Comparison of Algorithms for Solving Traveling Salesman Problem

Haider A Abdulkarim, Ibrahim F Alshammari

Abstract— ***Travel Salesman Problem is one of the most known optimization problems. While an optimal solution cannot be reached, non-optimal solutions approach optimality and keep running time fast. In this paper, the most used algorithms to solve this problem are comparedin terms of route length, elapsed time and number of iterations. The TSP is simulated using different scenarios examples and the convergence is checked for each case.***

Index Terms—TSP, Nearest Neighbor, Genetic Algorithm.

I. INTRODUCTION

Travel Salesman Problem (TSP) was first formulated in1930 by Karl Menger and since then it became one ofthe most studied problems in optimization. The problem isdescribed as follows: given a set of cities, the travel cost(distance) between each two cities and one salesman, theoptimum voyage path is calculated, which the salesman shouldfollow to visit every city exactly once and return to thestarting city, while maintaining the travel cost at minimum. The applications to TSP extend well beyond a simple salesmantour to vehicle routing, logistics, printing and soldering of PCBboards. The TSP is considered Non-Deterministic Polynomial-time hard (NP-hard). To solve it, both exact and approximatesolutions exist. The TSP exact solutions imply trying allthe permutation combinations with a complexity of O(n!). Therefore, even with a small number of cities (n=10) therunning time combinations is 3628800, which is unpractical. The approximate solution, on the other hand, uses heuristicalgorithms to find an approximation to the optimal solutionin less number of steps and hence reduced complexity. Theheuristic solution might not be optimal (shortest path andminimum cost) but is still considered valuable in that itrequires shorter time compared to the optimum one. Theheuristic algorithms might also be used in conjunction withother optimization algorithms to optimize the solution.

II. HEURISTIC ALGORITHMS TO SOLVE TSP

There are many algorithms used in dynamic programming to solve the TSP. Since finding the optimum solution might not be feasible (large number of cities), Held-Karp lower bound is used to evaluate the performance of a given algorithm. That is, Held-Karp boundary specifies how close a given solution (using a given algorithm) to the optimum solution. The Held-Karp bound is the relaxed solution to the linear programming of the TSP. Usually, HK lower bound is about 0.8% below the optimal tour length.

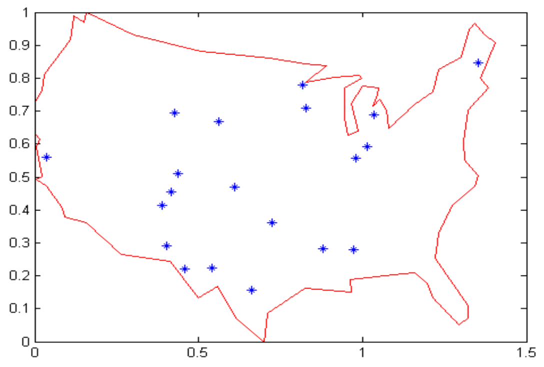

Fig. 1. TSP Example of 20 Cities

A. Nearest Neighbor

it is the simplest heuristic algorithm used to solve TSP. The algorithm can be summarized as follows:

1) Select a random city n and set is as the starting city n0 a random city n and set is as the starting city n0
2) Find the nearest unvisited city and go there
3) Mark the current city as visited
4) Are there any unvisited cities? If yes, go to (2)
5) Return to the starting city

To evaluate the nearest neighbor algorithm solution, the map of the United States is considered. Its area is 9.9 million kilometers. The rectangular area is 4313 km width and 2545 km length. The cities are distributed randomly across that area. The first TSP example is shown in Fig. 1. There are 20 cities (nodes) randomly distributed within 4313-by-2545 Km area. The goal is to calculate the optimal route to visit each city once and return to the starting point. It is worthy to mention that this example is a symmetric TSP in that the cost of travel from city A to city B is exactly the same of that from B to A. The asymmetric TSP considers the non-equality of travel costs from both cities, but it is beyond the scope of this paper. The optimal solution is shown in Fig. 2 with a total combined length of 4616 Km. This solution has the largest complexity and largest number of iterations, yet it finds the shortest route.

Manuscript published on 30 August 2015.
***** Correspondence Author (s)

Haider A Abdulkarim, Scholar (Assistant Lecturer), Computer Techniques Department, Al Israa University College, Baghdad, Iraq.
Ibrahim F Alshammari, Scholar (Assistant Lecturer), Computer Techniques Department, Al Israa University College, Baghdad, Iraq.

