An Approach for Solving Minimum Spanning Tree Problem Using a Modified Ant Colony Optimization Algorithm

Article *in* American Journal of Applied Mathematics · December 2022

DOI: 10.11648/j.ajam.20221006.11

2 authors:

Oshan Niluminda
Rajarata University of Sri Lanka
16 PUBLICATIONS **31** CITATIONS

SEE PROFILE

Uthpala Ekanayake
Rajarata University of Sri Lanka
47 PUBLICATIONS **116** CITATIONS

SEE PROFILE

American Journal of Applied Mathematics
2022; 10(6): 223-235
http://www.sciencepublishinggroup.com/j/ajam
doi: 10.11648/j.ajam.20221006.11
ISSN: 2330-0043 (Print); ISSN: 2330-006X (Online)

An Approach for Solving Minimum Spanning Tree Problem Using a Modified Ant Colony Optimization Algorithm

Kankanam Pathiranage Oshan Niluminda*,
Ekanayake Mudiyanselage Uthpala Senarath Bandara Ekanayake

Department of Physical Sciences, Faculty of Applied Sciences, Rajarata University of Sri Lanka, Mihinthale, Sri Lanka

Email address:

kponiluminda@gmail.com (Kankanam Pathiranage Oshan Niluminda)

*Corresponding author

To cite this article:

Kankanam Pathiranage Oshan Niluminda, Ekanayake Mudiyanselage Uthpala Senarath Bandara Ekanayake. An Approach for Solving Minimum Spanning Tree Problem Using a Modified Ant Colony Optimization Algorithm. *American Journal of Applied Mathematics*. Vol. 10, No. 6, 2022, pp. 223-235. doi: 10.11648/j.ajam.20221006.11

Received: October 22, 2022; **Accepted**: November 4, 2022; **Published**: December 8, 2022

Abstract: In real life, people often want to do tasks at the lowest possible cost while also taking into consideration travel time and distance. The term "minimum spanning tree" refers to the spanning tree that has a weight that is less than or equal to the weight of all other feasible spanning trees. A spanning tree is created when every vertex in a network is linked and has no cycles in it; there must be no other spanning tree with a lesser weight. The minimum spanning tree problem has been solved using a variety of methods that have been published in the literature. They provide the best answer to the minimum spanning tree problems given an undirected graph using Prim's and Kruskal's algorithms. A probabilistic method for resolving computational issues that may be simplified to finding the best route via graphs is the Ant Colony Optimization Algorithm (ACOA). This algorithm has been developed based on how ants search for a route between their nest and a food source while foraging. This paper proposes a novel technique and effective method for studying the large scale of the problem and determining the minimum cost-spanning tree of a connected weight undirected graph with fewer iterations using the Modified Ant Colony Optimization Algorithm (ACOA). When the graph has a large number of nodes, this novel approach is simpler and easier to implement than other existing algorithms, and by comparing our methods to Prim's and Kruskal's, we can get the same results.

Keywords: Ant Colony Algorithm, A Minimum Weight Spanning Tree, Prim’s Algorithm, Kruskal’s Algorithm, Undirected Graph

1. Introduction

A spanning tree T is a connected undirected graph that connects all the vertices of G = (V, E) with no cycles in the graph. A disconnected graph has no spanning tree because it cannot be stretched to all of its vertices. If you remove one edge from the spanning tree, it will be disconnected [1]. A loop is formed by adding one edge to the spanning tree. A loop is an edge that connects two vertices [2, 3]. A complete undirected graph can have $n^{(n-2)}$ spanning trees, where n is the number of nodes. A graph that connects all pairs of vertices connected by an edge is called a complete graph [4]. Every connected and undirected graph has at least one spanning tree, but a disconnected graph doesn’t have any spanning trees. A spanning tree can be constructed from a complete graph by removing edges. The Minimum Spanning Tree (MST) is a special type of spanning tree that minimizes the weight (or "length") of the edges of the tree. The total cost of the minimum spanning tree can be calculated by the summation of the weights of all the edges in the tree. Many applications use a minimum spanning tree, including telecommunications networks, civil network planning, cluster analysis, water supply networks, and transportation networks [5, 2]. Several algorithms for finding the graph's minimum spanning trees have been developed. These algorithms are divided into two types based on how they are implemented. Line (Edge)-based MST and Node (Vertices)-based MST are two examples. Kruskal's and Reverse Delete's algorithms are line-based algorithms, whereas Prim's and Dijkstra's algorithms are node-based [6, 7]. Otakar Borvka, a Czech researcher, developed the first known algorithm for finding a minimum

spanning tree in 1926, and this algorithm is known as Borvka's algorithm. Following that, Prim and Kruskal created two distinct algorithms. Vojtech Jarnk proposed one of the algorithms in 1930, which Robert C. Prim implemented in 1957. Edsger W. Dijkstra rediscovered it in 1959 and named it Prim's algorithm. Joseph Kruskal published another algorithm, Kruskal's algorithm, in 1956 [5]. The Reverse deletes algorithm, also known as the "Greedy Algorithm," is the inverse of Kruskal's algorithm for finding the shortest path [8].

