A Survey on Travelling Salesman Problem

Sanchit Goyal
Department of Computer Science
University of North Dakota
Grand Forks, North Dakota 58203
sanchitgoyal01@gmail.com

Abstract

The Travelling Salesman problem is one of the most popular problems from the NP set, it is also one of the hardest too. The solution to this problem enjoys wide applicability in a variety of practical fields. Thus it highly raises the need for an efficient solution for this NP Hard problem. However even the most common deterministic solutions to the problem are known to possess an exponential time complexity. There have been many efforts in the past to provide time efficient solutions for the problem, both exact and approximate. This paper surveys on some of these deterministic and non-deterministic efforts. It also presents and analyzes a greedy non−deterministic algorithm to solve the travelling salesman problem. The paper relates the Travelling Salesman problem with the hamiltonian circuit problem in order to demonstrate the existing property of reducibility between the two problems and the its advantages in providing a solution to the TSP.

1 Introduction

The Travelling Salesman Problem (TSP) is a problem in combinatorial optimization studied in both, operations research and theoretical computer science. Given a list of cities and their pair wise distances, the task is to find a shortest possible tour that visits each city exactly once. It was first formulated as a mathematical problem in 1930 and is one of the most intensively studied problems in optimization.

Problems having the TSP structure most commonly occur in the analysis of the structure of crystals [1], in material handling in a warehouse [2], the clustering of data arrays [3]. Related variations on the traveling salesman problem include the resource constrained traveling salesman problem which has applications in scheduling with an aggregate deadline [4]. The prize collecting traveling salesman problem [5] and the orienteering problem [6] are also special cases of the resource constrained TSP. Most importantly, the traveling salesman problem often comes up as a sub problem in more complex combinatorial problems, the best known and important one of which is the vehicle routing problem [7], that is, the problem of determining for a fleet of vehicles which customers should be served by each vehicle and in what order each vehicle should visit the customers assigned to it.

1.1 Versions of TSP

The TSP has been most commonly expressed in two forms. The combinatorial optimization version i.e. the problem of finding the minimum hamiltonian cycle in a graph of cities, and the decision version i.e. the problem of checking the existence of a hamiltonian cycle in a graph smaller than a given weight. In the theory of computational complexity, the combinatorial optimization version belongs to the NP Hard set of problems implying that there is no polynomial time algorithm even for checking the correctness of a given solution to the problem, whereas the decision version belongs to the class of NP complete problems implying that a solution can be checked in polynomial time. Though without a proof that P $\neq$NP, it can only be assumed that there is no efficient algorithm for solving any version of the TSP. In other words, it is likely that the worst case running time for any algorithm for TSP increases exponentially with the number of cities, so even some instances with only hundreds of cities will take many CPU years to solve exactly[15].

1.2 Classification of TSP

Different instances of the TSP are also divided into different classes based on the arrangement of distance between the cities or the type of graph in concern. In the Symmetric TSP, the distance between two cities is the same in each direction, forming an undirected graph. This symmetry halves the number of possible solutions. In the Asymmetric TSP, paths may not exist in both directions or the distances might be different, forming a directed graph.

2 Reducibility between TSP and Hamiltonian Circuits

In the absence of any efficient algorithms for the problems of NP set, the existing property of reducibility amongst them can be considered a slight relaxation for the design specialists. A travelling salesman's dilemma can be typically depicted as the problem of finding the shortest hamiltonian circuit in a graph. This can be easily shown by the fact that the path of a travelling salesman starts and ends at the same vertex and covers all the vertices in the graph exactly once, which is nothing but a hamiltonian cycle of a graph. Also since the salesman needs to travel minimum while covering all the vertices thus the corresponding hamiltonian Cycle is also of minimum size in the graph.

While designing efficient solutions to the real world situations based on NP problems, one can easily configure the algorithms to rule out specific cases from the search space based on the quick tests available for some of the closely related problems from the NP set. Worst case for a typical instance of the Travelling Salesman Problem, i.e. the absence of any path can be ruled out by initially checking for the existence of a hamiltonian circuit in the graph in concern.

A graph can be checked for existence of hamiltonian path easily by taking help of many existing theorems in the graph theory. Some of these conditions are shown. An undirected graph G with N nodes is hamiltonian i.e. a hamiltonian circuit exists in the graph, if either of the following holds true-

1. G is a complete graph
2. For n $\geq$ 3 vertices. If each vertex has degree at least n$/2$. $($Diracs Theorem$)$[16]
3. For n $\geq$ 3 if, for each pair of non-adjacent vertices, the sum of their degrees is n or greater $($Ore's Theorem$)$[17] and many more...

