International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No.03, February 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

Implementing Tabu Search on Traveling Salesman Problem

E.Gangadevi

Department of Computer Science, Loyola College, Chennai, Tamil Nadu, (India)

ABSTRACT

The Travelling Salesman Problem (TSP) is a classic algorithm problem which focuses on optimization. This problem is solved using various search algorithms. This paper proposes to solve the TSP using Tabu search. Tabu search is one of the most widely used metaheuristic algorithms to solve TSP. It guarantees to give a near optimal solution to TSP. The main objective of this paper is to solve the TSP with Tabu search which reduces to return to the recently visited areas of the search space, which is called as a cycling which results in using a short term of memory space.

Keywords*: *Optimization, Adjacency Matrix, Adjacency List, Tabu list, Shortest Path.

I. INTRODUCTION TO TRAVELLING SALESMAN PROBLEM

Traveling Salesman Problem can be explained as a NP complete and a combinatorial optimization problem where from the starting node, it should visit every other node only once to cover a minimum distance. The travelling salesman problem was defined by the Irish mathematician W. R. Hamilton and by the British mathematician Thomas Kirkman. The major objective of TSP is to find the shortest tour through a set of N vertices so that each vertex is visited exactly once. In our problem, it can be stated as a network with N_i number of vertices, with n1 as a source and travel time or cost with ordered vertex pairs [1]. There are so many applications of TSP, which includes vehicle routing, job sequencing, combinatorial data analysis and so on.

II.TSP ALGORITHM

The exact algorithms are designed to find the optimal solution to the TSP, that is, the tour of minimal length. They are computationally expensive because they must (implicitly) consider all solutions in order to identify the optimum[2]. These exact algorithms are typically derived from the integer linear programming (ILP) formulation of the TSP.

Minimize

$$\sum_i \sum_j d_{ij} \; x_{ij}$$

Subject to

$$\sum_j x_{ij} = 1, \qquad i = 1 \ldots, N$$

$$\sum_i x_{ij} = 1, \quad j = 1, \ldots, N$$

$$(x_{ij}) \in X,$$

$$x_{ij} = 0 \text{ or } 1,$$

Where N is the number of nodes, d_{ij} is the distance between the nodes i and j, and the x_{ij}'s are the decision variables.x_{ij} is set to 1 when i and j are included in the tour, and 0 otherwise [3].

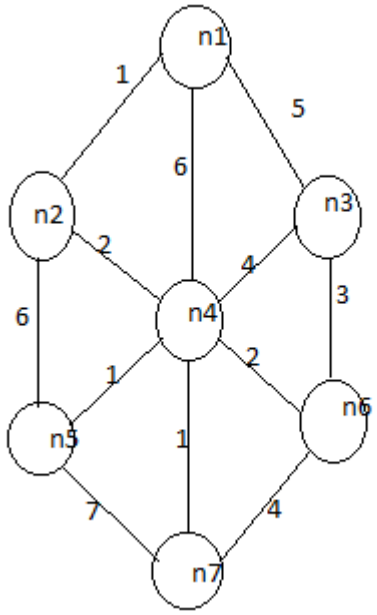

Fig.1 Weighted Graph

Also, we represent a weighted graph, adjacency matrix and adjacency list to solve our problem. The path matrix and the distance matrix are usually represented by the following equations.

$$Pij = \begin{cases} 1 \text{ if there is a path between } i \text{ and } j \\ 0 \qquad\qquad\qquad\qquad \text{otherwise} \end{cases} \qquad .1$$

International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No.03, February 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

$dij = (wij\ edge\ weight\ between\ i\ and\ j)$ (□.

III. ADJACENCY MATRIX

One of the simplest ways to represent a graph is to use a two-dimensional matrix. Here, each of the rows and columns represent a vertex in the graph. The value which is stored in the cell at the intersection of row and column defines that there is an edge or a path from one vertex to the other vertex [4].

Every adjacent vertices are connected by an edge. For the above example graph from Fig.1, we have presented the Adjacency matrix [5]. Each value in the matrix denotes the weight of an edge from one vertex to the other vertex.

$$\begin{bmatrix} 0 & 1 & 5 & 6 & 0 & 0 & 0 \\ 1 & 0 & 0 & 2 & 6 & 0 & 0 \\ 5 & 0 & 0 & 4 & 0 & 3 & 0 \\ 0 & 2 & 4 & 0 & 1 & 2 & 1 \\ 0 & 6 & 0 & 1 & 0 & 0 & 7 \\ 0 & 0 & 3 & 2 & 0 & 0 & 4 \\ 0 & 0 & 0 & 1 & 7 & 4 & 0 \end{bmatrix}$$

IV.ADJACENCY LIST

Adjacency list is a more space efficient way to implement a connected graph. It is efficient in terms of storing the values for the edges [6]. When we have a more number of vertices and edges, this will help to save more space. If the source is L1 then the adjacency list for l1 = {L2, L3, L4} and the representation is shown in Fig.2.

