A hybrid genetic algorithm for solving bi-objective traveling salesman problems

Mei Ma and Hecheng Li
School of Mathematics and Statistics, Qinghai Normal University, Xining, 810008, China.
Email: 1450670013@qq.com

Abstract. The traveling salesman problem (TSP) is a typical combinatorial optimization problem, in a traditional TSP only tour distance is taken as a unique objective to be minimized. When more than one optimization objective arises, the problem is known as a multi-objective TSP. In the present paper, a bi-objective traveling salesman problem (BOTSP) is taken into account, where both the distance and the cost are taken as optimization objectives. In order to efficiently solve the problem, a hybrid genetic algorithm is proposed. Firstly, two satisfaction degree indices are provided for each edge by considering the influences of the distance and the cost weight. The first satisfaction degree is used to select edges in a "rough" way, while the second satisfaction degree is executed for a more "refined" choice. Secondly, two satisfaction degrees are also applied to generate new individuals in the iteration process. Finally, based on genetic algorithm framework as well as 2-opt selection strategy, a hybrid genetic algorithm is proposed. The simulation illustrates the efficiency of the proposed algorithm.

1. Introduction

Traveling salesman problem is a typical combinatorial optimization problem. The problem can be described as follows: Given a list of n cities and the distances between each pair of cities, a salesman starts from some city, and then visit all other cities exactly once and returns to the origin city. How does the salesman plan his/her travelling rout such that the tour distance is the shortest among all possible routes? Despite the fact that the description is very simple, the problem is extremely difficult to solve, and has been proven to be a NP-hard problem [1]. In real life a lot of problems can be modelled as TSP problems, such as gene sequencing problems, the dartboard design problem, the hole punching problem, etc. [2, 3, 4]. In these TSP models, the single objective TSP, only minimizing tour distance, is the most common version.

In real world application, some problems always involve more than optimization objective, such as in a real trip, the tourists maybe take distance, cost and other comfort index into account at the same time. As a result, it is necessary to study some multi-objective TSP models and develop efficient solution methods for these problems. In recent 20 years, the multi-objective traveling salesman problems (MOTSPs) have been widely studied. One of the first papers considering the MOTSP is the study of Borges and Hansen [5], where the global convexity for a MOTSP was studied and the landscape of local optima was analysed.

In recently, evolutionary algorithm has been applied to solve MOTSP. Yan [6] proposed a new MOEA in 2003, in which an “inver-over” operator was defined to replace the traditional crossover and mutation operators, two kinds of selection operators were adopted, and niches are defined by using a sharing function to preserve the diversity of the population. The simulation results show that MOEA outperforms SPEA in some bi-objective instances. Elaoud [7] proposed a multi-objective genetic...
algorithm based on a dynamic selection of crossover and mutation operators. Experimental results show synergy effects among different operators. Samanlioglua [8] used a memetic random key genetic algorithm to find weakly Pareto optimal solutions for a symmetric multi-objective traveling salesman problem. The memetic algorithm adopted a “target-vector approach” and the local search. The random keys representation ensured that feasible tours are maintained during the application of genetic operators. The simulation results show the algorithm is better than that in [9]. Gao [10] developed a new approach, which combined MOEA/D with probabilistic model based methods for MOTSP. In this approach, an MOTSP is divided into a set of scalar objective sub-problems, and both priori and learned information are used to build a probabilistic model for each sub-problem. By the cooperation of neighbour sub-problems, MEDA/D could optimize all sub-problems simultaneously.

In [11], the authors considered the application of estimation of distribution algorithm (EDA) and genetic algorithm (GA) for bi-objective travelling salesman problem (BOTSP). In their paper, three existing EDAs are improved, and the refinement operator and a local search operator are developed. Subsequently, the EDAs are combined with GA to overcome the shortcomings of these two algorithms. The results show that the hybrid algorithms are capable of finding a set of good trade-off solutions. Psychas [12] dealt with multi-objective TSP using three hybrid evolutionary algorithms, and solved 2–5 objective TSP. Lust and Teghen [13] designed two phase Pareto local search (2PPLS) to find a good approximation of the efficient set of the BOTSP, in which PLS is a powerful tool to generate potentially efficient solutions.