In his Ph.D. thesis in 1992, Marco Dorigo proposed the Ant Colony Optimization Algorithm (ACOA). It is a method of calculating the shortest path between a source and a destination. It is based on the behavior of ants as they travel from their nest (colony) to a food source in search of food. The method they use aids in determining the best route between the source and destination [9, 17]. The Ant Colony Optimization Algorithm (ACOA) is a probabilistic technique that can be used to solve a wide range of optimization problems [10, 18]. For example, the minimum spanning tree problem, the shortest path problem, the traveling salesman problem, the traffic assignment problem, the scheduling problem, and so on [4, 7]. The modified ACOA is used in this research to present a unique strategy and efficient method for analyzing the problem at a large scale and finding MST of a connected weight undirected graph with fewer iterations. In the end, the proposed method will compare with other existing methods.

2. Preliminaries

In graph G, each vertex is represented by a point and each edge by a line. An edge connects two vertices, which are commonly referred to as the edge's endpoints. A graph must have at least one vertex but no edges.

2.1. Basic Definitions

2.1.1. Graph

A graph G is defined as a set of a finite number of vertices or nodes (V) connected by edges or arcs (E). Otherwise, a graph is made up of two elements: vertices (V) and edges (E).

2.1.2. Degree of the Vertex

The degree of a vertex in an undirected graph is the number of edges that intersect with it. If the graph contains a loop, the loop at a vertex contributes twice as much to the vertex's degree. The degree of vertex $V(G)$ is denoted by $\deg(v)$ or $\delta(v)$ [13].

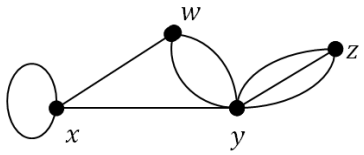

Figure 1. *Connected Graph.*

In figure 1:

$$V(G) = \{w, x, y, z\}$$

$$E(G) = \{wx, xx, wy, yw, xy, yz, zy, yz\}$$

$$\delta(w) = 3, \delta(x) = 4, \delta(y) = 6, \delta(z) = 3$$

2.1.3. Adjacent Vertices

In graph G (V, E), two edges are said to be adjacent if they share at least one vertex, and two vertices are said to be adjacent if they are joined by an edge [12].

2.1.4. Subgraphs

A subgraph of a graph G (V, E) is a graph with vertices that belong to V(G) and edges that belong to E(G). Subgraphs of a graph can be obtained by deleting edges and vertices from the given graph [12].

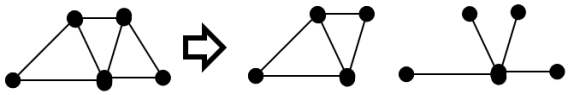

Figure 2. *Edges & Vertices deleting.*

2.1.5. Regular Graph

A regular graph is one in which every vertex has the same degree as a simple graph. i.e $\delta(v)$ is constant for all $v \in V(G)$.

2.1.6. Loop

A loop is defined in a graph as an edge that has the same start and end vertex (joins a vertex to itself).

2.1.7. Cycle

A cycle is defined as any path that begins and ends at the same vertex.

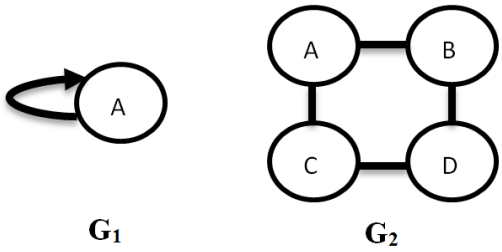

Figure 3. *Loop & Cycle.*

In above figure 3, G_1 graph has a loop and G_2 graph has a cycle.

2.1.8. Directed and Undirected Graph

Edges in undirected graphs do not have a direction. Each edge can be traversed in both directions, indicating a two-way relationship. A simple undirected graph with three nodes and three edges is depicted in this figure. Edges in directed graphs have a direction.

2.2. Spanning Tree

An undirected, connected graph's subgraph is a spanning tree.

2.3. Minimum Spanning Tree

The minimum spanning tree is the spanning tree in which

the sum of the edge weights is the smallest. The weight of the spanning tree is the sum of the weights assigned to the spanning tree's edges.

