Communications on Applied Electronics (CAE) – ISSN : 2394-4714
Foundation of Computer Science FCS, New York, USA
Volume 5 – No.10, September 2016 – www.caeaccess.org

A Preliminary Study on Minimum Spanning Tree Algorithm Approach for Travelling Salesman Problem

Evans Baidoo
Kwame Nkrumah
University of Science and Technology
Department of Computer Science

ABSTRACT
Much attention has be drawn by the Travelling salesman problem lately as it is one of the problem in mathematics and computer science which although is easy to understand but very difficult to solve.

In this paper, a preliminary study is undertaken to construct a minimum spanning tree algorithm to approximately solve the TSP.
An implementation of the travelling salesman problem using a modified pure minimum spanning tree algorithm is also presented. The propose algorithm provides an evaluation of the cost of a round trip and it executes in practical time. The algorithm verifies from a constructed tour and presents a shortcut path to all destinations. The proposed approach performance is benchmark with a case study.

Keywords
Minimum Spanning Tree, Travelling Salesman Problem

1. INTRODUCTION
The Travelling Salesman Problem normally stated as TSP for short, refers with identifying the perfect path that a salesman would take while touring between cities. The optimal solution to any given TSP would be an inexpensive way to visit a set number of cities, tripping each city only once, and then returning to the starting city. Due to its vast applications TSP, a combinatorial problem, from 1930 where it was first formulated, has proven to be one of the most comprehensively studied problems in optimization. The much interest into the study of TSP is due to its countless applications in engineering field which often include scheduling problems [2], Vehicle routing [3], constructing phylo-genetic trees [4], physical mapping problems [5], and integrated circuit designs [6]. From the mentioned applications, extra distance covered creates further costs in term of monetary or time. This forms a genuine inspiration for mathematicians, engineers and academician to suggest new techniques in effectively solving TSP.

Historically, Sir William Rowan Hamilton and Thomas Penyngton Kirkman, developed in the 1800's the mathematics related to the TSP. Presently there is no solution to the TSP that has contented mathematicians as TSP solution has eluded many mathematicians for years. Although the problem is computationally problematical, a great figure of heuristics and exact methods are identified, so that a few cases with tens of thousands of cities can be worked out.

Applegate, Bixby, Chvatal, and Cook in 1994, explained TSP consisting of 7,397 cities. A little after 4 years, they solved the problem with 13,509 cities in United States. W. Li initiated a multi-start search method to dynamic travelling salesman problem in 2011. His algorithm consists of the relations of change and search over time [7]. In year 2001, Applegate, Bixby, Chvtal, and Cook identified the optimal tour of 15,112 Germany cities.

A great deal of methods have enveloped for solving TSPs. over the years methods such as classical algorithms (Tabu-Search, Simulated Annealing etc), evolutionary algorithms (Genetic, memetic etc), natured inspired algorithms (Bee Colony optimization, Ant Colony optimization, Firefly algorithm etc), and deterministic algorithms among others have been suggested.

This paper begins with a brief discussion on TSP including it mathematical formulation in section II; followed by section III describing the basic concept of Minimum spanning Tree algorithm as well as the optimal solution for TSP. Section IV implements and evaluates the algorithm before going into section 5 which concludes the paper finding.

2. TRAVELLING SALESMAN PROBLEM
The Travelling salesman problem represent those problem class which is difficult to traditionally solve and even if it is possible to solve, computations takes a longer time to complete. Much works have gone into discussing and solving TSP such as Naive and Dynamic Programming. Both of the solutions are infeasible. In fact, there is no polynomial time result existing for this problem as the problem is a known NP-Hard problem. At present the only method known to optimally solve the travelling salesman problem of varying size, is by detailing each probable tour and probing for the tour with smallest amount of cost. But this approach will require a large number of time to compute due to permutation complexities as the number of tours increases where n is the number of cities and the number of tours is n!. Although a method as such will result in the optimal solution, is clearly not very feasible because of its high time consumption requirement to compute all the tours. In place of this approach an approximation algorithm can be used. Even though it will produce a result that isn't necessarily the best tour but instead a tour that is close to the best tour in relatively less time.

