	41	0	12	27	24
7	15	5	34	31	39	12	0	51	23
8	29	20	19	28	16	27	51	0	54
9	17	11	38	31	37	24	23	54	0

1) Two individuals are selected from the population as parent through classic roulette wheel method, and denoted them as *A* and *B*.

A=1 2 3 4 5 6 7 8 9；B=3 4 6 8 1 2 9 7 5

The fitness values of *A* and *B* are: *f*(*A*)=1/245；*f*(*B*)= 1/276.

2) One intersection point is determined randomly and regarded it as the starting city of offspring individual. Hypothesis, the random city is "*4*", and the offspring individual *AA* can be described as follows.

*AA= 4 * * * * * * * *,*

The city "*4*" is deleted from parent and the location of deleting city is represented by "*", so the parent can be described as follows.

A_1 *=1 2 3 * 5 6 7 8 9;* B_1 *=3* 6 8 1 2 9 7 5;*

3) According to the table 1, the distance of the city "3" connect with the city "4" and the distance of the city "5" connect with the city "4" are compared, and selects the minimum one as the next visit city. If there are equal, then random select either city as the next city. Similarly, the visited city is deleted from parent.

Base on the table 1, we know that the nearer to the city "4" is the city "5", so regard it as the next visited city. The offspring individual *AA* can be described as follows.

*AA= 4 5 * * * * * * *;*

4）Repeating the step 3 until all cities have been selected. Through this method, we can get the offspring individual *AA*.

AA= 4 5 3 2 9 1 8 6 7;

Then the second offspring individual can be produced through repeating the above steps, hypothesis, the random city is "*8*", we can get the second offspring individual *BB*, *BB= 8 1 9 2 3 4 5 7 6*. And the fitness values of the two offspring individuals are *f(AA)=1/195, f(BB)=1/178*, the value of the offspring individual is far exceeds its' parent's. Therefore, the offspring individuals inherit better genes from their parent. This method is introduced into this paper to improve the quality of offspring individuals and increase the convergence speed of GA algorithm.

2.3 The implement of an improvement evolutionary strategy

The improved evolutionary strategy of optimized crossover genetic algorithm can be described as follows. Firstly, using the method which mentioned in section 2.1 to generate the initial population that containing *n* individuals, and regard them as the parent. The fitness values of parent are calculate and sorted in descending order, and select the *q* elitist individuals with high order to preserve. Secondly, selection operator and crossover operator are adopted to generate the *n* new individuals. Calculating the fitness values of those new individuals and sorting the new individuals in descending order, the *q* elitist individuals with high order are selected to preserve again. Thirdly, the *q* nonredundant elitist individuals are selected from 2*q* preserved individuals. Fourthly, mutation operator is adopted to generate new individuals and sorting them again. The *n* poor individuals of the sequence are replaced by the *q* nonredundant elitist individuals. The *n* poor individuals of the sequence are replaced by the *q* nonredundant elitist individuals. If the stop condition is met, then terminates the loop and outputs the best route; otherwise regards the offspring as the parent, repeats the steps above until the stop condition is met. The evolutionary strategy frame is plotted in Figure 1.

