2011 Fourth International Joint Conference on Computational Sciences and Optimization

A New Design of Genetic Algorithm for Solving TSP

Yingying Yu, Yan Chen, Taoying Li
Transportation Management College
Dalian Maritime University
Dalian, China
uee870927@126.com

Abstract— **In this paper, we develop an algorithm that is able to quickly obtain an optimal solution to TSP from a huge search space. This algorithm is based upon the use of Genetic Algorithm techniques. The algorithm employs a roulette wheel based selection mechanism, the use of a survival-of-the-fittest strategy, a heuristic crossover operator, and an inversion operator. To illustrate it more clearly, a program based on this algorithm has been implemented, which presents the changing process of the route iteration in a more intuitive way. Finally, we apply it into a TSP problem with fifty cities. By comparing with other published techniques, we can easily know that the proposed algorithm can efficiently complete the search process and derive a better solution.**

Keywords- TSP; GA; selection operator; crossover operator; mutation operator

I. INTRODUCTION

Genetic Algorithm (GA) was first advanced by J.H.Holland professor in Michigan University in 1975. It derived from Darwinian evolution, Mendelian genetics and Weizmann species selection theory. The basic idea is to form an algorithm which is aimed at complex system optimization computing, and gives an optimal solution by the procedure searching based on the simulation of natural genetic mechanism and evolution theory [1]. In short, it is a process of repeating the genetic operators mainly include selection, crossover and mutation, according to a certain probability, and evolving the individuals of the population continuously to produce more excellent population.

GA is a kind of non numeric parallel algorithm. In recent years, in the aspects of solving the continuous variable function optimization problems and the combination of discrete variable optimization problems, GA shows high robustness, global optimality, implicit parallelism and adaptability [2]. Now as an intelligent optimization method, it is widely used in many fields such as in the application of solving the Traveling Salesman Problem (TSP) which is very famous and dramatic.

II. BRIEF DESCRIPTION ABOUT TSP

TSP was first proposed in 1800s by W.R.Hamilton and the British mathematician Thomas Kirkman [3]. It can be simply described as: a traveling salesman is going to n cities to promote his goods. He starts from some city c_0, and has to visit all of the rest cities only one time each. Finally he returns back to the starting city c_0. The problem is that we need to find the shortest route which meets the above requirements.

In this paper, $R=(c_0,c_1,\cdots c_{n-1})$ represents the route. If R meets the condition that the value of formula (1) is the minimum, R is the best route.

$$f(R)=\sum_{i=0}^{n-2} d(c_i,c_{i+1})+d(c_{n-1},c_0) \quad (1)$$

Among which, c_i represents No. i city, $i=(0,1,2\cdots n-1)$, n is the number of the cities. $d(c_i,c_j)$ stands for the distance from city c_i to city c_j. According to the practical significance, we know $d(c_i,c_j)=d(c_j,c_i)$.

TSP is not only a very old and difficult problem in the field of combinatorial mathematics but also a typical NP-hard problem in combinatorial optimization. It is easy to get the shortest route with few cities by enumeration. Whereas if n is very giant, there are (n-1)! possible combinations, besides the searching space of the routes will present a trend of explosive growth. Under this condition, it is impossible for us to find the optimal route through general searching method. Thus, a great number of new type optimization computing methods come to the fore, in order to obtain the optimal solution of the TSP. Among which, GA gets the most favor of people and becomes one of the most effective methods to solve the TSP, because of its wide adaptability and the character which is no needs to gain further insight into the problems and depends less on the specific fields of the problem.

III. INTRODUCTION OF GENETIC ALGORITHM

Genetic Algorithm is a process of iterative searching depending on the fitness. A simple GA mainly consists of the following steps:

(1) Confirm the mode of coding, encode each individual and then create the initial population.

(2) Compute the fitness value of each individual in the population.

This work was supported by the Doctoral Fund of Ministry of Education of China (Grant No. 200801510001), the National Natural Science Foundation of China (Grant No. 70940008).

978-0-7695-4335-2/11 $26.00 © 2011 IEEE
DOI 10.1109/CSO.2011.46

(3) According to the individual fitness value, select some individuals from the population through selection operator, and then obtain the parent individuals.

(4) According to some certain crossover rate p_c, perform the crossover operator to the individuals in the population, and then get the new generation.

(5) According to some certain mutation rate p_m, perform the mutation operator to the individual in the population.

(6) Repeat step (2) to step (5) until meeting the termination conditions-the fitness value reaches the predefined value or the iteration number exceeds the maximum.