The TSP can be formulated into mathematical description just like a lot other mathematical problems. Countless literatures had tried this formulation but among which the easiest to appreciate is in [8] where the writers illustrated TSP as follows:

"The TSP can be described on a complete undirected graph G=(V,E) if it is symmetric or on a directed graph G=(V,A) if it is asymmetric. The set $V = \{1,\ldots, n\}$ is the vertex set, $E = \{(i, j) : i, j \in V, i < j\}$ is an edge set and $A = \{(i, j) : i, j \in V, i \neq j\}$ is an arc set. A cost matrix $C = (C_{ij})$ is defined on E or on A. The cost matrix fulfils the triangle inequality at any time

Communications on Applied Electronics (CAE) – ISSN : 2394-4714
Foundation of Computer Science FCS, New York, USA
Volume 5 – No.10, September 2016 – www.caeaccess.org

$C_{ij} \leq C_{ik} + C_{kj}$ for all i, j, k. In particular, this case planar problems for which the vertices are points $P_i = (X_i, Y_i)$ in the plane, and $C_{ij} = \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}$ is the Euclidean distance.

The triangle inequality is also satisfied if C_{ij} is the length of a shortest path from i to j on G."

In complex situation where the Euclidean distance is not used to compute the cost (distance)C_{ij}, the Manhattan distance may be employed.

The Manhattan distance equation as in (1)

$$C_{ij} = |X_i - X_j| + |Y_i - Y_j| \quad (1)$$

$$C_{ij} = int[6378.388\ acos(0.5 \times ((1 + g_1) \times g_2) - (1 - g_1) \times g_3) + 1] \quad (2)$$

Practically, the cost (distance) is determined using geometric distance, explained by [10] where with a given city i and city j, of longitude (lo) and latitude (la) the cost can be determined using (2), where

$$g_1 = \cos(lo_i - lo_j)$$

$$g_2 = \cos(la_i - la_j)$$

$$g_3 = \cos(la_i - la_j)$$

To convert coordinate input to longitude and latitude in radian where

PI, π = 3.141592;

deg = int (x[i]);
min = x[i]- deg;

$$la_i = \pi * \frac{deg + \frac{5*min}{3}}{180}$$

deg = int (y[i]);
min = y[i] - deg;

$$lo_i = \pi * \frac{deg + \frac{5*min}{3}}{180}$$

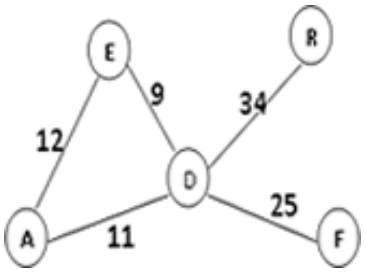
When the cost function, C_{ij} satisfies the triangle inequality, an approximate algorithm can be considered for TSP that produces a tour whose cost is certainly not more than twice the cost of an optimal tour. This paper discusses the Minimum Spanning Tree (MST) algorithm to solve Traveling Salesman Problem, which is explained in detail in the next section.

For experimental study, a case study TSPLIB [11] was taken. This case study consists of 17-city problem (Groetschel)

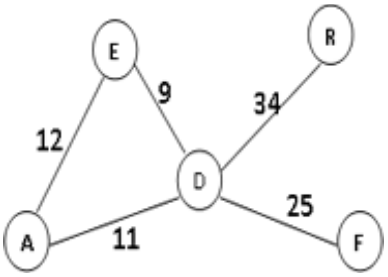
3. MINIMUM SPANNING TREE

3.1 Basic Concepts

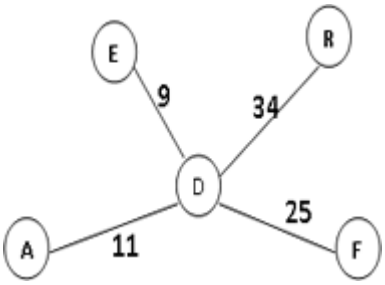
Much study has gone into the minimum spanning tree (MST) to a great extent of this century and yet in spite of its obvious simplicity, it is still not entirely understood. The MST is simply finding a spanning tree of an undirected, joined graph, where each edge has some real number such that the sum of the weights of the selected edges is least. The MST has a rich and long history which is clearly explained in the works of Graham and Hell [12], Maffioli [13] and Pierce [14]. Maffioli's assessment takes a greater view and categorizes various structures of optimum undirected tree problems, placing much prominence on their complexity of computation. Graham and Hell concentration was exclusively on the MST; not only did they trace their independent sources but also survey the algorithms. They gave an outstanding survey of end results from the most primitive known algorithm of Boruvka to the discovery of Fibonacci heaps, which were fundamental to the algorithms in Gabow et al. [15] and Fredman and Tarjan [16]. Nevertheless, this revealing studies stop in 1985 where classical algorithms were mainly stressed.
A minimum spanning tree (MST) of a graph with an associated weights or costs to each edge is a spanning tree whose total of the weights of its edges is no larger than the weight of any other spanning tree.