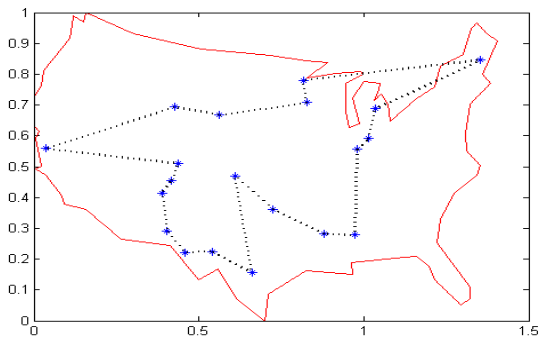

Fig. 2. TSP Example of 20 Cities: Optimum Solution

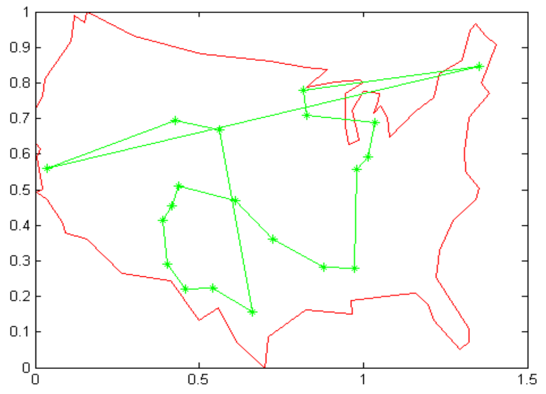

Fig. 3. TSP Example of 20 Cities: Nearest Neighbor

Solving the same example with nearest neighbor algorithm, we obtain the route shown in Fig. 3. The solution has a longer combined length (15800 Km) but finds a solution in O(N^2 log$_2$(N)) iterations, where N is the number of cities to be visited. The nearest neighbor keeps the solution within 25% of the Held-Karp lower bound.

B. Genetic algorithm

This algorithm is well known for solving complex problems, where the optimal solution is hard to find. It resembles the process of natural selection. The algorithm calculates the fitness function for each member of the population. Then it creates new individuals of the population. It uses mutation to add randomization to the process, similar to that of the natural genome. Finally, it selects the individual (solution) with the higher fitness function. Applying genetic algorithm toTSP requires certain limitations. For instance, in every route, each city should not be repeated, otherwise, loops appear. In addition, only valid routes are considered in the algorithm. For instance, a city lies in the far west cannot be considered in the part of the route lying in the far east. The TSP solution for 20 cities using Genetic Algorithm is shown in Fig. 4. With higher number of iterations than the nearest neighbor, Genetic Algorithm is able to find a shorter route (11900 km).

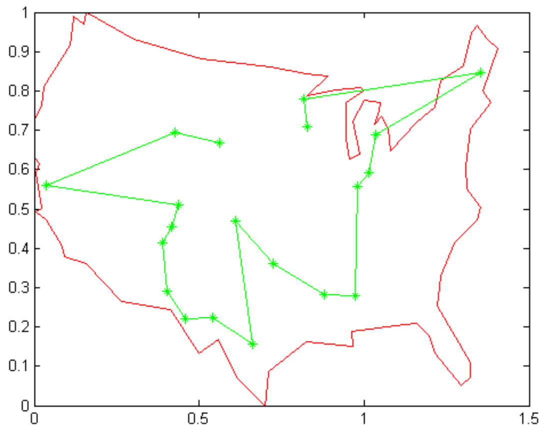

Fig. 4. TSP Example of 20 Cities: Genetic Algorithm

C. Greedy Heuristic Algorithm

This algorithm belongs to the heuristic algorithms category, which searches for the local optima and optimizes the local best solution to find the global optima. It begins by sorting all the edges and then selects the edge with the minimum cost. It continuous selecting the best next choices given a condition that no loops are formed. The computational complexity of the greedy algorithm is O(N ^{2} log$_2$(N)) and there is no guarantee that a global optimum solution is found. On the other hand,the greedy algorithm terminates in a reasonable number ofsteps and keeps the solution within 15 20 % of the Held-Karplower bound. The same TSP example is considered and thetour is calculated using the greedy algorithm as shown in Fig. 5with a combined total length of 12900 Km. Although thissolution has a marginally decreased total distance compared tothe nearest neighbor algorithm, it has yet a higher complexityand execution time. It is worthy to mention that all the algorithms converged,due to the small number of cities and hence the limited numberof route probabilities. Therefore, all of them converge to avalid solution, and by ”valid” we mean not too long route.