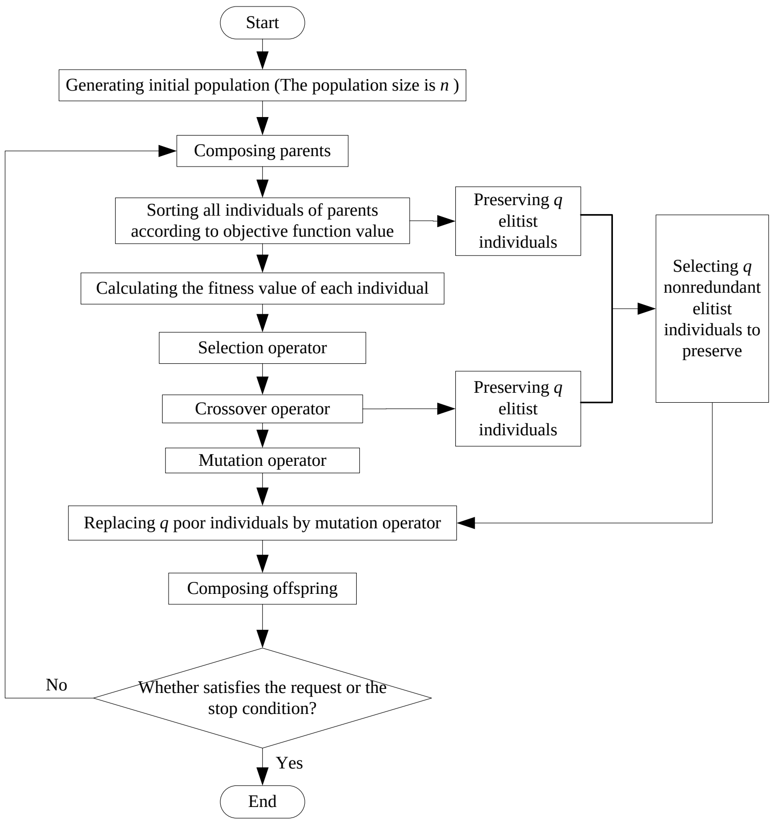
Start
Generating initial population (The population size is *n*)
Composing parents
Sorting all individuals of parents according to objective function value
Preserving *q* elitist individuals
Calculating the fitness value of each individual
Selection operator
Crossover operator
Preserving *q* elitist individuals
Selecting *q* nonredundant elitist individuals to preserve
Mutation operator
Replacing *q* poor individuals by mutation operator
Composing offspring
Whether satisfies the request or the stop condition?
No
Yes
End

Figure1. Block diagram of improved evolutionary strategy

The improved evolutionary strategy has many advantages compared to the traditional Genetic Algorithm. In evolutionary strategy of the traditional Genetic Algorithm, the mutation operation is operated directly after the crossover process is finished, this method leading the excellent individuals generated in crossover process may be damaged in the mutation process.

As a contrast, an improved evolutionary strategy proposed in this paper has been used to preserve the excellent individuals come from the parent, generated in the crossover process and the mutation process. Therefore, the improved evolutionary strategy can keep the nice genes of population reserved and improve the performance of algorithm.

3. Experimental results

3.1. Basic operation and compared conditions

In order to detect the validity of the optimized crossover genetic algorithm, we apply this improvement algorithm to solve some classical examples of TSP and the results are compared with the "basic genetic algorithm" and the crossover operator genetic algorithm[15]. All these operators and parameters of the two algorithms are listed in table 2.

Table2 Basic operators and parameters of the two algorithms

NO.	Basic Operation	OCGA	BGA	Parameters
1	Encoding mode	Path representation	Path representation	---
2	Population initialization	Greedy choice method	Greedy choice method	Population size n=100
3	Fitness function	$f(A)=1/d(A)$	$f(A)=1/d(A)$	---
4	Selecting operation	Roulette wheel method	Roulette wheel method	---
5	Crossover operation	Optimal crossover operation	The method of reference[10]	$P_c=0.9$
6	Mutation operation	Inversion operator	Inversion operator	$P_m=0.5$
7	Elitism selection	Preserve elitist individuals	Preserve elitist individuals	$P_s=0.1$
8	Terminal condition	Reaching maximum iteration	Reaching maximum iteration	gen=1000

Annotation: In table 2, OCGA is optimal crossover genetic algorithm; BGA is the basic genetic algorithm. The inversion operator is presented to select two gene positions randomly and inverse the chain between the two positions in a chromosome.

3.2. Cases and calculation result analysis

The classical examples of TSP are wxp20 take from the reference [16, 17], which includes twenty cities, fourteen cities, fifty-one cities, seventy cities, one hundred and five cities, one hundred and twenty-four cities, two hundred and twenty-five cities, four hundred and ninety-three cities. According to the size of cities, we divide them into four groups, and list them in table 3.