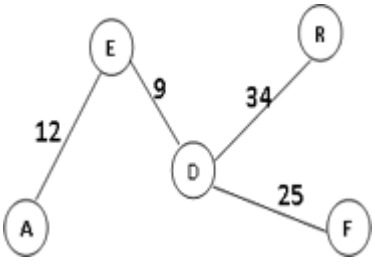
Weighted graph



Tree A: Weight (W) = 91



Tree B: Weight (W) = 79



Tree C: Weight (W) = 80

By this, it can be described as a combinatorial optimization problem. A general problem definition of the MST by [17] is given an undirected graph G= (V,E), where V indicates vertices of set with n = |V| and E edges of set with m = |E| and a real number w(e) for each edge e ∈ E referred to as the weight of edge e, the MST problem is defined as finding a spanning tree T* on G, such that $w(T^*) = \min_T \sum_{e \in T} w(e)$ is the minimum taken over all possible spanning trees of G. This problem can be solved by a lot of different algorithms. Such algorithm depending upon the assumptions taken, may be a randomized algorithm which can solve in linear expected time, linear worst case time or a solution which might be close to linear but not entirely linear.

A lot of direct applications can be linked to MST. This is in the areas of wiring connections, computer and communication networks, flow network piping, leased-line telephone and power networks, as well as transportation network connections. MST proffers some form of solution to certain problems to which it less applicable directly, such including classification problems, clustering and reliability of network. MST algorithm can be used in quite a lot of exact and approximation algorithms for the capacitated Minimum Spanning Tree problem, the matching problem and multi-terminal flow problem, in this case occurring as a sub-problem in the solution of other problems.

3.2 Modeling TSP Using MST Algorithm

The minimum spanning tree may or may not be unique depending on the pair-wise distinction of its weight or cost on its edges.

Given a set number of cities to visit to which every city is reachable from every other city somehow forming a tour, then a spanning tree of the tour is a connected sub-tour in which there are no cycles. The minimum spanning solution will be the minimum cost that will be needed to tour all cities at least once.

The travelling salesman problem is finding the least path or cost to touring all cities under discussion once and returning to the starting city. To adapt the MST algorithm to solve the traveling salesman problem, certain assumptions need to be met for this model to work. In this article, it is assume for expediency that the input graph is complete to guarantee that the starting city is adjacent to the last city

3.2.1 Adopting MST into TSP Algorithm

For a complete graph, G (V, E)

1. Choose an "anchor" vertex u ∈ V [G].
2. Let the u be the beginning and finishing point for salesman.
3. Construct a MST of the graph with u as anchor

   STEP 1: Make a set QSet that maintain track of vertices already contained in MST.

   STEP 2: Allot a key value to all vertices in the input graph. All key values can be initialized as ENDLESS. Allot key value as 0 for the anchor, u so that it is picked first.

   STEP 3: While QSet does not contain all vertices

   a) Pick a vertex r which is not there in QSet and has minimum key value.

   b) Add r to QSet.

   Renew key value of all adjoining vertices of r. Loop through al neighboring vertices to renew the key values.

   For each adjoining vertex z, if weight of edge r-z is less than the previous key value of r, update the key value as weight of r-z

4. Look into the vertex list attained in step 3 and purge from it all recurring occurrences of the same vertex
5. Assume H to be the order of vertices toured in a pre-order walk of QSet, connect all unconnected nodes.
6. Present the Hamiltonian cycle T that trips the vertices in the order H.

