# Heuristic Algorithm Strategies to Solve Travelling salesman Problem

[1]Sk. Mastan, Research Scholar, Department of Mathematics, **JNTUA**, Ananthapuramu.

masthanshaik119@gmail.com

[2]U. Balakrishna, Department of Mathematics, SITAMS,Chittoor.

prasanti_balu@rediffmail.com

[3]G. Sankar Sekhar Raju, Department of Mathematics, JNTUA COLLEGE OF ENGINEERING, Pulivendula.

rajugss@yahoo.com

**Abstract:** The term **heuristic** is used for algorithms which find solutions among all possible ones, but they do not guarantee that the best will be found, therefore they may be considered as approximately and not accurate algorithms. These algorithms, usually find a solution close to the best one and they find it fast and easily. Sometimes these algorithms can be accurate, that is they actually find the best solution, but the algorithm is still called heuristic until this best solution is proven to be the best. The method used from a heuristic algorithm is one of the known methods, such as greediness, but in order to be easy and fast the algorithm ignores or even suppresses some of the problem's demands.

Keywords: Heuristic algorithm, Spanning tree, DFTT.

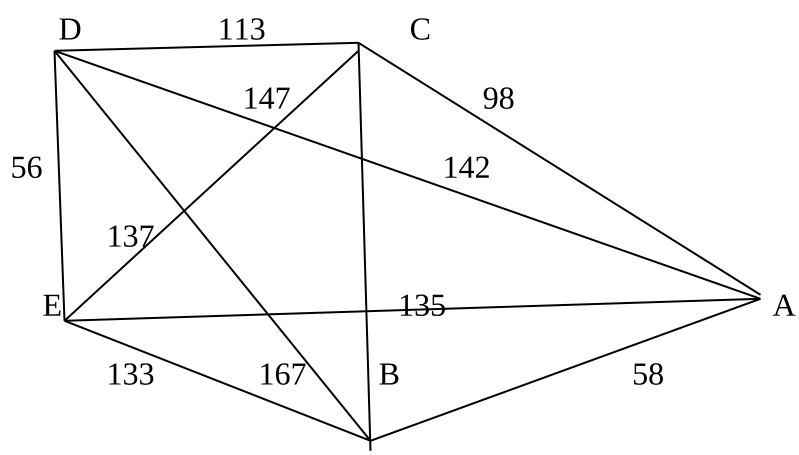
## I. INTRODUCTION

State as follows: A traveler S wants to visit each of n cities exactly once and return to its satisfies point with minimum distance/mileage for Heuristic algorithm for Travelling Sales man problem as follows: Example: Let's number the cities from 1 to ***n*** ,and let city 1 be the city-base of the salesman. Also let's assume that **c(i,j)** is the visiting cost from **i** to **j**. There can be **c(i,j)<>c(j,i).**Apparently all the possible solutions are ***(n-1)!***.Someone could probably determine them systematically, find the cost for each and everyone of these solutions and finally keep the one with the minimum cost. These requires at least ***(n-1)!*** steps. If for example there were 21 cities the steps required are ***(n-1)!=(21-1)!=20!*** steps. If every step required a ***msec*** we would need about **770 entries** of calculations. Apparently, the exhausting examination of all possible solutions is out of the question. Since we are not aware of any other quick algorithm that finds a best solution we will use a heuristic algorithm. According to this algorithm whenever the salesman is in town ***i*** he chooses as his next city, the city ***j*** for which the ***c(i,j)*** cost, is the minimum among all ***c(i,k)*** costs, where ***k*** are the pointers of the city the salesman has not visited yet. There is also a simple rule just in case more than one cities give the minimum cost, for example in such a case the city with the smaller ***k*** will be chosen. This is a greedy algorithm which selects in every step the cheapest visit and does not care whether this will lead to a wrong result or not.

**Input** : Number of cities n and array of costs c(i ,j) i, j=1,..n (We begin from city number 1)
**Output**:Vector of cities and total cost.

