ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 1, January 2016

A Survey of Travelling Salesman Problem Using Heuristic Search Techniques

N.Sathya, Dr.A.Muthukumaravel

Research Scholar (Ph.D), Bharathiar University, Coimbatore, Tamilnadu, India

Department of MCA, Bharath University, Chennai, Tamilnadu, India

ABSTRACT: Combinatorial optimization is widely applied technique in multiple domains. It is also important for user to be cognizant that many combinatorial optimization problems would not yield realistic solution in practice. It might be costlier and unworthy to spend on combinatorial optimization solutions. Heuristic Search Techniques would help in such cases to obtain a near-optimal solution within reasonable time and better accurate. This paper will focus on the determining the Travelling Salesman Problem and applying the Heuristic Search Techniques. It is often necessary to compromise the requirements of mobility to construct a control structure which are no longer guarantee the right answer. This paper also will demonstrate the approach of solving the optimization problem by Depth-first search, Breadth-first search and Best-first search.

KEYWORDS: *Heuristic search,BFS,DFS,TSP.*

I. INTRODUCTION

The Travelling Salesman Problem (TSP) is one of the most famous combinatorial problems of all time. A salesman visits N cities with given positions and returns finally to his city of origin. Each city is to be visited only once, and the problem is to find the shortest possible route. In the field of the Travelling Salesman Problem (TSP), there are three main versions of TSP studied. The type of problem depends on how the input function is given Euclidian symmetric or asymmetric, or random distance matrixes. The term Euclidian comes from the representation of each city and the distance of the edges between them. Travelling Salesman Problem has been one of the most interesting and challenging problem in the literature.

II. LITERATURE REVIEW

The origins of the TSP are unclear. A hand book for Travelling Salesman from 1832 mentions the problem and includes example tours through Germany and Switzerland, but contains no mathematical treatment. The TSP was first defined by the Irish Mathematician HAMILTON in 1800. Vienna and Harvard was designed the general form of the TSP in 1930s Hassler Whitney at Princeton University introduced the name Travelling Salesman Problem soon after.

In USA and Europe, at 1950s and 1960s, the problem became increasingly very popular in scientific circles. These scientists and mathematicians George Dantzig, Delbert Ray Fulkerson and Selmer M. Johnson were expressed the problem of Integer Linear Problem (ILP) at the RAND Corporation in Santa Monica and they developed the cutting plane method for its solution. With these new methods they solved an instance with 49 cities to optimality by constructing a tour and proving that no other tour could be shorter.

III. VIEW OF TODAY TRAVELLING SALESMAN PROBLEM

Today, because of the ever-expanding computer capabilities in speed and memory, problems with less than 100 cities can be solved to optimality within reasonable time. These problems can of course not be solved with just brute force enumeration and distance checking. For the 16-city Travelling Salesman Problem, the problem of Homer's

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 1, January 2016

Ulysses attempting to visit the cities described in The Odyssey exactly once, there are 653,837,184,000 distinct routes. Enumerating all such roundtrips to find the shortest one, took 92 hours on a powerful workstation. By applying test criterions on the fitness of solutions, most routes do not have to be checked, and provide us with a tool to solve Travelling Salesman Problems directly. But unless P=NP proves to hold, a question which by now looks more likely to be P NP

IV. TRAVELLING SALESMAN PROBLEM

A salesman has a list of cities, each of which he must visit exactly once. There are direct roads between each pair of cities on the list. Find the route the salesman should follow for the shortest possible round trip that both starts and finishes at any one of the cities.

Definition

The general problem can be stated as the following: If we are given a set of places {n1, n2, … n N} and for each pair {ni, nj} of distinct cities the distance between them is d(ni, nj). If a line is drawn trough all the places visiting each place only once, and end up where it started. The goal is to find an ordering of the place that minimizes the total length of that line. This creates a Hamiltonian cycle in the graph of places Stated more formal:Using the same notation, the mathematical expression is to minimize:

$$\sum_{I=1}^{N-1} d\left(V_{\alpha(i)}, V_{\alpha(i+1)}\right) + d\left(V_{\alpha(N)}, v_{\alpha(1)}\right)$$

V. HEURISTIC SEARCH TECHNIQUES

A heuristic is a technique that improves the efficiency of a search process, possibly by sacrificing claims of completeness. Heuristics are like tour guides. A heuristic search function is a function that maps from problem state descriptions to measures of desirability, usually represented as numbers. Example of some simple heuristic search functions, Chess, Travelling Salesman |Problem, Tic-Tac-Toe

VI. BREADTH- FIRST SEARCH

In Breadth- First Search (BFS) we start at a vertex v and mark it as having been visited. The vertex v is at this time said to be unexplored. A vertex v is at this time said to be unexplored. A vertex is said to have been explored by an algorithm when the algorithm has visited all vertices adjacent from it.