3.2.2 Pseudo code

```
accomplishSet = {0};            // You can use any node...
  Unreached_Set = {1, 2, ..., N-1};
  Spanning_Tree = { };

  while (UnReached_Set ≠ empty )
  {
   locate edge v = (q, r) such that:
      1. q ∈ accomplishSet
           2. r ∈ UnReached_Set
           3. v has smallest cost

    Spanning_Tree = Spanning_Tree ∪{v};

    accomplishSet   = accomplishSet ∪{r};
    UnReached_Set = UnReached_Set - {r};
  }
          Scan and remove already listed vertex if any
          Connect all vertex to form a Hamiltonian cycle
          List accomplishedSet in pre-order walk
```

The next section shows an execution of the suggested algorithm

3.2.3 Implementation

Let us consider a given set of cities. The challenge of the problem lies in identifying the shortest path passing from all vertices once. Taken R as the starting city

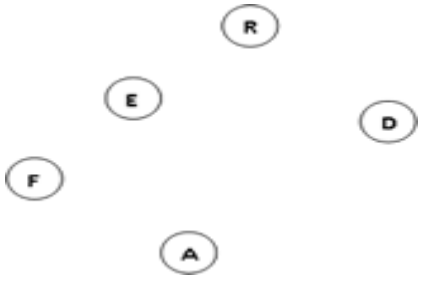


Fig. 1: A set of vertices

Communications on Applied Electronics (CAE) – ISSN : 2394-4714
Foundation of Computer Science FCS, New York, USA
Volume 5 – No.10, September 2016 – www.caeaccess.org

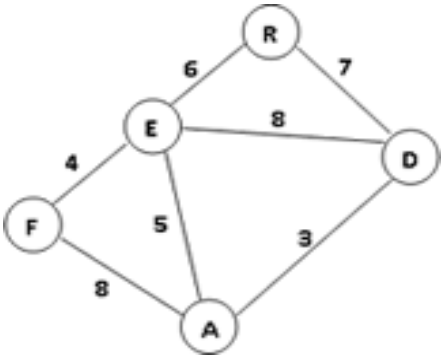


Fig. 2: A given closed graph with weights

Construct the MST with R as the root, maintain a set,

Qset = {}, and A = ∅. Select vertex with the lightest key value. Include vertex in Qset.

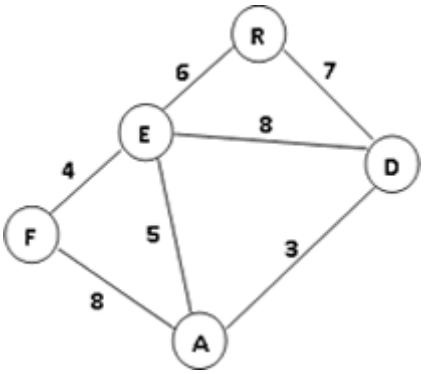


Fig 3

Stage 0

QSet = { R }

Unvisited key Set = {E, F, A, D, R}

Lightest edge = {R, E}

A = { }

Update key values of adjoining vertex. Find an edge with lightest cost which is not there in QSet that connects a reached vertex to an unreached vertex:

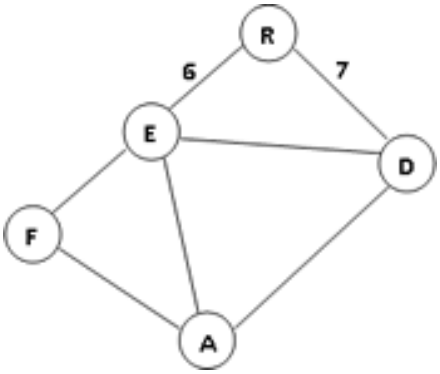


Fig 4

Stage 1

QSet = {R, E}

Unvisited key Set = {F, A, D, R}

Lightest edge = {R, E}

A = {{R, E}}

Renew key value of all adjoining vertices. Loop through all neighboring vertices to renew the key values until *QSet* includes all vertices of the given graph.

For each adjoining vertex z, if weight of edge r-z is less than the previous key value of r, update the key value as weight of r-z

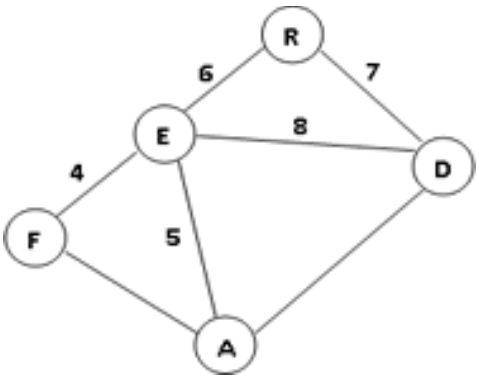


Fig 5

Stage 2

Lightest Edge = (E, F)

