International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:1, No:1, 2007

A Comparison between Heuristic and Meta-Heuristic Methods for Solving the Multiple Traveling Salesman Problem

San Nah Sze, and Wei King Tiong

***Abstract*—**The multiple traveling salesman problem (mTSP) can be used to model many practical problems. The mTSP is more complicated than the traveling salesman problem (TSP) because it requires determining which cities to assign to each salesman, as well as the optimal ordering of the cities within each salesman's tour. Previous studies proposed that Genetic Algorithm (GA), Integer Programming (IP) and several neural network (NN) approaches could be used to solve mTSP. This paper compared the results for mTSP, solved with Genetic Algorithm (GA) and Nearest Neighbor Algorithm (NNA). The number of cities is clustered into a few groups using k-means clustering technique. The number of groups depends on the number of salesman. Then, each group is solved with NNA and GA as an independent TSP. It is found that k-means clustering and NNA are superior to GA in terms of performance (evaluated by fitness function) and computing time.

***Keywords*—**Multiple Traveling Salesman Problem, Genetic Algorithm, Nearest Neighbor Algorithm, k-Means Clustering.

I. Introduction

THE well-known traveling salesman problem (TSP) can be generalized into multiple traveling salesman problem (mTSP). MTSP consists of determining a set of routes for *m* salesman who all start from and return back to a home city or depot. Previous studies show that mTSP can be solved using Genetic Algorithm (GA) [1], Integer Programming (IP) and several neural network (NN) approaches [2]. The purpose of this paper is to compare the solution of mTSP obtained using *k*-means clustering technique and nearest neighbor algorithm (NNA) and GA.

The organization of this paper is as follows. In Section 2, the approaches and methods that are chosen in this paper are discussed in detail and then the results are presented and discussed in Section 3. Then, the conclusion of this paper is included in Section 4.

Manuscript received November 30, 2006.

S. N. Sze is with the Department of Computational Science and Mathematics, Faculty of Computer Science and Information Technology, University Malaysia Sarawak, Kota Samarahan, Sarawak 94300 Malaysia (phone: 60-82-583665; fax: 60-82-583764; e-mail: snsze@fit.unimas.my).

W. K. Tiong is with Department of Computational Science and Mathematics, Faculty of Computer Science and Information Technology, University Malaysia Sarawak, Kota Samarahan, Sarawak 94300 Malaysia (phone: 60-82-583726; fax: 60-82-583764; e-mail: wktiong@fit.unimas.my).

II. Approaches and Methods

This paper considers mTSP as a symmetric problem. Let the number of cities be *n* and the number of salesman be *m* where $n \geq m$, *n* and *m* are positive real numbers. The *n* cities must be partitioned into *m* tours, with each tour resulting in a TSP for one salesman. All salesmen must start from home city and return to home city again at the end of their tours. The main objectives for solving mTSP are to minimize the total distance for all salesman and also computing time of the solution.

In this paper, mTSP is solved using two approaches. The first approach is to use *k*-means clustering technique to partition the *n* cities into *m* tours. *K*-means is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem [3]. Then, each cluster or tour is solved using NNA as a TSP to find the shortest path for each salesman and thus to minimize the total distance for all salesmen. NNA is chosen because Sze [4] found that the performance of NNA is good if the number of cities is large for TSP.

The second approach is to use standard GA to solve mTSP. In this paper, two-part chromosome representation is chosen as shown in Fig. 1. The first part shows the cities that must be visited by each salesman in sequence while the second part shows the number of cities per salesman.

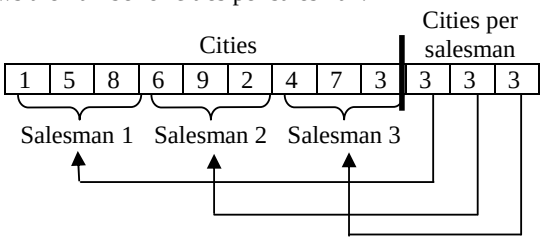

Fig. 1 two-part chromosome representation

If the number of cities can be divided equally among the salesmen, then each salesman has the same number of cities to be visited. Otherwise, the number of cities per salesman needs to be rounded since the number of cities has decimal point after division. However, the number of cities for the last salesman is the remainder from the total number of cities after deducing with the number of cities for previous salesman.

Fitness function, $F(x)$, is used to evaluate the fitness of the path produced by GA. The fitness function is inversely proportionate to the sum of distances between cities and is given in (1). The shorter path has higher fitness value. The constant value 90000 is chosen by trial and error so that it yields significant difference in fitness values for different path.

$$F(x) = \frac{90000}{\sum_{i=1}^{m}\sum_{j=1}^{n}\sqrt{\left(x_{j+1}-x_j\right)^2+\left(y_{j+1}-y_j\right)^2}} \quad (1)$$

A program is developed by Microsoft Visual Basic to solve the mTSP. Results from this program are presented in the following section.