Some variants of MOTSP are also studied. Wang et al. [14] described an uncertainty and multiple objectives in traveling salesman problem (UMTSP). They proposed a new approach to obtain Pareto efficient route in the uncertain multi-objective TSP, which converts the uncertain multi-objective TSP into an uncertain single objective TSP. A new ABC algorithm combined with genetic operators is designed to this problem. Li [15] introduced a parallel search system for dynamic multi-objective traveling salesman problem (DMO-TSP). He designed a multi-objective TSP in a stochastic dynamic environment. Yang et al. [16] proposed a multi-algorithm solver called multi-algorithm co-evolution strategy (MACS) for DMO-TSP, which can be used for solving small size DMO-TSP with low degree of conflict objectives in real time. In order to solve larger size DMO-TSP, Yang et al. [17] proposed a synchronized parallel multi-algorithm solver.

In the present manuscript, we consider a BOTSP in which both the distance and the cost are minimized, and proposed an efficient genetic algorithm based on two designed satisfaction degree indices of each edge. These two satisfaction degree indices are used to select edges and generate new individuals in the iteration process. Based on genetic algorithm framework as well as 2-opt selection strategy, a hybrid genetic algorithm is proposed.

This paper is organized as follows. The discussed problem is presented in Section 2. A hybrid genetic algorithm is given by applying 2-opt selection strategy in Section 3, and this is followed by the simulation in Section 4. We finally conclude our paper in Section 5.

2. The Bi-objective TSP Problem

BOTSP can be described as follows: give $N$ cities and two costs $c^{1}_{ij}$ and $c^{2}_{ij}$, $i, j = 1, 2, ..., N$, on the edge from city $i$ to city $j$. The purpose of solving BOTSP is to find a tour, i.e. a cyclic permutation $\rho$ of the $N$ cities, such that both the distance and the cost are minimized. The mathematical formulation can be written as

$$
\min \quad f_1(\rho) = \sum_{i=1}^{N-1} c^{1}_{\rho(i), \rho(i+1)} + c^{1}_{\rho(N), \rho(1)}
$$

$$
\min \quad f_2(\rho) = \sum_{i=1}^{N-1} c^{2}_{\rho(i), \rho(i+1)} + c^{2}_{\rho(N), \rho(1)}
$$

In the problem as above, two objectives always conflict just as in other multi-objective optimization. When any one is minimized, the other maybe obtains the worse objective value. As a result, the optimal solution set of (1) is a set of trade-off solutions, i.e. Pareto optimal set.
3. Proposed Algorithm
In the section, we present a hybrid genetic algorithm (Algorithm 1) using two designed satisfaction degree indices.

3.1. The Two Satisfaction Degree Indices
In order to produce a good initial solution, we want the two weights to be as small as possible. We divide the two weights into three levels: high, middle and low, respectively. The distance weight classification is as follows:

Firstly, to sort the values in the distance matrix in descending order, and get the corresponding sequence \( S \), the number of elements of \( S \) is \( |S| \).

Secondly, to divide the sequence \( S \) from left to right into three groups \( S_1, S_2, S_3 \), and the element numbers in these groups are \( \frac{|S|}{3}, \frac{|S|}{3} \) and \( |S| - 2 \frac{|S|}{3} \), respectively. In these three groups, the elements in \( S_1 \) have largest distances while \( S_3 \) the smallest distances.

Similarly, the cost matrix is divided into \( T_1, T_2 \) and \( T_3 \).

Thus, according to the levels of distance and cost weights, all edges can be divided into nine categories: low-low, low-high, low-middle, middle-low, middle-middle, middle-high, high-low, high-middle, and high-high. Among them the former is the level of the distance weight, while the latter is the level of cost weight. In the paper, based on some numerical experiments, we suggest taking a value for each category as the first satisfaction degree, and these suggested values are shown in

| Table 1. Values of the first satisfaction degree |
|-----------------------------------------------|
| Cost | high | medium | low |
|------|------|--------|-----|
| high | 0.1  | 0.3    | 0.7 |
| medium | 0.3 | 0.5    | 0.8 |
| low  | 0.7  | 0.8    | 0.9 |

From the Table 1, one can see that the lower the weight on the edge, the larger the first satisfaction degree indicator, and so these edges are more likely to be chosen. When two weights are located at the same level, the nine categories of the edges can be reduced to six categories (low- low, low- medium, low- high, medium - medium, high- medium, high- high). In the process of edge selection, we always choose the edge which has the larger value of the first satisfaction degree. But in the same class, in order to choose the edge as good as possible, the second satisfaction degree is executed for a more “refined” choice.