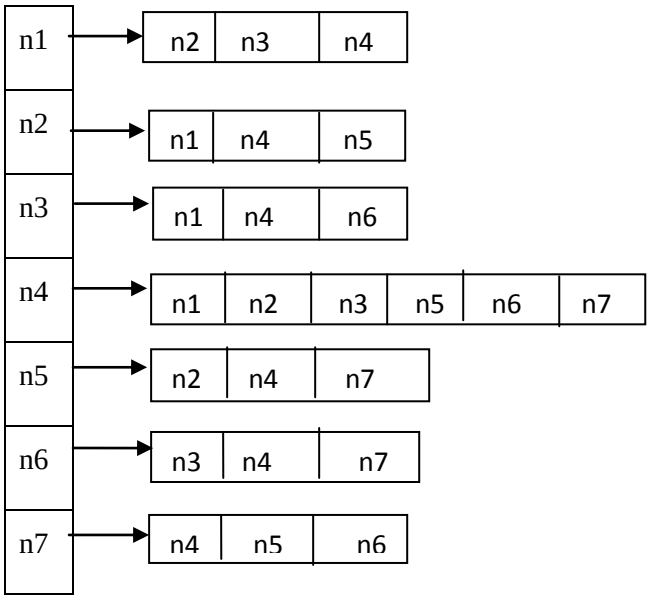

Fig. 2 Adjacency List

V.TABU SEARCH

Tabu Search is a Global Optimization algorithm and a Metaheuristic for controlling an embedded heuristic technique [7]. Tabu Search algorithm is to force an embedded heuristic from returning to recently visited areas of the search space, which is called as cycling. The plan of this approach is to maintain a temporary memory for all the changes of recent moves within the search space and preventing future moves from undoing those changes. In addition to this, an intermediate memory will be used for other search spaces [8]. Tabu search is based on local search. The search starts with an initial solution for the problem. That initial solution is taken as a current solution and searches for the best solution i.e. a group of tours that can be simply reached from the current solution. After this step, it takes the best solution from the neighborhood as the current solution and the search process continues.

Tabu search is terminated when it meets the maximum iteration count conditions, solution quality objectives or involving in the execution time [9]. Tabu search maintains a list of neighbor generations moves and ignores the solution that can be reached using tabu moves while searching the neighborhood of the solution. When a move is entered in to the list of tabu moves, it waits there for a pre-specified number of tabu search iterations called as the tabu tenure of the move. The list of tabu move changes continuously at the search execution and thus tabu search becomes an adaptive memory search algorithm [10]. The applications of Tabu search are Employee scheduling, Graph coloring, Graph partitioning, Non-linear covering, Network topology design, Computer channel balancing, Flow shop sequencing etc.

VI.TABU SEARCH FOR TSP

This paper proposes to apply Tabu search to the Traveling Salesman Problem. Some of the steps used by tabu for solving TSP are mentioned as follows:

1. Representing the solution:

A feasible solution is represented as a sequence of cities or referred as nodes or vertices, each will be visited only once .Always, the first and the last visited nodes are fixed to 1[11].The starting node will not be specified at this stage and by default it is 1.In our case, the solution representation will have the values as,

n2→n5→n4→n7→n6→n3→n1

2. Initial Solution:

This step focuses on finding the nearest node starting from the first node. It also finds the nearest unvisited node from the current node until all the nodes are visited.

3. Neighborhood:

At each visit, the neighborhood with the best objective value is selected. The fitness function will be decided here. The exchange of node also takes place at this stage.

4. Tabu List:

This step prevents the process from cycling in a small set of solutions and the tabu list stores the recently visited nodes. The tabu structure stores the number of iterations for which a given pair of nodes are restricted from swapping [12].

5. Aspiration criterion:

Sometimes, when there is no danger in cycling also, the tabu search will restrict the moves. So, at these situations, the tabu search process needs to be revoked [13].

6. Diversification:

The searching process might get trapped in a space of local optimum at some situations. Those processes should be allowed to search other parts of the solution [14].

VII. TEST RESULT

Finally, by using this tabu search method, the above weighted graph is tested and an optimal solution is found. In this graph, the source and the destination is n1. The main objective of this paper is the salesman should visit all the nodes exactly once, starting from the source node and ends his tour at the destination at a minimum cost [15]. This condition is completely satisfied by the tabu search.