(i) Algorithm for Breadth First Search

Algorithm BFS(*v*)

```
// A breadth first search of G is carried out beginning
// at vertex v. For any node i, visited [ i] = 1 if i has
// already been visited. The graph G and array visited[]
// are global; visited [] is initialized to zero.
{
    u := v; // q is a queue of unexplored vertices.
    visited[v] := 1;
   repeat
    {
            for all vertices w adjacent from u do
            {
                    if (visited [w] = 0) then
                    {
                            Add w to q; // w is unexplored.
```

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

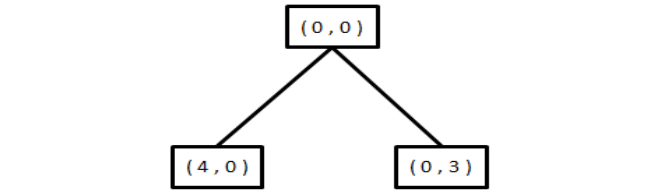
International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

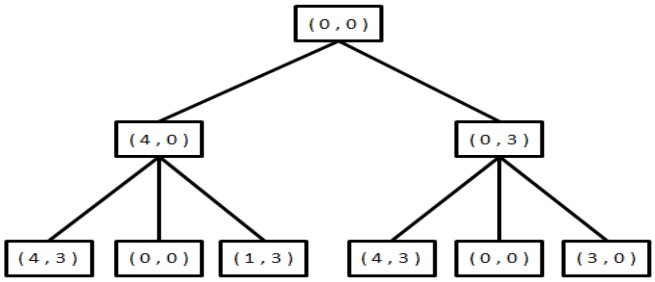
Vol. 4, Issue 1, January 2016

```
                    Visited[w] := 1;
                    }
                }
                if q is empty then return; // No unexplored vertex.
                Delete u from q; // Get first unexplored vertex.
        } until (false);
}
```

Example



One level of a Breadth First Search Tree



Two levels of a Breadth First Search Tree

(ii) Algorithm of above Breadth-first Search

1. Create a variable called NODE- LIST and set it to the initial state.
2. Until a goal state is found or NODE- LIST is empty do:
   (a) Remove the first element from NODE-LIST and call it E. If NODE-LIST was empty, quit.
   (b) For each way that each rule can match the state describe in E do:
      i. Apply the rule to generate a new state.
      ii. If the new state is a goal state, quit and return this state.
      iii. Otherwise, add the new state to the end of NODE-LIST.

VII. ADVANTAGES OF BREADTH FIRST SEARCH

- Breadth first search will not get trapped exploring a blind alley. This contrasts with depth first searching , which may follow a single, unfruitful path for a very long time, perhaps forever, before the path actually terminates in a state that has no successors. This is a particular problem in depth first search if there are loops unless special care is expended to test for such a situation.
- If there is a solution then breath first search is guaranteed to find it. Furthermore if there are multiple solutions then a minimal solution will found. This is guaranteed by the fact that longer paths are never explored until all shorter ones have already been examined.

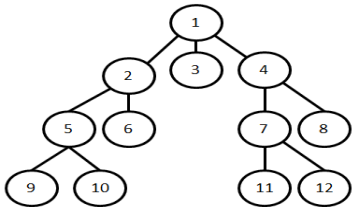
ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