Given weight matrices \( C = (c_{ij})_{N \times N} \) and \( D = (d_{ij})_{N \times N} \). In order to define the second satisfaction degree effectively, a normalization procedure need to be executed as follows:

\[
c'_{ij} = \frac{c_{ij}}{\sum_{k=1, k \neq i}^{N} c_{ik}}, \quad d'_{ij} = \frac{d_{ij}}{\sum_{k=1, k \neq i}^{N} d_{ik}}
\]

The second satisfaction degree of any two cities is defined as:

\[
F(i, j) = 2 - d'_{ij} - c'_{ij}
\]

3.2. Population Initialization Based on Satisfaction Degree
In the execution process of genetic algorithms, the random generation of the population individuals can sometime affect the evolution of the populations. In this manuscript two satisfaction degrees as above are adopted to generate new individuals.
Set $L = (l_{ij})_{N \times N}$ is the first satisfaction degree matrix, $l_{ij}$ is taken from a set composed of the different elements in Table 1, i.e. $l_{ij} \in \{0.9, 0.8, 0.7, 0.5, 0.3, 0.1\}$. The second satisfaction degree matrix is defined as $F(i, j)$. The details are given as follows:

1) Set $X = \{1, 2, ..., N\}, Y = \emptyset$. First, to randomly choose an integer $t$ from 1 to $N$ as the start city, then set $X = X \setminus \{t\}, Y = Y \cup \{t\}$.

2) Determine city which follows city $\tau$. If there exists an unique $\tau \in X$ such that $l_{ti} = \max\{l_{tk}, k \in X\}$, then set $X = X \setminus \{i\}, Y = Y \cup \{i\}$. Otherwise, set $X_1 = \{i | l_{ti} = \max\{l_{t,j}, j \in X\}$, and use the roulette wheel selection to decide the city behind the city $\tau$:

   i) Calculate the probability of the elements in the set $X_1$,
   \[ p(i) = \frac{F(t,i)}{\sum_{j=1}^{N} F(t,j)}, \quad i \in X_1 \]  
   (4)

   ii) Calculate the cumulative probability of each element in the set $X_1$
   \[ q(i) = \sum_{j=1}^{i} P(j) \]  
   (5)

   iii) To generate randomly numbers $r$ in $[0, 1]$;

   iv) If $q(i - 1) < r \leq q(i), i = 2, ..., n$, then set $X = X \setminus \{i\}, Y = Y \cup \{i\}$;

   v) Repeat step 2) until the set $X$ becomes an empty set, and then an individual is generated;

3) Repeat step 1) and 2) $N_p$ times, and then form the initial population with size $N_p$.

3.3. Crossover Operator

In this subsection, we propose a new crossover operator based on the 2-opt permutation algorithm in which the satisfaction degree information is used. The crossover process is given as follows:

1) Calculate the second satisfaction degree for a given path $\pi = (\pi_1, \pi_2, ..., \pi_n)$;

2) The replacement probability of each edge $(\pi_i, \pi_j)$ is determined according to the second satisfactory degree.

\[ P(\pi_i, \pi_j) = \frac{1}{\sum_{i=1}^{n-1} \frac{1}{F(\pi_i, \pi_j)} + \frac{1}{F(\pi_n, \pi_1)}} \]  
(6)

3) Delete two edges chosen by replacement probability, and add the new edges by 2-opt permutation algorithm.

3.4. Improved Genetic Algorithm

In the subsection, we propose an improved genetic algorithm based on new crossover operator and population initialization procedure.

Step1. (Initial population) using population initialization method to form the initial population $pop(0)$, we denote the crossover probability by $p_c$ and the mutation probability by $p_m$. Let $t = 0$.

Step2. (Fitness) Evaluate each individual in $pop(t)$ by calculating objective values, and select non-dominated solutions according to the dominance and crowding distance just as NSGA-II does.

Step3. (Crossover and mutation) The crossover operator and the insertion mutation are adopted to generate genetic offspring.

Step4. (Next generation of population) The elitist selection scheme is applied to generate the next generation of population $pop(t + 1)$, and the non-dominated set is updated.
Step5. (Stopping criterion) If the termination condition is satisfied, then the algorithm is stopped, and output the dominated solution set; otherwise, let $t = t + 1$, go to Step 3.