QSet = {R, E, F}

Unvisited key set = {A, D, R}

A = {{R, E}, {E, F}}

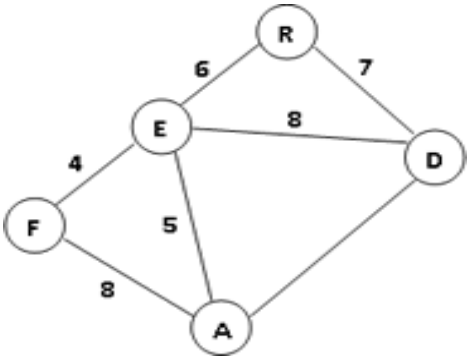


Fig 6

Stage 3

QSet = {R, E, F, A}

Lightest Edge = (E, A)

Unvisited keyset = {D, R}

A = {{R, E}, {E, F}, {E, A}}

Communications on Applied Electronics (CAE) – ISSN : 2394-4714
Foundation of Computer Science FCS, New York, USA
Volume 5 – No.10, September 2016 – www.caeaccess.org

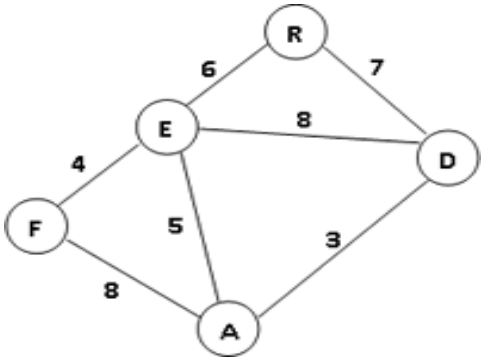


Fig 7

Stage 4

Lightest Edge = (A, D)

QSet = {R, E, F, A, D}

Unvisited keyset = {R}

A = {{R, E}, {E, F}, {E, A},{A, D}}

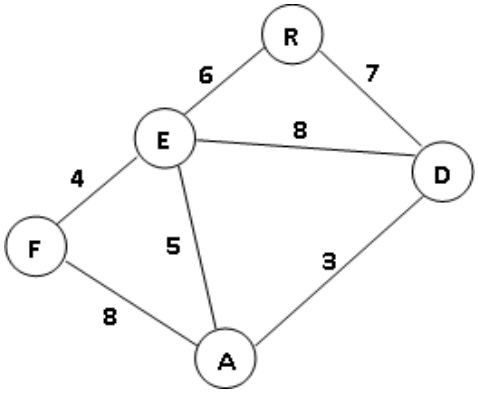


Fig 8

Stage 5

Lightest Edge = (D, R)
QSet = {R, E, F, A, D, R}

Unvisited Key set = { }

A = {{R, E}, {E, F}, {E, A}, {A, D}, {D, R}}

Done!!

All vertices are now connected.

Purge from A set all recurring occurrences of the same vertex

A = {{R, E}, {E, F}, {E, A}, {A, D}, {D, R}}

Flushing out recurring occurrences, we get an equivalent of QSet.

QSet = {R, E, F, A, D, R}

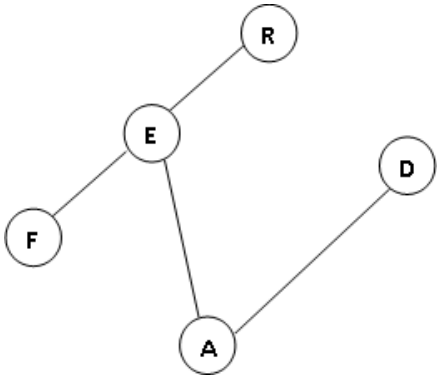


Fig 9: Minimum Spanning Tree construction.

The cost of minimum spanning cost of the tour is

(6 + 4 + 5 + 3 + 7) = 25.

Pre order walk of the vertices is visited as (R, E, F, A, D, R).

Scan and remove all duplicate vertexes in pre-order walk. Join all unconnected vertex to form Hamilton cycle.

In the case of this example, connect the visited set (R, E, F, A, D, R) which form a Hamiltonian cycle and use their cost to represent the new tour cost. New cost is (6 + 4 + 8 + 3 + 7) =28

This illustrates a tour which is returned by the complete algorithm.