- (* starting values *)
- C=0
- cost=0
- visits=0
- e=1 (*e=pointer of the visited city)
- (* determination of round and cost)
- for r=1 to n-1 do
  - choose of pointer j with
  - minimum=c(e,j)=min{c(e,k);visits(k)=0 and k=1,..,n}
  - cost=cost+minimum
  - e=j
  - C(r)=j
- end r-loop
- C(n)=1
- cost=cost+c(e,1)

We can find situations in which the TSP algorithm don't give the best solution.We can also succeed on improving the algorithm.For example we can apply the algorithm t times for t different cities and keep the best round every time.We can also unbend the greeding in such a way to reduce the algorithm problem ,that is there is no room for choosing cheep sides at the end of algorithm because the cheapest sides have been exhausted.




Now to calculate five cities of Route and Total distance of rules hold:

| **Route** | **Total Distance ( Rules)** |
|---|---|
| **A-B-D-C-E-A** | **610** |
| **A-C-B-D-E-A** | **598** |
| **A-D-B-C-E-A** | **728** |
| **A-B-E-D-C-A** | **458** |

Minimal spanning tree (MST) for Heuristic Algorithm solving to the DFTT length is exactly twice the MST's weight MST weight is not more than length of optimal Tour. Skipping visities

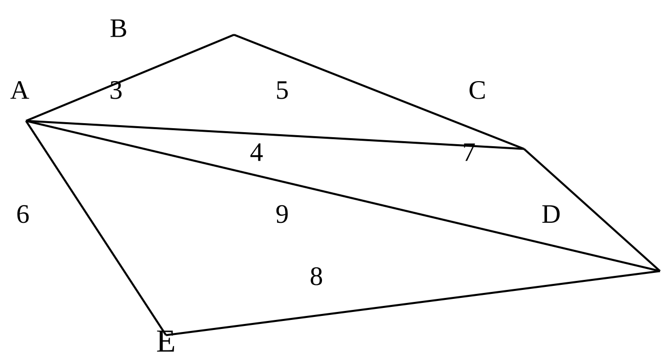
| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | - | 3 | 4 | 9 | 6 |
| B | 3 | - | 5 | 12 | 9 |
| D | 4 | 5 | - | 7 | 10 |
| E | 6 | 9 | 10 | 8 | - |

Nodes along the DFTT give a tour that is at most as long as the DFTT (Depth First Tree Tour) is triangular inequalities the tour length is at most twice the optimal length 15+10+25+30+25 = 105, 2 * 60 = 120.

We solve another spanning tree of Heuriastic alogithm for solving problem as hold:

The spanning tree is A-B-C-D-E-A is 29.

The matrix form and diagram as follows:



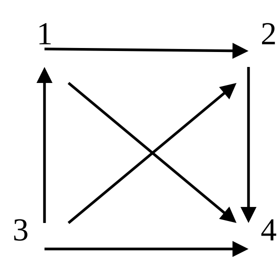

Algorithms are two types: Algorithms: Design, Domain knowledge, any language, H/W & OS, Analyze. Programe: Implementation, Programmer, Programme language H/W & OS and testing.

Characterization of Algorithmn: Input, output definiteness, finiteness and Effectiveness.