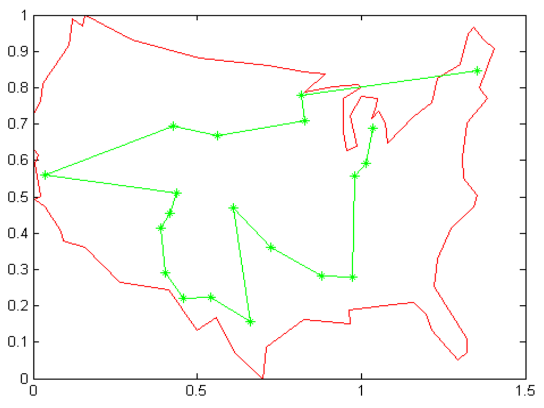

Fig. 5. TSP Example of 20 Cities: Heuristic Algorithm

III. TSP WITH LARGER NUMBER OF CITIES

To approach realistic models, larger number of cities are considered and the corresponding execution time and route lengths are compared. For instance, 100 randomly distributed cities within the US borders are considered, as shown in Fig. 6. The results using the three algorithms and the best calculated routes are shown in Fig. 7, Fig. 8 and Fig. 9, respectively. Table I depicts the resemblance between Genetic and Greedy Heuristic algorithms in terms of the optimal route length. The Greedy Heuristic solution route is shorter by 2168 km. It also finds the fastest solution to the problem (0.18 seconds). On the other hand, Nearest Neighbor approaches relatively longer path (26,664 km) but consumes less iterations than both the Greedy and Genetic algorithm. This is due to the iterative nature of the algorithm. The Genetic algorithm does not guarantee to finds the shortest path, although it approaches it. In this example, again all of the three algorithms converge. To see the difference between the algorithms results clearer, a second example of 1000 randomly distributed cities is considered. After simulation, Table II is obtained. The Greedy Heuristic is again the winner of the shortest path, with a length of 72801 km. The nearest neighbor solution route is longer by 11,137 km but has less computation time. On the other hand, the Genetic algorithm has no guarantee of finding the optimal solution and hence its route is the longest (282866). Hence, the Genetic algorithm does not converge.

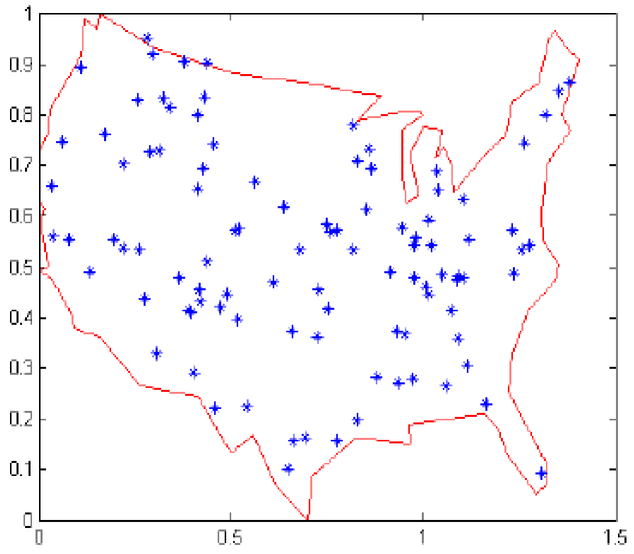

Fig. 6. 100 randomly distributed cities.

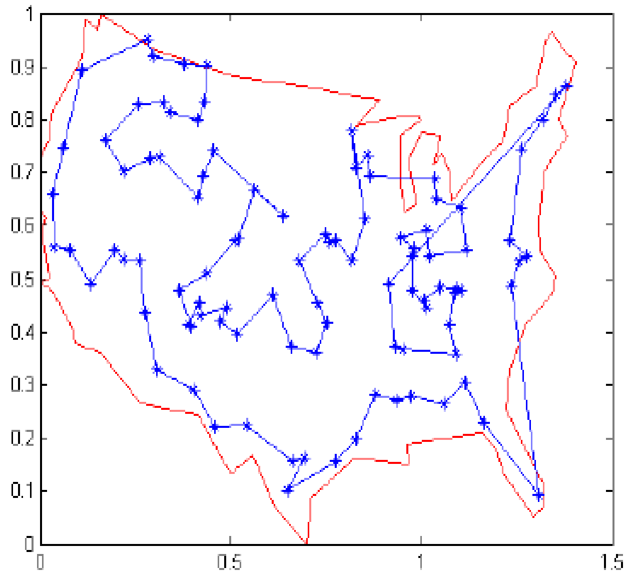

Fig. 7. Nearest neighbor solution to 100 cities problem, unwanted intersections are observed.

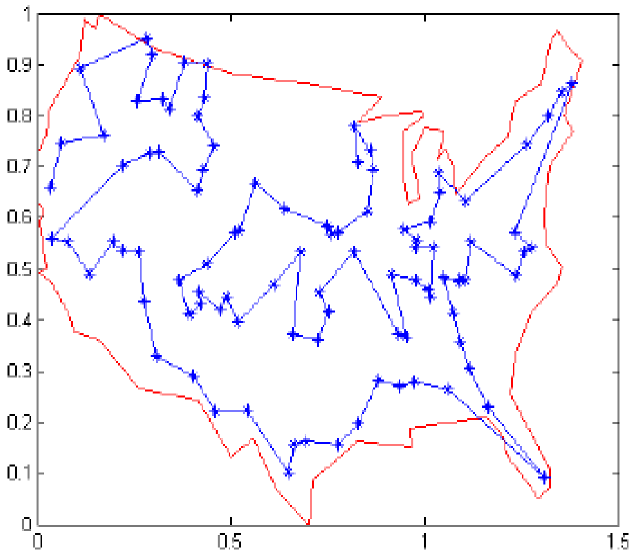

Fig. 8. Genetic algorithm solution to 100 cities problem.