4. Simulation

We select the several bi-objective TSP instances to illustrate the efficiency of the proposed algorithm. The instances (oliver30, eil51, rand100, rand300, kroab100, kroac100, krobc100, kroab150, kroab200) are selected from TSPLIB, and the first objective is to minimize the distance. For the instances oliver30, eil51, rand100 and rand300, the second cost between the edges are randomly generated. Three Euclidean instances (Euclab100, Euclab300, Euclab500) are considered, the costs on the edges correspond to the Euclidean distances between two points. We also use three mixed instances (mixed100, mixed300, mixed500), where the first cost comes from Euclidean instances while the second cost is randomly generated.

Two indicators are selected to measure the quality of the solutions.

1. The hyper volume indicator (to be maximized) [18]: the volume of the dominated space is defined by approximations sets, limited by a reference point. In Figure 1, $\{P_1, P_2, P_3\}$ is a set of the non-dominated objective vector, $F_1,F_2$ is the objective function. The hyper volume indicator of this set is the area of ABCDEFGH, where $Ref = (r_1,r_2)$ is a reference point, and $r_i = \max(f_i(x)) + 0.1 \times (\max(f_i(x)) - \min(f_i(x)))$, $i = 1,2$.

![Figure 1. The hyper volume indicator in the two-objective case](image)

2. Relative error: Let the optimal value of the single objective optimization problem be A (only when the first objective is considered in single objective TSPs) and the smallest value of the first objective function obtained by the proposed algorithm be B in bi-objective optimization, then the relative error between A and B can be obtained as follows:

$$e = \frac{A - B}{B}$$

(7)

Genetic parameters are taken as follow: the population size $N_p = 50$, the crossover probability $p_c = 0.85$, the mutation probability $p_m = 0.1$ and the maximum number of generations $G = 200$ for instances olvier 30 and eil51, $G = 500$ for other instances. The proposed algorithm (Algorithm 1) is executed 10 independent runs.

We first compare the results of the Algorithm 1, for the “kro” instances, with 2PPLS in [13]. The results are shown in Table 2, where H represent the average hyper volume of 10 runs, Time represent the average CPU time of 10 runs.
Table 2. Comparison of algorithm 1 with 2PPLS on “kro” instances

| Instances  | Algorithm | $H(10^8)$ | Time(s) | $e$   |
|------------|-----------|-----------|---------|-------|
| kroab100   | Algorithm 1 | 228.52    | 34.83   | -0.0394 |
|            | 2PPLS     | 226.11    | 35.57   |       |
| kroac100   | Algorithm 1 | 223.90    | 28.71   | -0.0323 |
|            | 2PPLS     | 226.32    | 30.3    |       |
| kroad100   | Algorithm 1 | 228.01    | 25.63   | 0.0158 |
|            | 2PPLS     | 227.41    | 26.44   |       |
| krobc100   | Algorithm 1 | 228.45    | 35.47   | 0.0271 |
|            | 2PPLS     | 227.38    | 35.82   |       |
| krobd100   | Algorithm 1 | 228.61    | 35.29   | 0.0017 |
|            | 2PPLS     | 226.12    | 37.18   |       |
| krocd100   | Algorithm 1 | 232.67    | 28.33   | 0.0186 |
|            | 2PPLS     | 230.89    | 27.56   |       |
| kroab150   | Algorithm 1 | 601.49    | 100.37  | 0.0427 |
|            | 2PPLS     | 592.51    | 91.21   |       |
| kroab200   | Algorithm 1 | 1012.19   | 259.41  | 0.0375 |
|            | 2PPLS     | 1076.08   | 211.76  |       |

From Table 2, we can see that the hyper volume indicators on most of the instances are improved except for kroac100 and kroab200, and the relative error is small.

We compare the results of Algorithm 1 with 2PPLS on the other instances in Table 3. For the instances oliver30 and eil51, the relative error is 0. It follows that the proposed algorithm found the same solutions as those provided by the single objective optimization. We can see that the proposed algorithm finds better hyper volume indicators than 2PPLS for these the instances except for rand300, Euclab500 and mixed500. But the running time of 2PPLS is higher on the instances Euclab300, Euclab500 and mixed500.