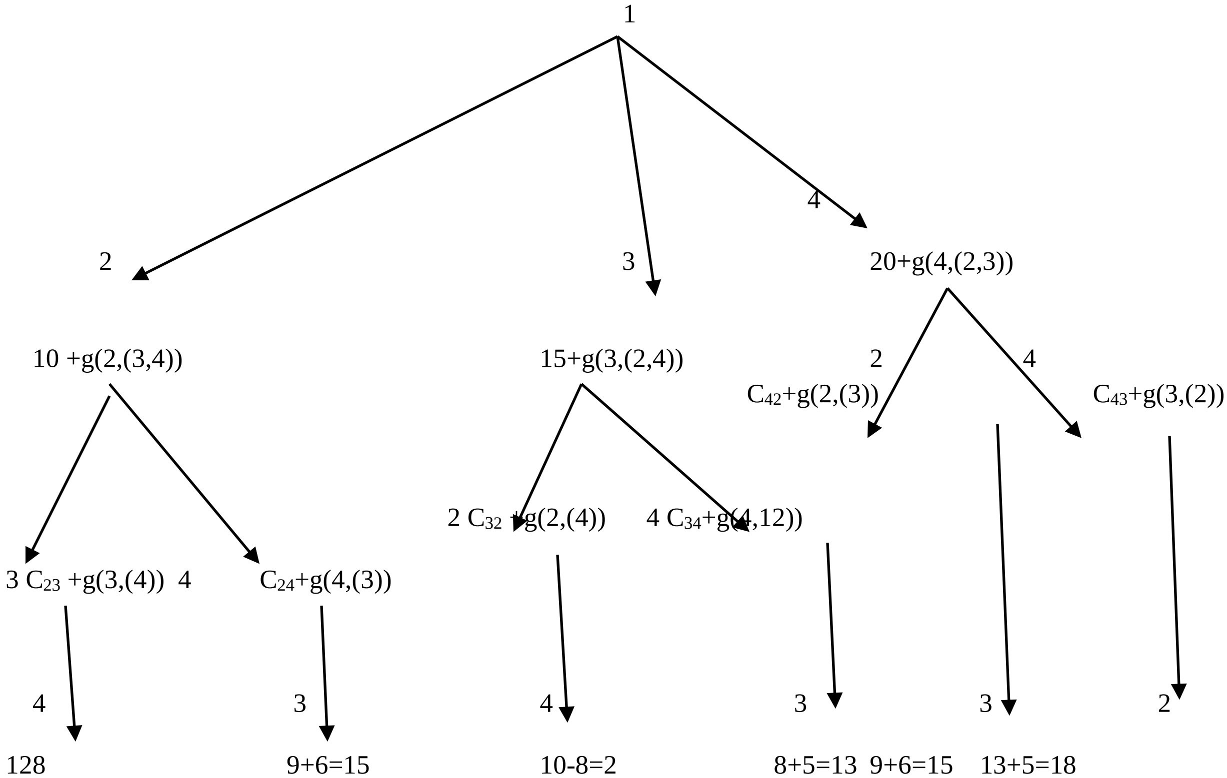
For solving Travelling Salesman Problem by Heuristic Algorithm example as hold:

$g(i,s) = \min_{k \in s}\{C_{ik} + g(k, S - \{k\}\}$ based on this formula to find different types states to be calculated.

$$\begin{pmatrix} 0 & 10 & 15 & 20 \\ 5 & 0 & 9 & 10 \\ 6 & 13 & 0 & 12 \\ 8 & 8 & 9 & 0 \end{pmatrix}$$



The shortest route of Heuriastic Algorithm is $g(1,\{2,3,4\}) = {}_{k\in(2,3,4)}^{min}\{C_{ik} + g(k,\{2,3,4\} - \{k\}$




Finally the above shortest all routes of Heuristic Algorithm results are hold:

$g(2,4) = 5; g(3,4) = 6; g(4,4) = 8.$

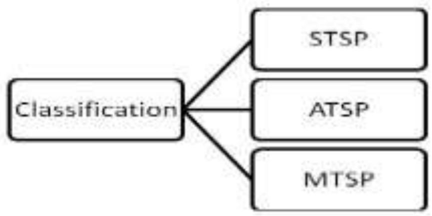
$g\big(2,(3)\big), g\big(2,(4)\big), g\big(3,(2)\big), g\big(3,(4)\big), g\big(4,(2)\big).$

$g\big(2,(3)\big) = 15, g\big(2,(4)\big) = 18, g\big(3,(2)\big) = 18, g\big(4,(2)\big) = 20, g\big(4,(2)\big) = 13, g\big(4,(3)\big) = 15$

$g\big(2,(3,4)\big) = 25; g\big(3,(2,4)\big) = 25; g\big(4,(2,3)\big) = 23 \;\; finally\; g\big(1,(2,3,4)\big) = 35$

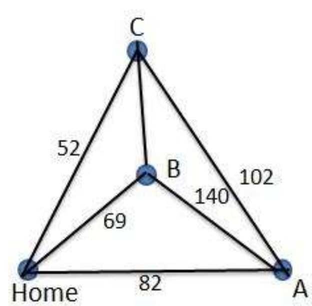
The shortest route of Heuristic Algorithm is 35.