IV. A NEW METHOD DESIGNED FOR SOVLING TSP

A. Encoding and Initialization - the Population

Encoding is the top problem in GA, and it is also the critical step when designing the algorithm. Encoding method determines how to carry out the operations and affect the efficiency of the algorithm to a large extent [4]. In TSP, there are some common encoding methods, including adjacency representation, ordinal representation, path representation, matrix representation, edge representation and so on [5].

In this paper, the path representation which is also a relatively common encoding method was adopted. The order of traveling to different cities is regarded as the individual encoding. For example, for some TSP problem with 8 cities, one of the individual encoding is (4,2,0,7,3,6,1,5), indicates the traveling route starts from city 4, then passes the following cities in turn, city 2, city 0,…,city 5, and finally returns back to city 4. Each city was accessed only one time.

After confirming the encoding mechanism, randomly generate the initial population with the number of the setting value *popsize*, based on the above mechanism.

B. Fitness Function and Fitness Value

During the evolution searching process, Genetic Algorithm seldom makes use of the external information, while the fitness function becomes the only fundamental criterion. That is to say, the algorithm realizes the searching process on the basis of the individual fitness value. Hence, the constitution of the fitness function is vital to the algorithm. It will not only directly affect the convergence rate of GA, but also relate to the fact that whether or not we can find the optimal solution. Fitness function varies case by case, and can be set depended on different concrete problems. Generally speaking, fitness function is transformed from the target function [4].

TSP problem aims at finding out the individual with the minimum value of the objective function. As a result, we can treat the reciprocal of the objective function as the fitness function, and the expression is $f(i) = 1/f(R_i)$. The longer the route length is, the smaller the fitness value is, and the lower the selected probability will be.

C. Selection Operator

The main content of the genetic algorithm is composed of three basic genetic operators, which are selection operator, crossover operator and mutation operator.

The major task of the selection operator is to select the individuals with higher fitness value preferentially, so that they can be reserved to the next generation, while the weaker individuals will be more easily eliminated. From this point, the property that the genetic algorithm imitates biological evolution is quite obvious, and the superior and inferior mechanism manifested itself clearly. The frequently-used operators include roulette wheel selection, random sampling, truncation selection, tournament selection, the strategy of best-individual to survive and so on.

In this paper, we adopt both of the roulette wheel selection [4] and the strategy of best-individual to survive. That is, first to make sure that the individual with the highest fitness value will be preserved into the next generation, and then calculate the rate P_i that each of the rest individuals is going to be selected in the whole population and the cumulative probability Q_i respectively.

$$P_i = f(i) \Big/ \sum_{i=1}^{popsize} f(i) . \tag{2}$$

$$Q_i = \sum_{j=1}^{i} P_j . \tag{3}$$

According to the comparison between random number *r* and Q_i, we determine which one should be retained to the next generation.

D. Crossover Operator

To put it simply, crossover operator is to generate new offspring. Its main process is to exchange partial genes of two paternal chromosomes which are randomly grouped and represented by *x* and *y* here based on some rules, and to produce two offspring. As for TSP, the most commonly used crossover operator contains Partially Matched Crossover, Order Crossover, Cycle Crossover, and etc. [5].

In this paper, we adopt a kind of heuristic crossover operator, and design the algorithm as follows [6]: first of all, with a predefined probability p_c which stands for the rate of crossover, pick out a certain amount of individuals as the parental crossover chromosomes, and then randomly make the parental chromosomes in pairs prepared for the crossover operator. Parental chromosomes are respectively marked as *x* and *y* in each pair. Next, randomly definite a position we can marked with city *c* as the starting point of the child traveling sequence. And now, the following paragraph gives a detailed introduction on how to get the offspring represented by *childx* and *childy*.

Generate new *childx*: calculate d_x and d_y which respectively means the distance between city *c* and the next

adjacent city c_x, c_y in route *x* and route *y*. Compare d_x with d_y. If $d_x \le d_y$ is true, c_x will be added to *childx*, else c_y will join in. After that, remove the city selected just now both in route *x* and in route *y*, in the meanwhile, set this city as the new city *c*, and repeat the above steps until there is only one city left in the parental chromosomes. Add the last city into *childx*, and now we obtain the first offspring *childx*. We must realize that the route is a closed-loop, which means the starting point is also the ending point.

Almost in the same way, we can generate new *childy* , and the only difference is that the meanings of d_x and d_y have changed. Under this condition, d_x means the distance between city *c* and the previous adjacent city c_x' in route *x*, not the next one, and so is d_y.