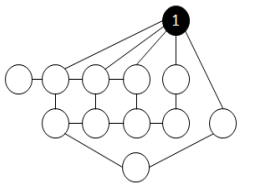
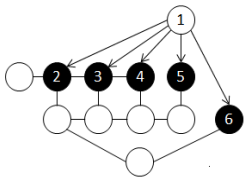
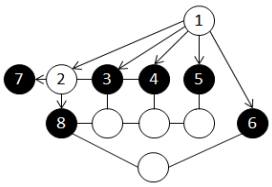
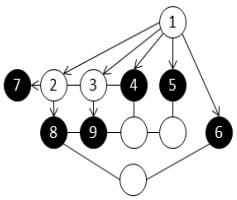
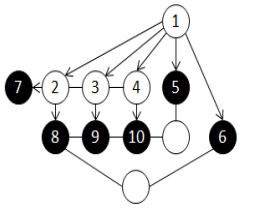
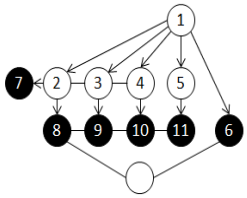
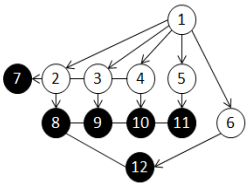
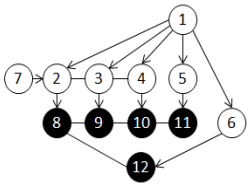
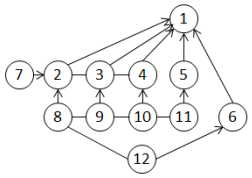
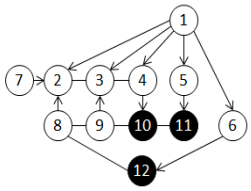
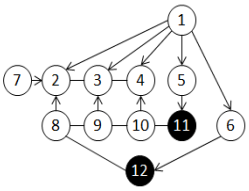
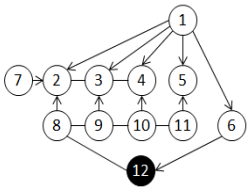
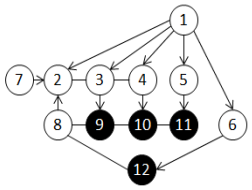
Vol. 4, Issue 1, January 2016

(i)Example of Breadth- First Search



Breadth - First Search

(ii)Graph Traversal of Breath First Search

Z

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 1, January 2016

VIII. DEPTH – FIRST SEARCH

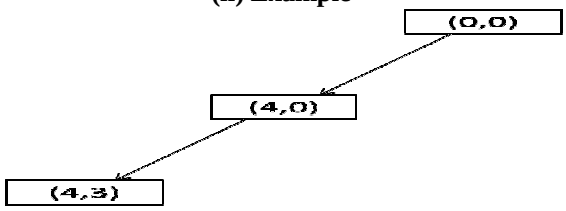
A depth first search of a graph differs from a BFS in that the exploration of a vertex v is suspended as soon as a new vertex is reached. At this time the exploration of the new vertex u begins. When this new vertex has been explored, the exploration of v contains. The search terminates when all reached vertices have been fully explored.

(i). ALGORITHM OF DFS

Algorithm DFS(*v*)

```
//Given an undirected (directed) graph G = (V,E) with
// n vertices and an array visited[] initially set
// to zero, this algorithm visits all vertices
// reachable from v. G and visited [] are global
{
        visited[v] := 1;
        for each vertex w adjacent from v do
        {
                if (visited[w] = 0) then DFS(w);
        }
}
```

(ii) Example



IX. CONCLUSION

This is a method of the way of combining the advantages of both Depth-First Search and Breadth First Search into a single method. Depth First Search is good because it allows a solution to be found without all competing branches having to be expanded. Breadth First Search is good because it does not get trapped on dead end paths. One way of combining the two is to follow a single path at a time, but switch paths whenever some competing path looks more promising than the current one does.

REFERENCES

1. Elaine Rich, Kevin Knight , 1983. Artificial Intelligence . Travelling Salesman Problem, Breadth First Search, Depth First Search, Best First search
2. Ellis Horowitz, Sartaj Sahni, Sanguthevar RajaSekaran 1993. Fundamentals of Computer Algorithms
3.Richard E. Korf 1996, Artificial Intelligence Search Algorithms
4.Gilbert Laporte, European Journal of operation Research 59 (1992) 231-247. The Travelling Salesman Problem: An Overview of exact and approximate algorithms
5.Marko Robnik S.konju, Effective use of memory in linear space best first search
6.http://www.ukessays.com/essays/computer-science/the-traveling-salesman-problem-computer-science-essay.php
7.http://www.cse.unsw.edu.au/~billw/Justsearch.html
8.http://en.wikipedia.org/wiki/Travelling_salesman_problem#Heuristic_and_approximation_algorithms
9.https://artificialintelligentsystems.wordpress.com/category/ai-searching-techniques/
10.http://en.wikibooks.org/wiki/Artificial_Intelligence/Search/Exhaustive_search/Breadth-first_search