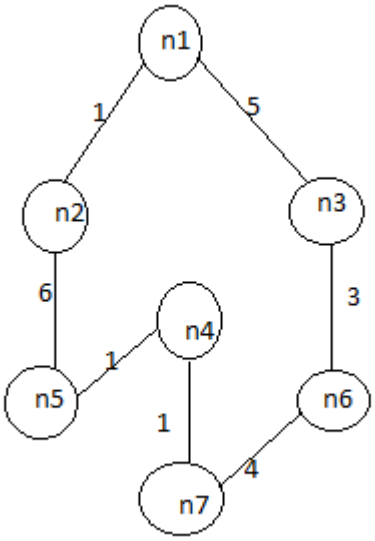

Fig 3. Test Result

It is usually obtained by altering one solution to get the next (better solution) according to some neighborhood structure [16]. Likewise, TS has a faster execution speed than other local search procedures because it does not

revisit already seen solutions, considering them tabu. This is possible because each move is recorded to avoid revisiting already seen [17].

Some of them have optimal solutions, while another's have the near to optimal solutions. There are different heuristics methods which are used to explore the solution space for TSP [18].

VIII. CONCLUSION

This paper proposed a new idea of implementing Tabu search algorithm to solve Traveling Salesman Problem. Although, there are various optimization algorithms emerging, each algorithm shows different set of solutions. Since, Tabu search has as advantage of taking cycling, this methods saves time and memory space. As a future enhancement, the same Traveling Salesman can be solved with more routes or nodes to travel and can add other nature inspired algorithms with tabu, to make the result more effective.

REFERENCES

[1] Sumanta Basu (2012)," Tabu Search Implementation on Traveling Salesman Problem and Its Variations: A Literature Survey", American Journal of Operations Research, 2012, 2, 163-173.

[2] Michel Gendreau a3b, Gilbert Laporte (1998)," A tabu search heuristic for the undirected selective travelling salesman problem", European Journal of Operational Research 106-539-545.

[3] Naveen Kumar,Karambir,Rajiv Kumar(2012). "A Study of Genetic Algorithm to solve TSP". ISSN-2229-371X.

[4] J.Holland,Adaptation in Natural and Artificial Systems,University of Michigan Press,Ann Arbor,1975.

[5] Varshika Dwivedi,Taruna Chauhan,Sanu saxena(2012)."Travelling Salesman Problem using Genetic Algorith".International Journal of Computer Applications(IJCA).

[6] Oloruntoyin Sefiu Taiwo (2013)." Application of Genetic Algorithm to solve TSP",International Journal of Advanced Research (IJOR), ISSN 2320-9194

[7] Saloni, G., & Poonam, P. (2013). Solving travelling Salesman Problem Using Genetic Algorithm. International Journal of Advanced Research in Computer Science and Software Engineering, 3, 376-380.

[8] Fred, G., James, K., & Manuel, L. (1995). Genetic Algorithms and Tabu Search: Hybrids for optimization. American Journal of Operations Research, 22, 111- 134

[9] Bajeh, A. & Abolarinwa, K. (2011). Optimization: A Comparative Study of Genetic and Tabu Search Algorithms. International Journal of Computer Applications, 31, 43-48.

[10] Guohui, Z., Liang, G., & Yang, S. (2010). A genetic Algorithm and Tabu Search for Multi Objective Flexible Job Shop Scheduling Problems. International Conference on Computing, Control and Industrial Engineering, 1, 251-254.

[11] David, L., Robert, E., Vasek, C., & William J. (2006). The Traveling Salesman Problem: A Computational Study. Princeton University Press.

[12] Arananayakgi, A. (2014). Reduce Total Distance and Time Using Genetic Algorithm. International Journal of Computer Science and Engineering Technology, 5, 815-819

[13] Otman, A., & Jaafar, A. (2010). A comparative Study of Adaptive Crossover Operators for Genetic Algorithms to Resolve the Traveling Salesman Problem. International Journal of Computer Applications, 31, 49-57.

[14] Alexandar, S. (2005). On the history of Combinatorial Optimization (Till 1960). American Journal of Mathematics, 12, 1-68.

[15] Zar, H., May, K. (2011). An Ant Colony Optimization Algorithm for Solving Traveling Salesman Problem. International Conference on Information Communication and Management, 16, 54-59

[16] Thamilselvan, R., & Balasubramanie, P. (2009). Integrating Genetic Algorithm, Tabu Search Approach for Job Shop Scheduling, International Journal of Computer Science and Information Security, 2, 42-53.

[17] Fiechter, C. (1994). A Parallel Tabu Search Algorithm for Large Travelling Salesman Problem. International Journal of Discrete Applied Mathematics, 51, 243-267

[18] Lionardo, Z. (2006). The Traveling Salesman Problem: A Comprehensive Survey. European Journal of Operational Research, 59, 231-247.