III. Results and Discussion

In this section, some results generated by the program developed are shown. Fig. 2 shows 30 cities generated randomly and three salesmen are assigned to visit the cities once. Each salesman will start from city 0 (home city) and back again to city 0 after the tour is complete. The whole process is repeated for ten times for both approaches to find the best path. GA is programmed with 500 of populations and 500 generations. Table I shows the total fitness value for three salesmen and the computing time for both approaches.

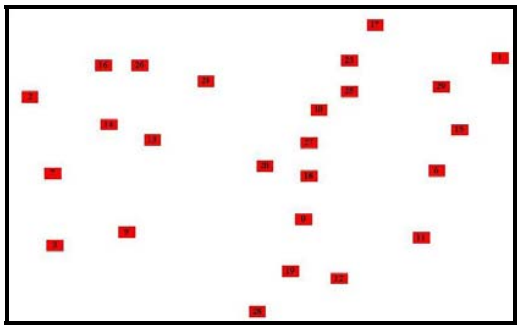

Fig. 2 30 cities are generated randomly

From Table I, the best path for *k*-means clustering and NNA is path 8 where the fitness value is 724 and the computing time is 2 seconds Fig. 3 shows the path produced using NNA. Salesman A starts from city 0 and visit city 18 first followed by city 27, city 10, city 25, city 23, city 17, city 29, city 15, city 16, city 1, city 20 and back again to city 0. The order of cities visited by salesman B and C is shown in Fig. 4.

TABLE I
Total Distance and Computing Time

	k-Means Clustering and NNA		Genetic Algorithm	
No.	Fitness Value	Computing Time (s)	Fitness Value	Computing Time (s)
1	698	2	493	29
2	710	2	517	29
3	700	2	489	29
4	655	2	535	30
5	682	2	542	29
6	638	2	482	29
7	653	2	514	29
8	724	2	516	30
9	700	2	453	30
10	700	2	548	30

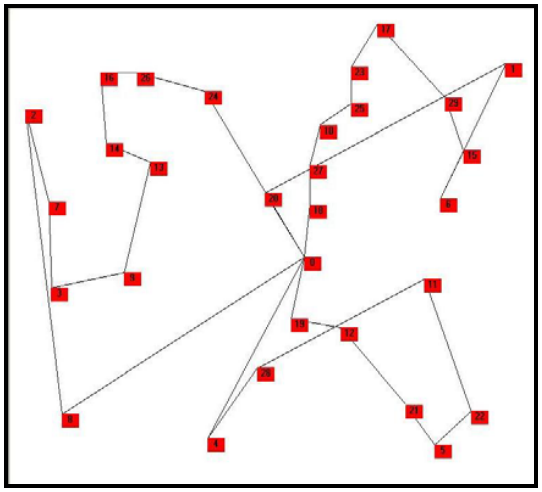

Fig. 3 The best path (*k*-means clustering and NNA)

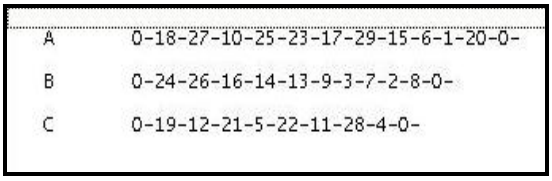

Fig. 4 Order of cities visited for each salesman (*k*-means clustering and NNA)

GA produced the best path at the simulation 10 with the fitness value 548 and computing time is 30 seconds. The best path for GA is shown in Fig. 5. The order of cities visited by each salesman is shown in Fig. 6.

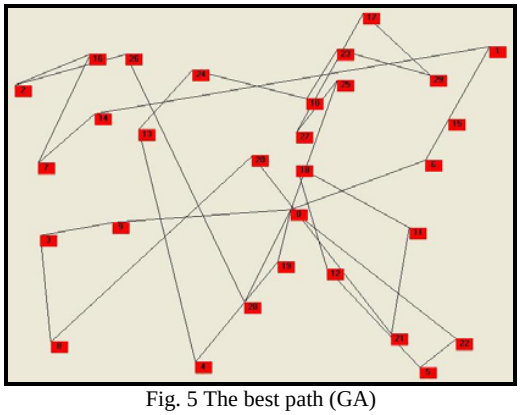

Fig. 5 The best path (GA)

01	0-25-27-17-29-23-10-24-13-4-19-0
02	0-22-5-12-18-11-21-20-8-3-9-0
03	0-6-15-1-14-7-16-2-26-28-0

Fig. 6 Order of cities visited for each salesman (GA)

To study the performance of these two approaches, the number of cities is increased while the number of salesmen remains the same, which are three. Table II shows the fitness value and the computing time of both approaches in this simulation.