The computational complexity theory makes it possible to validate the concepts of “easy” and “hard” problems and the distinction among them. Problems can be formally classified based on their complexity and if a problem strictly belongs to the family of NP-hard or complete problems, we know in advance that there is little chance of finding an efficient and exact algorithm to solve it. The algorithm for such a problem has an execution time bursting for increasing problem sizes, and in majority cases is not feasible for most practical purposes. Computers are playing every effective role in solving different complex problems but the matter of fact is that some problems are fundamentally harder to solve. Although, for some problems it is possible to develop intelligent algorithms that solve the problems expeditiously, however, it seems substantially hard even in some cases impossible to come up with any solution [24]. Davendra [13] defined TSP as, “Given a set of cities of different distances away from each other, and the objective of TSP is to find the shortest path for a salesperson to visit every city exactly once and return back to the origin city". TSP is an important applied problem with many fascinating variants; like theoretical mathematics, computer science, NP hard problem, combinatorial optimization and operation research [25]. TSP is classified as symmetric, asymmetric and multiple TSP based on the distance and direction between two cities in a graph (Figure 1). If distance between two cities is same in each direction it is symmetric with undirected nature otherwise it is asymmetric.




The core objective of TSP is to minimize the total traveling cost of object around tours. In order to understand TSP, let us explore the given below example. Figure 2 shows the road distance between the three towns i.e. ABC and additionally assume that a salesperson, whose business is to sells lubricant items to different companies located in these three cities, must travel (starting from home). Here the decimal values near the line edges in the diagram are the driving distances between the cities. In this example, we are assuming that we have a symmetric TSP i.e. the cost in going from A to B is the same as the cost in going from B to A but in asymmetric TSP the cost could not be the same between the two points.

In the given situation, one question arises in mind that how many TSP tours could be? The general answer to this question is, for the complete graph with n vertices, the number of different TSP routes would be Equation (1).




Let us calculate the cheapest tour by working on above Figure 2.

HABCH□82+140+90+52=364

HACBH□82+102+90+69=343

HCABH□52+102+140+69=363

Thus, going from H to A, then to C, then to B and then back H could be the best choice.

**. Related Work**

In literature TSP is used in two forms: i) combinatorial optimization version and ii) decision version. In first version it is used to find a minimum Hamiltonian cycle and in later version to check the existence of smaller graph.

Theoretical computer science and operations research, both fields of combinatorial optimization contains TSP. In this problem, set of cities are given with their distances to find the shortest route to each city without visiting a city twice. In 1930, it was first formulated as a mathematical model and applied to so many areas to find their optimal solutions e.g., Clustering of array of data [1], Handling of a warehouse materials [2] and crystal structure analysis [3]. Resource constrained scheduling problem with aggregate deadline also solved with TSP [4]. Researches [5, 6] took orienteering and prize collection problems as special cases of resource constrained TSP. One of the best known and more complex combinatorial problem is Vehicle routing problem, to determine the order of vehicle for customers serving from fleet of vehicles, is being solved by TSP [7].

**Discussions**

Previous section demonstrates the different methodologies to solve NP-hard TSP problem approximately. Although, different techniques had been devised previously, but, all available techniques are not efficient in terms of solution time and quality. Comparison by Maredia [10] shows that Nearest Neighbor heuristic works well but it is not sure that it will give us solutions good as brute force. Moreover, greedy approach is not a good approximation technique for TSP. Comparison of Branch and bound technique with brute force method is presented in Figure 3 (data extracted from Maredia [10]). SA algorithm obtains has ability to find the good quality final solutions by mechanism of gradually going from one solution to the next. The main difference of SA from the 2-opt is that the local optimization algorithm is often restrict their search for the optimal solution in a downhill direction which mean that the initial solution is changed only if it results in a decrease in the objective function value**.** However, it is shown by Kim et al. [11] that 2-opt algorithm works well when the problem size is less than 50 cities. For comparison of the approaches we are referring to data Kim et al. [11]. Data extracted from Kim et al. [11] is plotted in Figure 4 and 5, according to results it is obvious that the solution of TSP using the neural network outperforms than all other approaches for all problem sizes. However, if we compare the time consumed to solve the problems, it is easy to realize that neural network approach is taking much more time as compared to others. Genetic algorithm has been used for many optimization problems; however, the use of genetic algorithm for TSP has disadvantages of premature convergence and poor local search capability. These disadvantages can be overcome by adaptation of other efficient working algorithms e.g., artificial immune systems [13].

**Conclusion and Recommendations**

This paper provides the survey of different heuristics used to solve TSP. First, an example of TSP is presented to give the idea of TSP. This survey is limited to some selected approaches because it is not possible to cover all approaches here. So, only most relevant approaches are discussed here. Although a number of heuristics had been devised however; some are efficient with respect to time and some are outperforming in solution quality.