From Fig 9, a complete tour can be derived as illustrated in Fig 10 which solve the traveling salesman problem

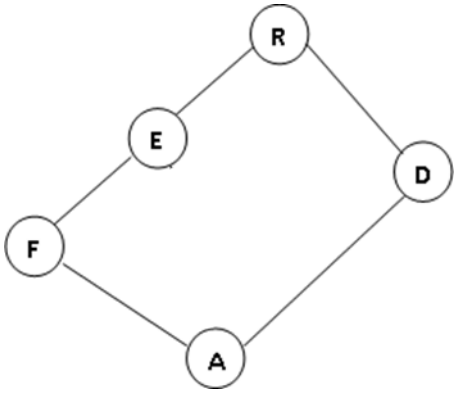


Fig 10: Complete Tour

In summary,

- construct the MST
- remove the edge(s) of all recurring vertex
- find an edge with the minimum weight that connects
- join unconnected edges together to form a Hamilton cycle.

A recommended naive approach to obtain an approximate optimal solution to the TSP is formulated as

mS = (MST * 3)/2 (3)

Where

mS = Modified solution

MST= Minimum Spanning tree cost

Communications on Applied Electronics (CAE) – ISSN : 2394-4714
Foundation of Computer Science FCS, New York, USA
Volume 5 – No.10, September 2016 – www.caeaccess.org

Table 1: Experimental result of the proposed Algorithm compared with other algorithms and Standard TSPLIB

TSP Example	Optimal Result (TSPLIB)	MST			B&B	GM[20]	FA[21]
		Minimum cost	TSP solution	Modified solution (ms)	Best results	Best results	Best results
City6	228	147	260	220.5	228	269	228
City10	1637	1099	1730	1648.5	1637	1639	1637
Gr17	2095	1421	2280	2135.5	2090	2187	2095

4. ANALYSIS

As indicated earlier, the case study is retrieved from a dataset directory containing a number of examples of data for the travelling salesman problem maintained by Gerhard Reinelt at University of Heidelberg hosted website. The website makes available numerous challenging problems that relates to TSPs. The challenging problem chosen consists of 17 cities in Groetschel due to its popularity. The following is the information provided by the problem.

```
NAME: gr17
TYPE: TSP
COMMENT: 17-city problem (Groetschel)
DIMENSION: 17
EDGE_WEIGHT_TYPE: EXPLICIT
EDGE_WEIGHT_FORMAT: LOWER_DIAG_ROW
EDGE_WEIGHT_SECTION
 0 633 0 257 390 0 91 661 228 0 412
227
 169 383 0 150 488 112 120 267 0 80
572 196
 77 351 63 0 134 530 154 105 309 34
29 0
 259 555 372 175 338 264 232 249 0
505 289 262
 476 196 360 444 402 495 0 353 282
110 324 61
 208 292 250 352 154 0 324 638 437
240 421 329
 297 314 95 578 435 0 70 567 191 27
346 83
 47 68 189 439 287 254 0 211 466 74
182 243
 105 150 108 326 336 184 391 145 0
268 420 53
 239 199 123 207 165 383 240 140 448
202 57 0
 246 745 472 237 528 364 332 349 202
685 542 157
 289 426 483 0 121 518 142 84 297 35
29 36
 236 390 238 301 55 96 153 336 0
EOF
```

To solve the problem the computation is undertaken using an intel(R) Core (TM_i3-3110M CPU@ 2.40GHz laptop equipped with 4GB RAM.
The minimum spanning tree graph is illustrated in fig 11.0
The result obtained has the path of the tree 0->12->3->12->6->7->6->16->7->5->16->13->14->2->10->4->10->9->3->8->11->15->4->1. A complete tour from fig 11 which solves the

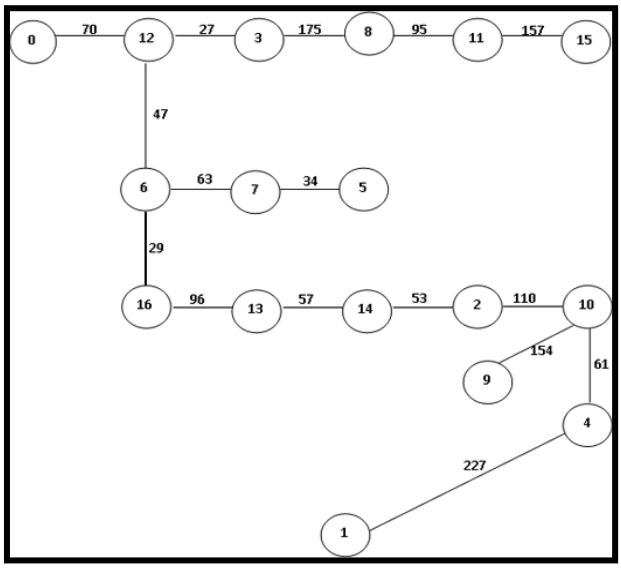


Figure 11.0: Minimum spanning tree path of Groetschel

Problem is illustrated in fig 12 and has the path 0->12->3->8->11->15->9->1->4->10->2->14->13->16->5->7->6->0 following a pre-order walk of the path as indicated by the proposed solution if it follows the triangle inequality.