The table below shows *k*-means clustering and NNA can solve mTSP even though the number of cities increases without affecting the computing time. The computing time still remains at 2 seconds even if the number of cities increases by 10. The computing time for GA increases as the number of cities increases too. When the number of cities is 90 the program needs about 2 minutes to compute.

TABLE II
FITNESS VALUE AND COMPUTING TIME WHEN THE NUMBER OF CITY INCREASES

No. of City	*k*-means clustering and NNA		Genetic Algorithm	
	Fitness Value	Computing Time (s)	Fitness Value	Computing Time (s)
10	1124	2	1025	12
20	802	2	660	19
30	685	2	465	29
40	697	2	337	40
50	499	2	291	51
60	484	2	208	67
70	441	2	199	80
80	464	2	171	97
90	438	2	149	115
100	395	2	132	138

Fig. 7 shows the path produced by the program for 20 cities for *k*-means clustering and NNA. It is obvious that the path is simple and not complicated. The order of cities that must be visited by each salesman is shown in Fig. 8.

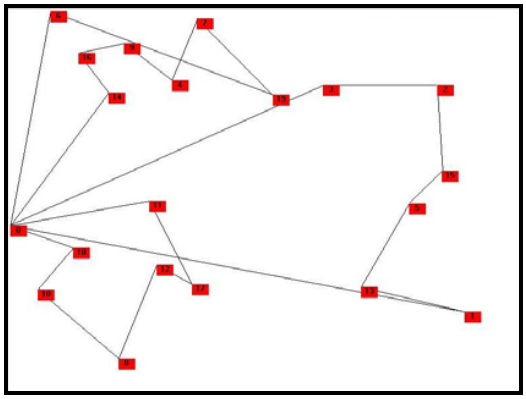

Fig. 7 The path for 20 cities (*k*-means clustering and NNA)

A	0-3-2-15-5-13-1-0-
B	0-18-10-8-12-17-11-0-
C	0-14-16-9-4-7-19-6-0-

Fig. 8 Order of cities for each salesman (*k*-means clustering and NNA)

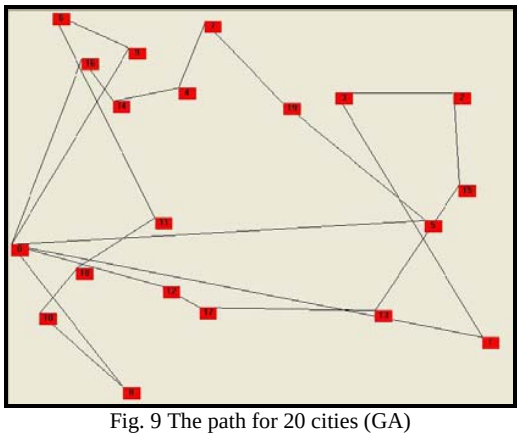

Fig. 9 The path for 20 cities (GA)

The path produced by the GA for 20 cities is shown in Fig 9 and Fig. 10. The result shows that the solution is practical and clear.

01	0-8-10-18-11-6-9-0
02	0-16-14-4-7-19-5-0
03	0-1-3-2-15-13-17-12-0

Fig. 10 Order of cities for each salesman (GA)

When the number of cities is increased to 100 cities, the path produced by *k*-means clustering and NNA is also not complicated. There are not many lines crossing each other. This is shown in Fig. 11 and once again shows that this approach is a good for solving mTSP. Fig. 12 shows the path for each salesman in sequence.

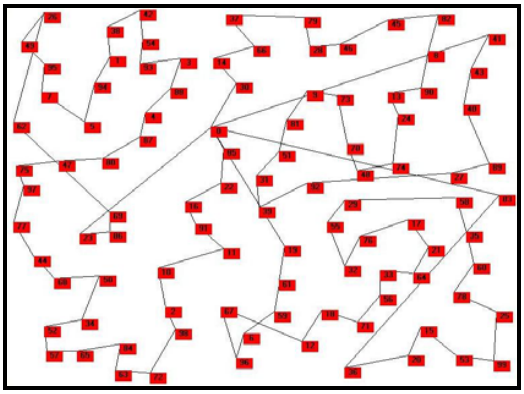

Fig. 11 The path for 100 cities (*k*-means clustering and NNA)