Table 3. Comparison of algorithm 1 with 2PPLS on the other instances

| Instances  | Algorithm | $H(10^8)$ | Time(s) | $e$   |
|------------|-----------|-----------|---------|-------|
| oliver30   | Algorithm 1 | 24.83     | 7.92    | 0.0   |
|            | 2PPLS     | -         | -       | -     |
| eil51      | Algorithm 1 | 42.91     | 11.35   | 0.0   |
|            | 2PPLS     | -         | -       | -     |
| rand100    | Algorithm 1 | 204.67    | 21.46   | -0.0051 |
|            | 2PPLS     | -         | -       | -     |
| rand300    | Algorithm 1 | 4780.29   | 380.52  | 0.0926 |
|            | 2PPLS     | 4804.58   | 351.27  | -     |
| Euclab100  | Algorithm 1 | 169.12    | 25.17   | 0.0078 |
|            | 2PPLS     | 168.88    | 28.60   | -     |
| Euclab300  | Algorithm 1 | 2309.91   | 459.72  | 0.0129 |
|            | 2PPLS     | 2309.69   | 783.14  | -     |
| Euclab500  | Algorithm 1 | 7163.26   | 5071.50 | 0.0412 |
|            | 2PPLS     | 7165.39   | 6309.10 | -     |
| mixed100   | Algorithm 1 | 334.57    | 25.06   | 0.0281 |
|            | 2PPLS     | 331.99    | 28.28   | -     |
| mixed300   | Algorithm 1 | 3411.99   | 440.27  | 0.0365 |
|            | 2PPLS     | 3410.44   | 447.06  | -     |
| mixed500   | Algorithm 1 | 10420.32  | 1946.81 | 0.0493 |
|            | 2PPLS     | 10440.41  | 2682.45 | -     |

5. Conclusions
In this manuscript two satisfactory degree indices are proposed, and used to generate individuals. Based on these procedures, a hybrid genetic algorithm is developed which is combined with 2-opt approach for solving bi-objective TSP. From the computational results, we can see the proposed
algorithm is efficient and robust.

6. Acknowledgments

The work was supported by the National Natural Science Foundation of China (No. 61463045), the Natural Science Foundation of Qinghai Province (No. 2013-Z-937Q), and the Key Laboratory of the Internet of Things of Qinghai Province.

7. References

[1] Larranaga P, Lozano J A 2002 Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation (Kluwer: Norwell).
[2] Blazewicz J, Kasprzak M and Kuroczycki W 2002 J. Heuristics. vol. 8, pp 495-502.
[3] Eiselt H A, Laporte G 1991 J. OR Society. vol. 42, pp 113–118.
[4] Reinelt G 1989 J. ORSA J Comput. vol. 4, pp 206–217.
[5] Borges P C, Hansen M P 2000 Essays and surveys in metaheuristics (Operations Research/Computer Science, vol. 15) (New York: Springer Science+Business Media), pp 129-150.
[6] Yan Z, Zhang L and Kang L 2003 Proc. 2nd Int. Conf. on Evolutionary Multi-Criterion Optimization (Berlin, Heidelberg: Springer-Verlag), pp 342-354.
[7] Elaoud S, Teghem J and Loukil T M 2010 J. Electronic Notes in Discrete Mathematics. vol. 36, pp 939-946.
[8] Samanlioglu F, Ferrell Jr W G and Kurz M E 2008 J. Comput. Ind. Eng. vol. 55, pp 439-449.
[9] Hansen M P 2000 J. Heuristics. vol. 6, pp 419-430.
[10] Li Y, Zhou A M and Zhang G X 2013 J. Comput. Math. Appl. vol. 66, pp 1857-1868.
[11] Ann S V, Kay T C and Yong C J 2011 J. Flex Serv Manuf. vol. 23, pp 207–241.
[12] Iraklis-Dimitrios P, Eleni D and Yannis M. 2015, J. Expert Syst. Appl. vol. 42, pp 8956–8970.
[13] Lust T, Teghem J 2010 J. Heuristics. vol. 16, pp 475-510.
[14] Wang Z T, Guo J S and Zheng M F 2015 J. Eur. J Oper. Res. vol. 241, pp 478–489.
[15] Li W Q 2014 J. Mathematics and System Science. vol. 4, pp 295-314.
[16] Yang M, Kang L S and Guan J 2008 Proc. Cong. on Evolutionary Computation (CEC08), pp 466-471.
[17] Yang M, Kang L S and Guan J 2008 Proc. 7th WSEAS Int. Conf. on Applied Computer & Applied Computational Science (ACACOS '08), pp 288-293.
[18] Zitzler E 1999 Evolutionary algorithms for multi-objective optimization: methods and applications (PHD thesis). (Zurich, Switzerland: Swiss Federal Institute of Technology (ETH)).