Such tests are quick fixes which tremendously improve the worst case performance of otherwise exponential time algorithms. Thus these techniques can prove extremely beneficial for designers when it is known that the input space for the problem mainly consists of worst cases of the TSP. Also it can be stated strongly that if we have an efficient way to either find out all the hamiltonian circuits in a graph or a minimum hamiltonian circuit in a graph, this implies there is an efficient way to solve for the travelling salesman problem too.

3 Naive Approach to Solution

A naive solution is a highly inefficient, generally brute force but an exact algorithm to solve the problem. The problem statement of the travelling salesman trivia can be depicted in a

number of alternate formats due to the inter-linkage between different NP problems, which in turn leads to a number of naive solutions to the problem.

Interpretation as the problem of finding the lightest hamiltonian circuit in the graph leads to one such algorithm. A naive approach can try finding all the hamiltonian circuits in the graph and choose the one with the smallest length. However finding hamiltonian Circuits in a graph itself is a NP complete problem.

For N nodes in a graph and one starting and ending node there are (n-1)! maximum possible hamiltonian cycles for a directed graph and thus an asymmetric TSP and (n-1)!/2 maximum possible hamiltonian cycles for an undirected graph and thus a symmetric TSP (since, order does not matter in its case). Therefore a naive algorithm which will not only find these circuits but also compare them is surely going to be O ($n!$) for assymeteric TSP and O ($n!/2$) for symmetric TSP in its worst case performance, which is extremely inefficient.

This naive approach considers all (n-1)! or (n-1)!/2 permutations of the vertices and eliminates the ones which are not viable hamiltonian circuits by comparing edges between each pair of vertices in the permutation being considered using an O(n) process. It also calculates the total weight of the permutation in the same process of verification and stores the values in O ((n-1)!) or O ((n-1)!/2) space depending on the type of TSP being considered. After it goes through all the possible cases of hamiltonian circuits, thus verifying and processing each, it finally finds out the hamiltonian cycle with the least weight by simply finding the lowest value in data structure used to store the total weights of the hamiltonian cycles. This process is typically O((n-1)!) or O((n-1)!/2) in worst case. Therefore the complete process can be easily calculated to be O(n.n!) for asymmetric TSP and O(n.n!/2) for symmetric TSP.

4 Solving the Travelling Salesman Problem

There have been many efforts to solve the travelling salesman problem ever since it was coined in 1930. The deterministic approaches strive to give an exact solution to the problem whereas the non deterministic try to provide a tour with a near minimal cost. This paper discusses few of such approaches in the following sub-section. A special case of a polynomial time non deterministic algorithm has also been proposed in this paper.

4.1 Deterministic Approaches to TSP

One of the earliest deterministic solutions to TSP was provided by Dantzig et al. [8], in which linear programming (LP) relaxation is used to solve the integer formulation by adding suitably chosen linear inequality to the list of constraints continuously. Held and Karp [10] presented a dynamic programming formulation for an exact solution of the trav-

elling salesman problem, however it has a very high space complexity, which makes it very inefficient for higher values of N[9].

The branch and bound technique based algorithm published in [14] was able to successfully increase the size of the problem solvable without using any problem specific methods. The algorithm branches into the set of all possible tours while calculating the lower bound on the length of the tour for each subset. Eventually it finds a single tour in a subset whose length is less than or equal to some lower bound for every tour. The algorithm however grows exponentially in time with the input, but it is able to calculate the TSP for 40 cities with appreciable average time consumption as displayed by the authors.

In the survey on exact algorithms for the Travelling Salesman problem, most attempts were found trying to address just a subset of the problem, instead of working on the complete problem space. This approach proved to be successful in almost all such cases, often slightly advantageous as far as time complexity is concerned. This happens due to the fact that not all problem instances are equally difficult. An efficient solution to the specific class of such instances can be used extensively by applications which deal mostly with these relatively easy instances only.

The algorithm depicted in [15] solves the Travelling Salesman problem for graphs with degree at most 3 in exponential time. It has a slightly better time complexity of $O(2^n/3)$ and a linear space complexity. However for a graph with n vertices and degree at most 4 this running time increases to O$((27/4+ \in)^{n/3}))$.

Other exact solutions to the problem as described in [9] include cutting plane method (introduced by Dantzig, Fulkerson, and Johnson), branch-and-cut (introduced by Padberg and Rinaldi), branch-and-bound (Land and Doig) and Concorde (introduced by Applegate, Bixby, Chatal, and Cook) . Although each of these algorithms become highly inefficient as we increase the number of cities in the tour.

4.2 Non-Deterministic Solution to TSP

The exact solutions provide an optimal tour for TSP for every instance of the problem; however their inefficiency makes it unfeasible to use those solutions in practical applications. Therefore Non$-$Deterministic solution approach is more useful for the applications which prefer time of run of the algorithm over the accuracy of the result. There has been a vast research in past to solve the TSP for an approximate result. Some of the implemented approximate algorithms as described in [9] are being listed here

4.2.1 Nearest Neighbour Algorithm

It follows a very simple greedy procedure: The algorithm starts with a tour containing a randomly chosen city and then always adds to the last city in the tour the nearest not yet visited city. The algorithm stops when all cities are on the tour.