Table3 Examples of grouping information table

NO.	Group	Size	Example
1	I	$0<\text{Size}\leq 50$	burma14,wxp20
2	II	$50<\text{Size}\leq 100$	eil51,st70
3	III	$100<\text{Size}\leq 200$	lin105,Pr124
4	IV	$200<\text{Size}\leq 500$	tsp225,d493

In table 3, there is relatively small number of cities in group Ⅰ and Ⅱ,so it easy to solved it for the algorithm. While, in group Ⅲ and Ⅳ, the optimal solution may be obtain very difficult as for the number of cities is too much. While the performance of algorithm can express more obvious when handle complex problems. The programs of the two algorithms has been written by using Matlab and run in the same computer twenty times indepently, and the average values are presented in table 4.

Table 4 The result of comparing two methods

Problem	Optimal solution	Algorithm	Average Optimal Solutions	Average Time/s	Average Iteration/times	Relative Error
bruma14	30.8785	BGA	30.9308	14.9296	456	0.0017
		OCGA	30.8785	7.7658	201	0
wxp20	24.5222	BGA	24.5222	2.0002	42	0
		OCGA	24.5222	1.736	25	0
eil51	426	BGA	431.5894	149.8781	645	0.0131
		OCGA	428.6116	81.6408	254	0.0061
st70	675	BGA	685.7737	626.5242	700	0.0159
		OCGA	683.3754	254.8343	537	0.0124
lin105	14379	BGA	14695.5415	944.0390	648	0.0221
		OCGA	14471.6247	595.6301	527	0.0064
Pr124	59030	BGA	60908.9264	1594.0556	586	0.0318
		OCGA	59860.5672	742.3430	550	0.0140
tsp225	3916	BGA	4223.5438	4629.2761	721	0.0785
		OCGA	3916.0990	2828.3500	693	0.00002
d493	35002	BGA	37294.3445	25826.9904	952	0.0655
		OCGA	36251.1126	15210.5366	696	0.03569

In table 4, the method of computing average iteration is

$$gen' = \sum_{i=1}^{10} gen_i / 20 \tag{2}$$

where, gen_i is the iteration when procedure gets the shortest route firstly. The method of computing average time is

$$t' = \sum_{i=1}^{10} (t \times gen_i / gen) / 20 \tag{3}$$

where, *t* is the time that the procedure runs *1000* times and *gen=1000*.

From table 4, we can conclude that the optimized crossover genetic algorithm is better than the basic genetic algorithm in average time and the average iteration. For the burma14, the run time under the twenty runs reduces by 47.98% and the iterations reduce 55.7% respectively compared with the basic genetic algorithm. For the wxp20, the run time reduces by 13.21% and the iterations reduce by 39.57%. For the eil51, the run time reduces by 45.53% and the iterations reduce by 60.61%. For the st70, the run time reduces by 59.33% and the iterations reduce by 23.29%. For the lin105, the run time reduces by 36.91% and the iterations reduce by 18.67%. For the Pr124, the run time reduces by 53.43% and the iterations reduce by 6.14%. For the tsp225, the run time reduces by 38.90% and the iterations reduce by 3.88%. For the d493, the run time reduces by 41.11% and iterations reduce by 26.89%.

The relative errors have not significant different between the two algorithms, as for the basic genetic algorithm adopts the same operator with optimized crossover genetic except crossover operator. Even though the relative error is reduces by 0.0001, it will make tremendous progress in the veracity of solution when there is a lot of cities. For example, in solving the problem of the d493, the relative error is reduced by 0.02981 compared with the basic genetic algorithm, but the average optimal solution is reduced by 1043.2319, it is tremendous progress.

In conclusion, the test results of the eight test functions show that the optimized crossover genetic algorithm presented in this paper have faster calculation speed and the number of iterations significantly reduced than the basic genetic algorithm, and the validity of optimized crossover genetic algorithm is powerfully illustrated.