A	0-85-22-16-91-11-10-2-98-72-63-84-65-57-52-34-50-6 8-44-77-97-75-47-80-87-4-88-3-93-54-42-38-1-94-5-7
B	0-30-14-66-37-79-28-46-45-82-8-90-13-24-74-48-70-7 3-9-81-51-31-39-92-27-89-40-43-41-0-
C	0-19-61-59-6-96-67-12-18-71-56-33-64-21-17-76-32-5 5-29-58-35-60-78-25-99-53-15-20-36-83-0-

Fig. 12 Order of cities for each salesman (*k*-means clustering and NNA)

Fig. 13 shows that the path constructed by GA for 100 cities. The path is messy and definitely longer for all salesmen. The path for each salesman in sequence is shown in Fig. 14.

K-means clustering technique and NNA are easy to understand and apply. Since mTSP involves more than one salesman, the assignment of cities to each salesman has become a major problem in mTSP. However this can be easily solved by *k*-means clustering technique. *K*-means clustering technique will cluster all cities into a few clusters and will directly put the closer cities together in one cluster. Thus, NNA can be easily applied in this situation. However, *k*-means clustering can vary significantly depending on initial choice of centroids. It is very sensitive to the initial conditions.

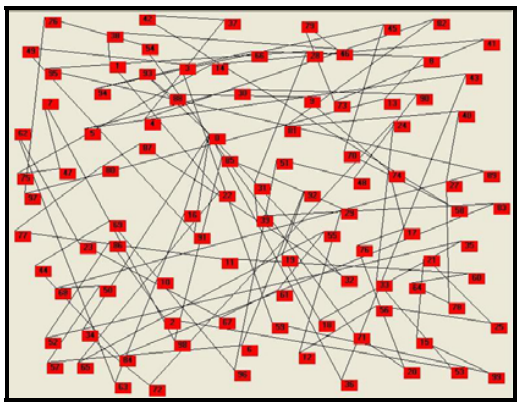

Fig 13 The path for 100 cities (*GA*)

O1	0-34-72-24-33-53-2-8 6-52-35-18-19-92-6-5 7-50-68-89-95-28-31- 39-83-84-44-69-96-51 -48-90-30-5-43-36-0
O2	0-93-88-9-74-17-76-2 0-71-54-26-97-13-8-6 6-94-45-70-58-42-37- 4-82-81-40-27-78-64- 60-11-77-80-87-32-0
O3	0-91-49-3-59-99-15-2 1-25-56-12-29-85-98- 7-47-75-14-41-73-79- 46-38-1-22-65-55-61- 67-23-62-63-10-16-0

Fig. 14 Order of cities for each salesman (GA)

NNA is commonly known as greedy algorithm and only performs well at the beginning construction of the traveling path. Thus, the solution produced by NNA is not necessarily an optimal solution.

GA works by generating a population of numeric vectors (called chromosomes), each representing a feasible solution to the problem. New chromosomes are created by crossover and mutation. Chromosomes are then evaluated according to the fitness function with the fittest surviving and less fit being eliminated. The result is a gene pool that evolves over time to produce better and better solutions to a problem. The key to finding a good solution using a GA lies in developing a good chromosome representation of solutions to the problem. A good GA chromosome representation should reduce or eliminate redundant chromosome from the population. In this study, the two-part chromosome representative can reduce the solution space to $n!\binom{n-1}{m-1}$.

However, GA performs worse than k-means clustering and NNA due to GA's randomness of search direction in big solution spaces. K-means clustering and NNA not only perform better than GA in term of result but also the shorter computing time.

IV. Conclusion

In general, it is found that *k*-means clustering and NNA are superior to GA in terms of performance (evaluated by fitness function) and computing time. *K*-means clustering and NNA are much simpler to understand and apply to mTSP compare to complex algorithms, such as GA.

References

[1] E. Carter and C. T. Ragsdale, "A new approach to solving the multiple traveling salesperson problem using genetic algorithms," *European Journal of Operational Research*, vol. 174, pp. 246-257, 2005.

[2] T. Bektas, "The multiple traveling salesman problem: an overview of the formulations and solution procedures," *Omega*, vol. 34, pp. 209-219, 2006.

[3] M. Matteucci. (2006, October 18). A Tutorial on Clustering Algorithms [Online]. Available: http://www.elet.polimi.it/upload/matteucc/Clustering/tutorial_html/kmeans.html,

[4] S. N. Sze, "Study on Genetic Algorithms and Heuristic Method for Solving Traveling Salesman Problem," M.S. dissertation, Faculty of Science, Universiti Teknologi Malaysia, Johor, Malaysia, 2004.