**References**


[1] Lawler EL, Lenstra JK, Rinnooy AHG, Shmoys DB. The Traveling Salesman Problem. Chichester: John Wiley. 1985.
[2] Ratliff HD, Rosenthal AS. *Order-Picking in a Rectangular Warehouse: A Solvable Case for the Traveling Salesman Problem*. PDRC Report Series No. 81-10. 1981.
[3] Bland RE, Shallcross DF. *Large Traveling Salesman Problem Arising from Experiments in X-ray Crystallography: a Preliminary Report on Computation.* Technical Report No. 730. 1987.
[4] Miller, Pekny J. Exact Solution of Large Asymmetric Traveling Salesman Problems. *Science 251*. 1991: 754-761.
[5] Balas E. The Prize Collecting Traveling Salesman Problem. *Networks 19*. 1989: 621-636.
[6] Golden BL, Levy L, Vohra R. The Orienteering Problem. *Naval Research Logistics 34*. 1987: 307-318.
[7] Christofides N. Vehicle Routing, in The Traveling Salesman Problem. In: Lawler, Lenstra, Rinooy Kan and Shmoys. *Editors*. John Wiley. 1985: 431-448.

[8] Rego C, Gamboa D, Glover F, Osterman C. Traveling salesman problem heuristics: leading methods, implementations and latest advances. *European Journal of Operational Research*. 2011; 211(3): 427-441.

[9] Cook W. History of the TSP. The Traveling Salesman Problem. Georgia Tech. 2009.
[10] Maredia A. History, Analysis, and Implementation of Traveling Salesman Problem (TSP) and Related Problems. Thesis. Department of Computer and Mathematical Sciences University of Houston-Downtown. 2010.
[11] Kim IB, Shim IJ, Zhang M. Comparison of TSP Algorithms, Project for Models in Facilities Planning and Materials Handling. 1998.
[12] http://www.math.uwaterloo.ca/tsp/history/milestone.html (assessed on 08-06-2015).
[13] Davendra D. Traveling Salesman Problem, Theory and Applications. INTECH open access publishers. 2010.
[14] Dantzig G, Fulkerson R, Johnson S. Solution of a large-scale traveling-salesman problem. *Operations Research 2*. 1954: 393-410.
[15] Held M, Karp RM. The traveling-salesman problem and minimum spanning trees: part II. *Mathematical Programming 1*. 1971: 6-25.

[16] Camerini PM, Fratta L, Maffioli F. On improving relaxation methods by modified gradient techniques. *Mathematical Programming Study 3*. 1975: 26-34.
[17] Grötschel M, Padberg M. On the symmetric traveling salesman problem: theory and computation. In: R. Henn, B. Korte, W Oettli. *Editors*. Lecture Notes in Economics and Mathematical Systems 157. Springer, Berlin. 1977: 105-115.
[18] Crowder H, Padberg MW. Solving large-scale symmetric travelling salesman problems to optimality. *Management Science 26*. 1980: 495-509.
[19] Padberg M, Rinaldi G. Optimization of a 532-city symmetric traveling salesman problem by branch and cut. *Operations Research Letters 6*. 1987: 1-7.
[20] Grötschel M, Holland O. A cutting plane algorithm for minimum perfect 2-matchings. *Computing 39*. 1987: 327-344.
[21] Applegate D, Bixby R, Chvátal V, Cook W. Finding cuts in the TSP (A preliminary report). *DIMACS Technical Report.* 1995: 95-05.
[22] Applegate D, Bixby R, Cook W, Chvátal V. On the solution of traveling salesman problems. Rheinische Friedrich-Wilhelms-Universität Bonn. 1998.
[23] Applegate D, Bixby R, Chvátal V, Cook W. The Traveling Salesman Problem: A Computational Study. Princeton, New Jersey, USA: Princeton University Press. 2006.

[24] Fatma A, Karkory A, A Abudalmola. Implementation of Heuristics for Solving Travelling Salesman Problem Using Nearest Neighbour and Minimum Spanning Tree Algorithms. *International Journal of Mathematical, Computational, Natural and Physical Engineering.* 2013; 7(10).
[25] http://en.wikipedia.org/wiki/Travelling_salesman_problem (assessed on 08-06-2015).
[26] Turban E. Decision support and expert system. Macmillan series in information system. 1990.
[27]