E. Mutation Operator

The selection and crossover operators have almost completed most of the searching process of Genetic Algorithm. But this is not enough; usually we still need to perform the mutation operator which is going to change some certain partial chromosome' genes so that it can keep the algorithm approaching to the optimal solution. Therefore, mutation is an inseparable part of the algorithm despite its low probability. As to TSP, the mutation operator consists of inversion, insertion, shift, and swap [2].

In this paper, we adopt the inversion operator. There is a brief instruction about the operator: get the mutation individual at random on the basis of the mutation rate. Then randomly generate two mutation positions, and arrange the genes between the two positions in reverse. After that, a new individual is generated by the mutation operator.

F. Termination Conditions

Genetic Algorithm is an iterative searching method. Typically, the solution obtained by this algorithm is the satisfactory solution but not the quite best one, so we have to definite the termination conditions for the algorithm.

In this paper, we take the iterations as the judging standard of convergence. When the iterations of the algorithm equals to the maximum that was predefined, this means the algorithm comes to an end and the current route is the optimal solution.

According to the opinion we described previously, we designed a method to solve the TSP and the algorithm flow chart is shown in Figure 1.

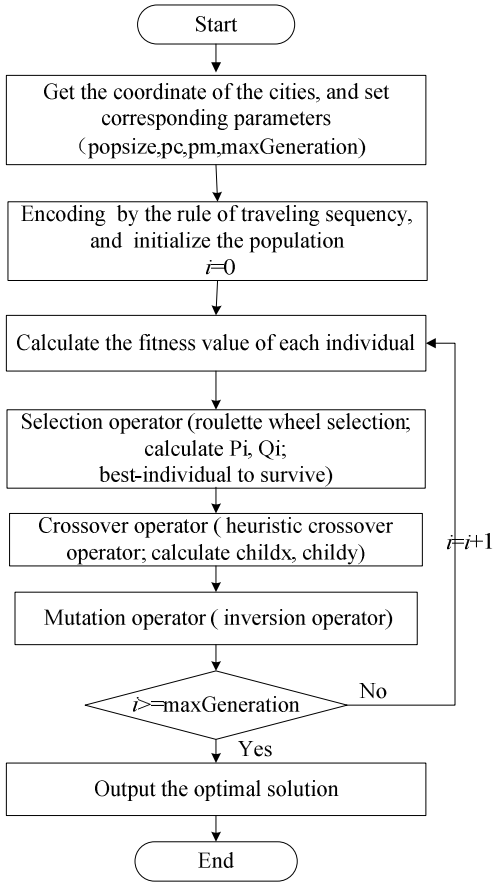

Figure 1. GA flow chart.

V. THE PROGRAM OF THE ALGORITHM

Finally, we realized the algorithm by using C# language with the help of Microsoft Visual Studio 2005. In this program, the input data was the coordinates of 50 cities (TABLE I [7].) stored in the database of SQL Server2005.

TABLE I. COORDINATES OF FIFTY CITIES

city	x	y	city	x	y
c0	10	75	c25	66	43
c1	36	9	c26	48	26
c2	91	78	c27	13	33
c3	54	53	c28	11	53
c4	8	51	c29	47	65
c5	78	51	c30	15	70
c6	73	14	c31	37	91
c7	23	56	c32	63	54
c8	15	40	c33	46	42
c9	26	60	c34	66	8
c10	74	98	c35	52	28

c11	91	16	c36	61	34
c12	92	75	c37	70	13
c13	63	73	c38	91	55
c14	95	16	c39	36	60
c15	21	84	c40	20	20
c16	4	23	c41	39	2
c17	41	29	c42	29	90
c18	36	74	c43	87	46
c19	56	15	c44	62	87
c20	14	18	c45	70	92
c21	26	80	c46	19	74
c22	49	13	c47	32	10
c23	24	94	c48	81	84
c24	5	92	c49	73	90

In the program, we define the value of the parameters as follows: the capacity of the initial population is 30, the maximum of the iterations is 500, the probability of crossover is 0.9, and the rate of mutation is 0.01. Through the procedure, we get the optimal solution and the shortest length equals 554.282 which proved to be improved than the result 561.851 in [7].

At the same time, there are four analog programming graphs presented below. They are the initial route graph, the intermediate stage graph and the optimal solution graph.