4. Conclusions

1) Based on the basic algorithms for TSP, especially the improvement Genetic Algorithm given by Yang huafen, this paper provides optimized crossover genetic algorithm and gives the method of generating offspring.

2) An improvement method of generating initial population is given, which improves the quality of initial population and accelerate the calculation speed.

3) An improved evolutionary strategies of the genetic algorithm is given, which preserves elite individuals produced in crossover. It overcomes the shortage of the traditional genetic algorithm whose excellent individuals produced by crossover may not survive in the process of mutation.

4) The Test results of the eight test functions show that the optimized crossover genetic algorithm has faster calculation speed and the number of iterations significantly reduced than basic the genetic algorithm, thus, the validity of optimized crossover genetic algorithm based on path representation is powerfully illustrated.

5. Acknowledgement

This research was supported by National Natural Science Foundation of China (No. G201119) and Heilongjiang Province Science Foundation for Youths (Grants No. JJ2016QN0645).

6. References

[1] DING Chao. CHENG Ye. HE Miao. Two-Level Genetic Algorithm for Clustered Traveling Salesman Problem with Application in Large-Scale TSPs. Tsinghua Science and Technology, (2007) 12(4): 459-465.

[2] Chiranjit Changdar. G.S. Mahapatra. Rajat Kumar Pal. A modified genetic algorithm-based approach to solve constrained solid TSP with time window using interval valued parameter. International Journal of Operational Research, (2016) 26(4): 398-421.

[3] Michael P. Poland. Christopher D. Nugent. Hui Wang, et al. Genetic algorithm and pure random search for exosensor distribution optimisation. International Journal of Bio-Inspired Computation, (2012) 4(6): 359-372.

[4] Ali R. Yildiz. A comparative study of population-based optimization algorithms for turning operations. Information Sciences, (2012) 210: 81-88.

[5] Ali R. Yildiz. Comparison of evolutionary-based optimization algorithms for structural design optimization. Engineering Applications of Artificial Intelligence, (2013) 26(1): 327-333.

[6] Qing-Hu Hou. An insertion algorithm and leaders of rooted trees. European Journal of Combinatorics, (2016) 53: 35-44.

[7] Ceyhan, Elvan. Bahadır, Selim. Nearest neighbor methods for testing reflexivity. Environmental & Ecological Statistics, (2017): 69-108.

[8] Hirotaka Niitsuma. Shin Ishii. Minoru Ito. Analog λ-opt approach to quadratic assignment problem. Systems and Computers in Japan, (2000) 31(10): 1-9.

[9] H. N. Mhaskar. Neural networks and approximation theory. Neural Networks, (1996) 9(4): 721-722.

[10] Stephanie Forrest. Genetic algorithms. ACM Computing Surveys, (1996) 28(1): 77-80.

[11] Yong Deng. Yang Liu. Deyun Zhou. An Improved Genetic Algorithm with Initial Population Strategy for Symmetric TSP. Mathematical Problems in Engineering, (2015) 2015: 1-6.

[12] Haldun Süral. Nur Evin Özdemirel. Ýlter Önder. et al. An Evolutionary Approach for the TSP and the TSP with Backhauls. Computational Intelligence in Expensive Optimization Problems, (2010) 2: 371-396.

[13] Shubhra Sankar Ray. Sanghamitra Bandyopadhyay. Sankar K. Pal. Genetic operators for combinatorial optimization in TSP and microarray gene ordering. Applied Intelligence, (2007) 26(3): 183-195.

[14] Haupt R, Haupt S. Practical genetic algorithm. New York: John Wiley and Sons, 2004.

[15] Yang Huafen. Improved Genetic Algorithm for TSP. Journal of Chongqing Institute of Technology (Natural Science Edition), (2007) 21: 87-90.

[16] Wang Xiaoping. Cao Liming. Genetic Algorithm—theory, application and software implementation. Xi an: Xi'an Jiaotong University Press, 2002.

[17] Reinelt, Gerhard. TSPLIB-- A Traveling Salesman Problem Library. ORSA Journal on Computing, (1991) 4(6): 376-376.