4.2.2 Insertion Algorithms

All insertion algorithms start with a tour consisting of an arbitrary city and then choose in each step a city k not yet on the tour. This city is inserted into the existing tour between two consecutive cities i and j, such that the insertion cost (i.e., the increase in the tour's length) d(i; k) + d(k; j) = d(i; j) is minimized. The algorithms stop when all cities are on the tour.

4.2.3 K-Opt Heuristics

The idea is to define a neighborhood structure on the set of all admissible tours. Typically, a tour t is a neighbor of another tour t if t can be obtained from t by deleting k edges and replacing them by a set of different feasible edges (a k-Opt move). In such a structure, the tour can iteratively be improved by always moving from one tour to its best neighbor till no further improvement is possible. The resulting tour represents a local optimum which is called k-optimal

Researchers have also taken help from literature on artificial intelligence and machine learning. Various algorithms optimizing the power of neural networks have been proposed for the approximation of the optimal cycle. Also a popular method is the Ant Colony optimization scheme.

5 Greedy Non−Deterministic Solution to TSP

A polynomial time non deterministic approach is being proposed here to solve the TSP in polynomial time. Although like other common greedy approaches, this approach too does not work as desired for some instances of the problem. However it halts in polynomial time for every instance and it provides an exact solution to the problem instances it works for.

The algorithm combines the use of selective edge elimination from the graph and Warshalls algorithm to find the minimum hamiltonian cycle in a graph. Warshalls algorithm is able to determine the path matrix for a graph in n^3 time from the connectivity matrix. Thus it can determine if there is a path between any two vertices of the graph at any instant in polynomial time.

```
procedure Warshall ()
 for(i = 0;i <max; i++)
  for(j = 0;j < max; j++)
   if(path[i][j] == 1)
    for(k = 0; k < max; k++)
      if(path[j][k] == 1)
        path[i][k] = 1;
```

Figure 1: Warshall Algortithm's Pseudocode

The algorithm starts from the starting node but works on the edges instead of the vertices. It picks up the largest unvisited edge repeatedly in the graph and removes it while taking care that the edge deletion should not make either of the 2 vertices adjacent due to that edge, disconnected from any of the other vertices in the graph. That is at every time step or every edge deletion there should be a path from every node in the graph to every other node. This condition is checked by running the Warshalls algorithm on the modified connectivity matrix of the graph at every time step. The algorithm halts after it has traversed all the edges in the original input graph. Since A graph with n nodes can have a maximum number of n^2 edges only, therefore the algorithm works in polynomial time $(O(n^5))$.

Existence of a Hamiltonian Cycle in a graph means that atleast a single path exists in the graph which connects all the vertices to each other and it will continue to exist even after the halting of the algorithm described. By executing this algorithm on the set of edges we are greedily removing the extra edges from the graph except the minimal cycle.

However, like any other greedy approach this algorithm also fails to deliver a solution for some problem instances. The algorithm halts with a minimum spanning tree of a graph instead of the Hamiltonian Cycle in a few cases. A major reason why greedy approaches although extremely efficient, do not work as required for some problem instances is due to the fact that in a greedy approach we always work on local minima while hoping to achieve the global minima by generalization. This approach fails terribly when there is a compromise required within the local minimas to achieve a better global minima. Same phenomenon occurs in the execution of the proposed algorithm. Since here we are removing the heavier edges without considering there impact on possible Minimal Hamiltonian Cycle, hence assuming that such a cycle consists of only the minimum weighted edges of all the edges possible edges between two adjacent vertices. Thus the proposed algorithm will not work for cases where a compromise on the weight of the edge between 2 vertices is required to achieve an overall minimum Hamiltonian Cycle Such a problem instance is shown in Figure 2.

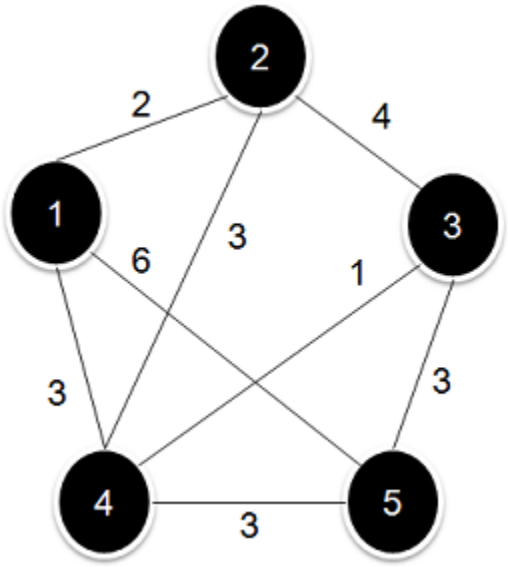

Figure 2: Example case when the proposed algorithm comes with a spanning tree instead of Hamiltonian Cycle.