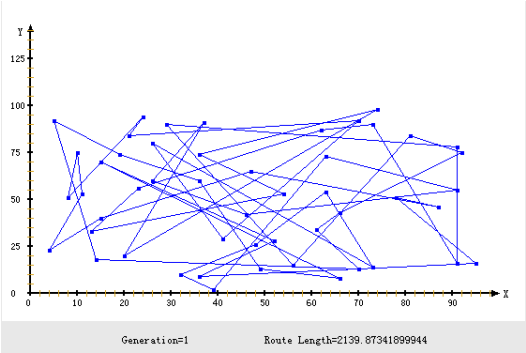

Figure 2. Route of the first iteration.

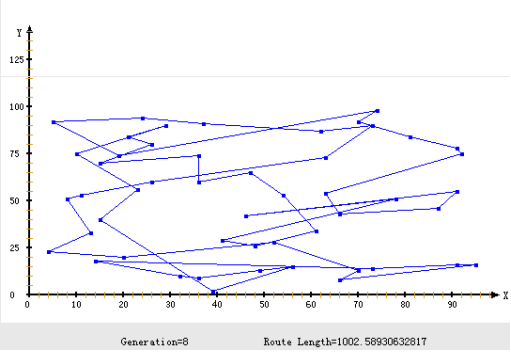

Figure 3. Route of the 8th iteration.

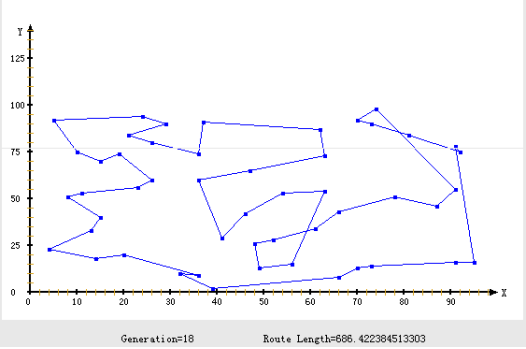

Figure 4. Route of the 18th iteration.

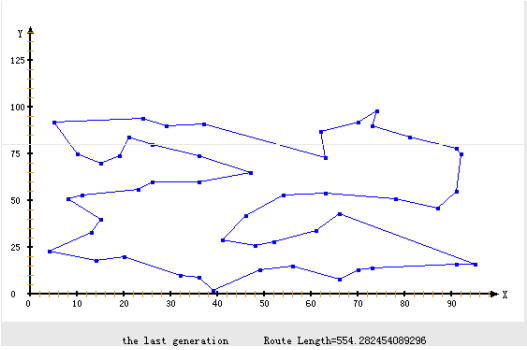

Figure 5. Route of the last iteration.

From this program, we can distinctly get the satisfactory solution. The algorithm based on this paper shows high capability of overall search by comparison, which further indicates that GA is more adaptive, stable and closer to the optimization, and it is able to solve the TSP problem successfully and effectively as well.

VI. CONCLUSIONS

Being compared with other algorithms, GA possesses excellent advantages in many fields. In this paper, we designed a new algorithm structure with the incorporation of three kinds of GA operators, which includes the combination of two selection operators, heuristic crossover operator and inversion mutation operator. To some extent, the quality and efficiency of the solution to solve the TSP problem has a

relatively great increasing. Of course, there's got to be a lot of other improved algorithms based on the properties of GA, and even some concrete algorithms combined with other complex methods. All these need our further efforts.

REFERENCES

[1] X. Bian, and L. Mi, "Development on genetic algorithm theory and its applications," Application Research of Computers, vol. 27, no. 7, pp. 2425–2429, 2010.

[2] B. Chen, and H.Z. Xu, "An Improved Genetic Algorithm and Its Application in TSP," Computer Engineering, vol. 28, no. 9, pp. 90–92,. 2002.

[3] D. Li, and H.X. Sun, "An Application Research of TSP Based on Genetic Algorithm," Science Technology of Heilongjiang Province, no. 13, pp. 27, 2009.

[4] X.Y. Lai, "Genetic Algorithm and Its Application," China Computer and Communication, pp. 117–119, 2010.

[5] C. Fang, "Genetic Algorithm and Its Application in TSP Problem," Journal of Sichuan College of Education, vol. 24, no. 1, pp. 110–112, 2008.

[6] D.P. Peng, Z.Y. Lin, and J.Q. Wang, "An Improved Genetic Algorithm for TSP Problem," Computer Engineering and Applications, no.13, pp. 91–93, 2006.

[7] L.X. Yan, "Solving Traveling Salesman Problem with Line-up Competition Algorithm," Operations Research and Management Science, vol. 8, no. 3, pp. 24–30, 1999.