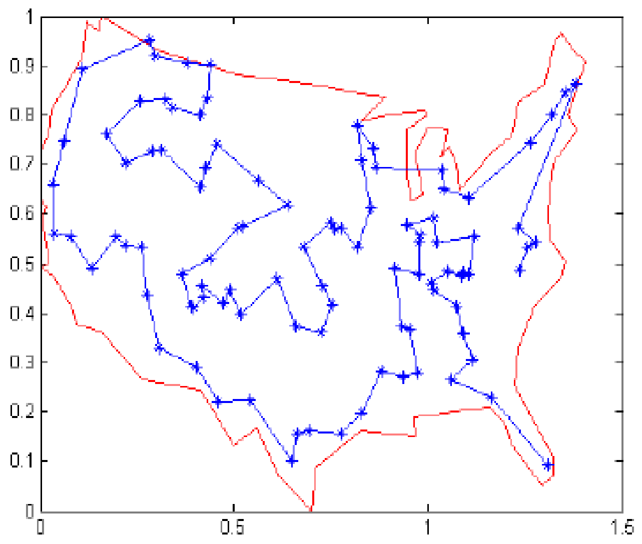

Fig. 9. Heuristic algorithm solution to 100 cities problem.

TABLE I

TSP OF 100 CITIES, ALGORITHM COMPARISON

Algorithm	Optimal route length (km)	Elapsed time (sec)	Iterations
Nearest Neighbor	26664	2.5	100
Genetic	25479	45	10000
Greedy Heuristic	23311	0.07	18

TABLE II

TSP OF 1000 CITIES, ALGORITHM COMPARISON

Algorithm	Optimal route length (km)	Elapsed time (sec)	Iterations
Nearest Neighbor	83938	95.5	1000
Genetic	282866	468	10000
Greedy Heuristic	72801	127	151

IV. CONCLUSION

In this paper, the Travel Salesman Problem is described. Then, some of the most used algorithms to solve it are depicted. Each algorithm is then simulated using the MATLAB and the solution is compared to the optimum one. Three scenarios are used: 20, 100 and 1000 cities within the US border. It can be concluded that, although the Greedy Heuristic consumes more iterations to solve the TSP, its result is the closest to the optimum solution. The result is yet feasible with a reasonable number of iterations. On the other hand, the Genetic Algorithm fails to find the shortest path but yet finds an alternative with longer distance. This is not a surprising result, since the Genetic process uses permutations between cities to find the best route but those permutations are random therefore they offer no guarantee on the optimal path. Therefore, it can be concluded that, in large number of nodes, the Genetic algorithm does not converge to a valid solution.

V. FUTURE WORK

To optimize the shortest path of the tour, a combination between two algorithms can be used. for instance, since the Nearest Neighbor and Greedy algorithms uses nearly similar steps and finds nearly similar route lengths, they can be combined to find one optimum path. Another possible solution is using the Lin-Kernighan Algorithm, which is based on graph partitioning of the problem space and searching each part separately.

REFERENCES

[1] M. M. Flood, “The Traveling Salesman Problem,” Opns. Res., 1956.
[2] C. Nilsson., Tewfik, “Heuristics for the Traveling Salesman Problem,” Linkoping University, pp. 473-480, June 1996.
[3] D. Johnson, L. McGeoch, The Traveling Salesman Problem: A Case Study in Local Optimization.
[4] B. Kim, J. Shim, M. Zhang, Comparison of TSP Algorithms, December, 1998.
[5] Donald Davendra, “TRAVELING SALESMAN PROBLEM, THEORY AND APPLICATIONS.”
[6] Corman H. Thomas, Leiserson E. Charles, Rivest L. Ronald, Stein Clifford, “Introduction to Algorithms,” Second Edition McGrawHill Book Company .
[7] del Castillo M. Jose, “A heuristic for the traveling salesman problem based on a continuous approximation,” Transportation Research Part B33 (1999) 123-152 .
[8] Valenzuela I. Christine, Jones J. Antonia, “Estimating the Held-Karp lower bound for the geometric TSP ,” European Journal of Operational Research 102(1997) 157-175.
[9] Cesari Giovanni, “Divide and Conquer Strategies for Parallel TSP Heuristics,” Computers Operations Research , Vol.23, No.7, pp 681-694, 1996 .
[10] Joseph Kirk, MATLAB implementation of nearest neighbor and genetic algorithm for TSP.