2.4. Minimum Weight Spanning Tree (MWST)

The Minimum Spanning Tree is the tree with the fewest cumulative edge weights. It is made up of all of the vertices of the same graph. A minimum spanning tree's weight is the sum of the weights assigned to each of the tree's edges [5].

If a connected undirected graph G (V, E), then the spanning tree is formally defined as $G(V, E')$ and $E' \subseteq E$. The weight of the minimum spanning tree is defined as equation (1):

$$w(T) = \sum_{e \in T} w(E') \tag{1}$$

Prim's and Kruskal's algorithms are two different algorithms which use to find the minimum weight-spanning tree in a weighted undirected graph.

2.5. Prim's Algorithm

A well-known greedy algorithm is Prim's algorithm. The implementation of this algorithm is given below [13]:

Step 1: Choose any vertex randomly.

Step 2: Select the minimum weight edge adjacent to that vertex.

Step 3: All edges that connect the tree to new vertices should be found.

Step 4: Find the edge with minimum weight among the other edges and include it in the current tree.

Step 5: If incorporating that edge generates a cycle, discard it and go for the next edge with the least weight.

Step 6: Repeat steps 3–5 until all vertices have been included and the Minimum Spanning Tree (MST) has been generated.

2.6. Kruskal's Algorithm

Another well-known greedy algorithm is Kruskal's Algorithm. The given graph must be weighted, connected, and undirected to use Kruskal's algorithm. The steps for implementing Kruskal's Algorithm are as follows [9, 10]:

Step 1: Sort all the edges from minimum weight to maximum weight.

Step 2: Take the minimum weighted edge to connect the graph's vertices.

Step 3: If adding an edge creates a cycle, discard it and move on to the next edge with the minimum weight.

Step 4: Continue adding edges until all of the vertices are joined and you have a Minimum Spanning Tree (MST).

2.7. Ant Colony Optimization Algorithm (ACOA)

An ant colony optimization algorithm is a population-based search technique for finding the optimal path that is based on the behavior of ants. Ants use pheromones to move from their nest (say, city *i*) to a food source (say, city *j)*. If an ant *k* is currently in city *i* and wants to move to city *j*, then the probabilistic transition rule and pheromone update rule apply [15].

2.7.1. Transition Rule

The probabilistic transition rule of the Ant Colony Optimization Algorithm is as follows in equation (2):

$$P_{ij}^{k} = \begin{cases} \frac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{z \in Allowed_k}[\tau_{iz}]^{\alpha}[\eta_{iz}]^{\beta}} & j \in Allowed_k \\ 0; Otherwise \end{cases} \tag{2}$$

Where, P_{ij}^{k} denotes the probability branch of k^{th} ant from i^{th} city to j^{th} city and the amount of pheromone on component between city i and city j is denoted by τ_{ij} and its heuristic value is denoted by η_{ij}. α and β are both parameters. $(0 \leq \alpha, \beta \leq 1)$ [15, 16].

2.7.2. Pheromone Update Rule

The local update rule of pheromone trails is applied for each route is given below (3), (4), and (5) equations:

$$\tau_{ij}(t+1) = \rho.\tau_{ij}(t) + \Delta\tau_{ij} \tag{3}$$

$$\Delta\tau_{ij} = \sum_{k=1}^{z} \Delta\tau_{ij} \tag{4}$$

$$\Delta\tau_{ij}^{k} = \begin{cases} \frac{\omega}{d_k} & if\ ant\ k\ travels\ on\ city\ (i,j) \\ 0 & ; Otherwise \end{cases} \tag{5}$$

Here, the distance of the best path is denoted by d_k. The parameter ω is to adjust the amount of pheromone deposited and generally, it is set to be 1 [11, 12].

3. Methodology

The Improved Ant Colony Optimization Algorithm, a proposed approach for finding an optimal solution, is discussed in this section.

3.1. The Modified Ant Colony Optimization Algorithm (MACOA)

For our development, we use the probabilistic transition rule of the Ant Colony Optimization Algorithm. The pheromone trail is used by an ant to move from one source to another. As a result, if i^{th} ant decides to travel next unvisited j^{th} destination (city) is given by the following formula (6).

$$P_{ij} = \begin{cases} \frac{\left[1/w_{ij}\right]^{0.5}}{\sum_{z \in Allowed_k}\left[1/w_{ij}\right]^{0.5}}; i^{th}\ ant\ visits\ the\ j^{th}\ city \\ 0; Otherwise \end{cases} \tag{6}$$

We assume $\alpha = 0$ and $\beta = 0.5$

Here,

P_{ij} = Probability to branch from i^{th} node to j^{th} node.

w_{ij} = Cost or distance between i^{th} node and j^{th} node.