6 Fastest known Solutions to TSP

6.1 Deterministic

The Concorde method[11] proposed by Applegate D, Bixby RE, Chvatal V, Cook W is widely regarded as the fastest TSP solver, for large instances, currently in existence. In 2001, it even won a 5000 Guilder prize from CMG for solving a vehicle routing problem the company had posed in 1996. It holds a record of solving a TSP instance having 15,112 nodes[11]. However it has the capability of solving only symmeteric versions of the problem[9].

6.2 Non−Deterministic

The Lin-Kernighan heuristic [12] is generally considered to be one of the most effective methods for generating optimal or near-optimal solutions for the symmetric traveling salesman problem (TSP). The heuristic grows by a relation of n^2 with time as shown in his paper [12] by W. Kernighan. However, the design and implementation of an algorithm based on this heuristic is not trivial. There are many design and implementation decisions to be made, and most decisions have a great influence on performance [13].

7 Problem Space Analysis

Due to the nature of TSP, most common solutions to the problem were found to run feasibly only for a graph with small number of nodes. Not much research was encountered in the survey over problem space analysis of the Travelling Salesman problem. However common approaches have been found to posses a critical region roughly around n = 40[14], afterwhich they start taking increasingly even more time to halt. This happens due to their non efficient time growth functions with respect to the number of nodes. Concorde a very well known fast and exact solution, however holds a record to have solved for 15,112 cities as well[11].

References

[1] R. E. Bland and D. F. Shallcross (1987). "Large Traveling Salesman Problem Arising from Experiments in X-ray Crystallography: a Preliminary Report on Computation," Technical Report No. 730, School of OR/IE, Cornell University, Ithaca, New York.

[2] H.D. Ratliff and A.S. Rosenthal (1981). "Order-Picking in a Rectangular Warehouse: A Solvable Case for the Traveling Salesman Problem," PDRC Report Series No. 81-10. Georgia Institute of Technology, Atlanta, Georgia.

[3] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys, eds. (1985). The Traveling Salesman Problem, John Wiley, Chichester.

[4] . Miller and J. Pekny (1991). "Exact Solution of Large Asymmetric Traveling Salesman Problems," Science 251, 754-761.

[5] E. Balas (1989). "The Prize Collecting Traveling Salesman Problem," Networks 19, 621-636

[6] B.L. Golden, L. Levy and R. Vohra (1987). "The Orienteering Problem," Naval Research Logistics 34, 307-318.

[7] N. Christofides (1985). "Vehicle Routing," in The Traveling Salesman Problem, Lawler, Lenstra, Rinooy Kan and Shmoys, eds., John Wiley, 431-448

[8] Dantzig GB, Fulkerson DR, Johnson SM (1954). Solution of a Large-scale Traveling Salesman Problem." Operations Research, 2, 393410.

[9] Hahsler, Michael; Hornik, Kurt (2007), "TSP Infrastructure for the Traveling Salesperson Problem

[10] Held M, Karp RM (1962). A Dynamic Programming Approach to Sequencing Problems." Journal of SIAM, 10, 196 - 210.

[11] Applegate D, Bixby RE, Chvatal V, Cook W (2000). TSP Cuts Which Do Not Conform to the Template Paradigm." In M Junger, D Naddef (eds.), Computational Combinatorial Optimization, Optimal or Provably Near-Optimal Solutions," volume 2241 of Lecture Notes In Computer Science, pp. 261{304. Springer-Verlag, London, UK.}

[12] S. Lin, B.W. Kernighan, An effective heuristic algorithm for the traveling salesman problem, Operations Research 21 (1973) ,498-516.

[13] K. Helsgaun, "An effective implementation of the lin-kernighan traveling salesman heuristic," European Journal of Operational Research, vol. 126, no. 1, pp. 106-130, 2000.

[14] Little JDC, Murty KG, Sweeney WD, Karel C, "An Algorithm for the Traveling Salesman Problem," Operational Research, vol. 11, no. 6, pp. 972-989, 1963.

[15] Eppstein, D. 2003b. The traveling salesman problem for cubic graphs. In Proceedings of the 8th Workshop on Algorithms and Data Structures. Lecture Notes in Computer Science, vol. 2748. Springer-Verlag, New York, 307–318.

[16] Dirac's Theorum, Wikipedia, [Online], Available: http://en.wikipedia.org/wiki/Dirac

[17] Ore's Theorum, Wikipedia, [Online], Available: http://en.wikipedia.org/wiki/Ore