In order to judge the effectiveness and efficiency of the proposed algorithm in solving TSPs, two other TSPLIB instances are added to the case study to test the end results against ones obtained by other algorithms.

Table 1 illustrate the algorithm instance City6, City10 and Gr17 of the experimental outputs and uses Greedy Method (GM), Branch and Bound (B&B), and Firefly Algorithm (FA) to perform experimental performance. As can be noted from table 1, the algorithm can identify a better path to all cities than all other algorithms under discussion in the examples listed. Though MST fails to locate the optimal path as provided by the standard library, the result gives optimal approximation of a complete trip.

A closer view from Table 1 indicate that the proposed algorithm returned a minimum cost of 1421 in Gr17 case study, a TSP solution of 2280 and a modified cost of 2131.5, 36.5 more than 2095 as provided by standard TSPLIB. The modified cost was obtained using equation (3) to estimate the optimal cost of the proposed algorithm against the standard TSPLIB. The TSP solution is algorithm adapted to solve the Travelling salesman problem in this paper.

Compared with B&B, GM and FA, in the case of all TSP instances, all can find an optimal path but partially deviating from the optimal results (standard TSPLIB) in some cases whiles the proposed algorithm minimum cost returns much less weight cost than the optimal result . The TSP solution of the MST may perform better than the Greedy method in some cases.

From Table 1, MST provides a tour cost which is never greater than the cost of the best travelling salesman tour. The modified solution proposed in this paper returned a cost which is better than the optimal result in small city problems but partially deviated from the optimal results and against other algorithms in larger TS problems. The cost of a full walk in all instances (in this case) visiting the towns twice will provide at most 2 time worse solution than the optimal tour and may not represent a solution to the Travel salesman problem. In all instances, FA returned the best results to all other algorithms. Results were exact as the standard TSPLIB cost.

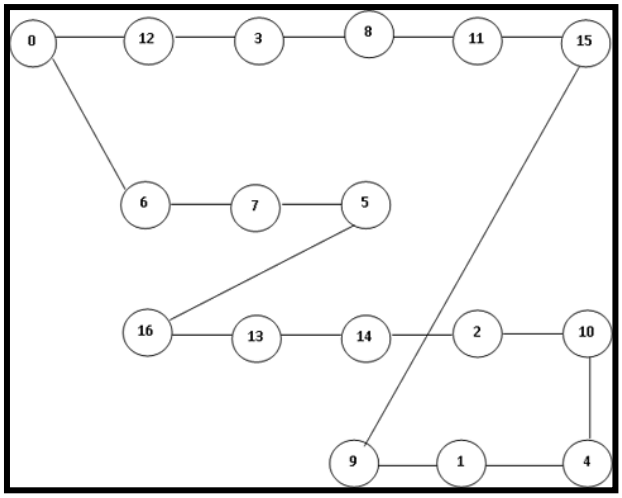


Figure 12: Complete tour of Groetschel 17

5. CONCLUSION

Basic Minimum spanning tree approach is applied with some modifications in the algorithm to adapt it for Travelling Salesman Problem (TSP). The objective is to find the shortest distance, construct and estimate the cost for the salesman to visit all the cities. Experimental result is obtained and compared to the standard case study TSP instances and other existing algorithms.

It shows that the proposed algorithm provides a better estimation result which at worse case provides at most twice the cost of the proposed algorithm to the best possible solution. In larger TS problems the proposed algorithm may fail to generate optimal tour but can be used to construct the traveling salesman path.

From the results obtained, it can be concluded that MST returns a lower bound on the cost of travelling to all cities.

In the course of the TSPLIB example, the algorithm indicated a high searching speed and accuracy of optimization with a superior applied value in solving combinatorial optimization and related problems.

For further studies, the results can be further improved and modeled by using christofides algorithm and other local search methods integrated with greedy approaches. Additionally, the study can be extended with a discrete state transition algorithm implementation in problem instances much larger than those considered in this study.

6. ACKNOWLEDGMENTS

Special appreciation goes to Mr. Bismark Boakye Yiadom, Mr. Derrick Lamptey and Mr. Stephen Oppong for their immense support and co-operation. The work was also supported by Mr. Dominic Asamoah, Kwame Nkrumah University of Science and Technology.

7. REFERENCES

[1] http://en.wikipedia.org/wiki/Travelling salesman problem.

[2] Whitely, D., Starkweather, T. and D'Ann, F. 1989. Scheduling problems and traveling salesman: The genetic edge recombination operator, in Proc.3rd Int. Conf. Genetic Algorithms, 1989, pp.133–140.

[3] Clarke, G. and Wright, J.W. 1964. Scheduling of vehicles from a central depot to a number of delivery points, Oper. Res., vol. 12, pp. 568–581, 1964.G. http://dx.doi.org/10.1287/opre.12.4.568

[4] Korostensky, C. and Gonnet, G. H. 2000. Using traveling salesman problem algorithms for evolutionary tree construction, Bioinformatics, vol. 16, no. 7, pp. 619–627, 2000. http://dx.doi.org/10.1093/bioinformatics/16.7.61

[5] Alizadeh, F., Karp, R. M., Newberg, L. A. and Weisser D. K., 1993. Physical mapping of chromosomes: A combinatorial problem in molecular biology, in Proc. 4th ACM-SIAM Symp. Discrete Algorithms (SODA), 1993, pp. 52–76.

[6] Kirkpatrick, S., Gelatt Jr, C. D. and Vecchi, M. P. 1983. Optimization by simulated annealing, Science, vol. 220, pp. 498–516, 1983. http://dx.doi.org/10.1126/science.220.4598.671

[7] Li, W.2011. A Parallel Multi-Start Search Algorithm for Dynamic Traveling Salesman Problem. 10th International Conference on Experimental Algorithm, pp 65-75, 2011.

[8] R. Matai, S. Singh, M. L. Mittal,"Traveling Salesman Problem: an Overview of Applications, Formulations, and Solution Approaches,"Traveling Salesman Problem, Theory and Applications. InTech, 2010.

[9] http://www.geeksforgeeks.org/greedy-algorithms-set-5-prims-minimum-spanning-tree-mst-2/

[10] Hendtlass, T. 2010. TSP Optimisation Using Multi Tours Ants, 17th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, pp. 523-532, 2010.

[11] https://people.sc.fsu.edu/~jburkardt/datasets/tsp/tsp.html

[12] Graham, R. L., and Hell, P. 1985. On the history of the minimum spanning tree problem. Annals of the History of Computing 1985;7:43}57.

[13] Maffioli, F. 1981. Complexity of optimum undirected tree problems: a survey of recent results. In: Ausiello G, Lucertini M, editors. Analysis and design of algorithms in combinatorial optimization. International Center for Mechanical Sciences. CISM Courses and Lectures, 266. New York: Springer, 1981. p. 107}28.

[14] Pierce, A. R. 1974. Bibliography on algorithms for shortest path, shortest spanning tree and related circuit routing problems (1956}1974). Networks1975;5:129}49.

[15] Fredman, M. L., and Tarjan, R. E. 1987. Fibonacci heaps and their uses in improved network optimizationalgorithms. J. ACM 34, 596–615.

[16] Gabow, H. N., Galil, Z., Spencer, T., and Tarjan, R. E. 1986. Efficient algorithms for finding minimum spanning trees in undirected and directed graphs. Combinatorica 6, 109–122.

[17] Cuneyt F. B. and Khalil S. H. 2001. Minimum-weight spanning tree algorithms. A survey and empirical study, Computer & Operations Research 28(2001) 767-785

[18] http://www.csl.mtu.edu/cs4321/www/Lectures/Lecture%2028%20-%20Approximation%20Algorithm.htm

[19] Cook, W. 2009. History of the TSP." *The Traveling Salesman Problem*. Oct 2009. Georgia Tech, 22 Jan 2010. <http://www.tsp.gatech.edu/index.html>.

[20] Lawler, E. L, Lenstra, J. K., Rinnooy Kan, A.H.G and.Shmoys, D.B 1985. The Traveling Salesman Problem. John Wiley & Sons.

[21] Sharad, N. et al. 2013. Solving Travelling Salesman Problem using Firefly Algorithm. International Journal for Research in Science & Advanced Technologies Issue-2, Volume-2, 053-057