The steps for solving the minimum spanning problem are as follows.

3.2. Proposed New Minimum Spanning Tree Algorithm

Step 1: Calculate the path based on the probability of every

vertex by using the Modified Ant Colony Optimization Algorithm.

Step 2: Determine the degree of each vertex.

Step 3: Select the vertices with the highest degree that are not adjacent to each other.

Step 4: Determine the maximum probability path and choose the best two paths from each selected vertex, deleting the other probability paths.

Step 5: If the degree of the vertex is below 3, proceed to step 6. Otherwise, proceed to step 4 for every vertex.

Step 6: If the subgraphs are disconnected, connect every subgraph by using the maximum probability of two paths.

Step 7: If the graph has cycles, remove those cycles by deleting the minimum probability paths.

Step 8: If the termination condition is chosen, the best vertices with the best paths are chosen.

The flow chart representation of the newly proposed method is also provided in Figure 6.

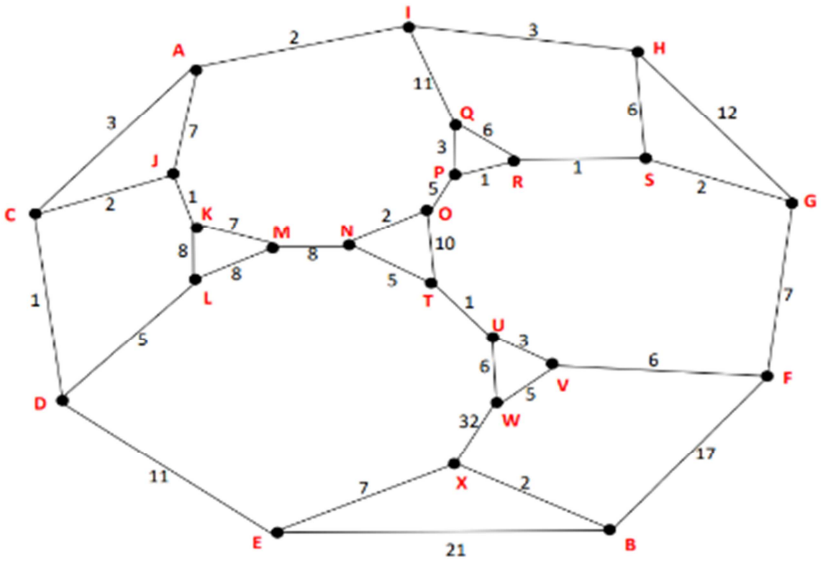

Figure 4. *Connected Undirected Graph 1.*

4. Results and Discussion

Graph theory has a wide range of applications in operations research. A graph is defined as a collection of points connected by lines in a finite number of ways. This paper, look at two different examples [13, 14] of how to use the proposed approach to find the minimum spanning tree of an undirected graph.

4.1. Example 1

Step 1:

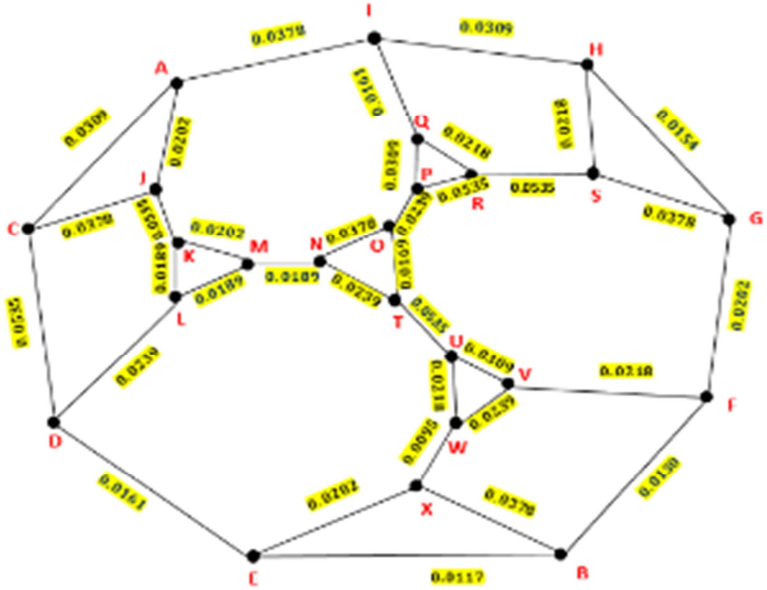

Figure 5. *Graph with probabilities.*

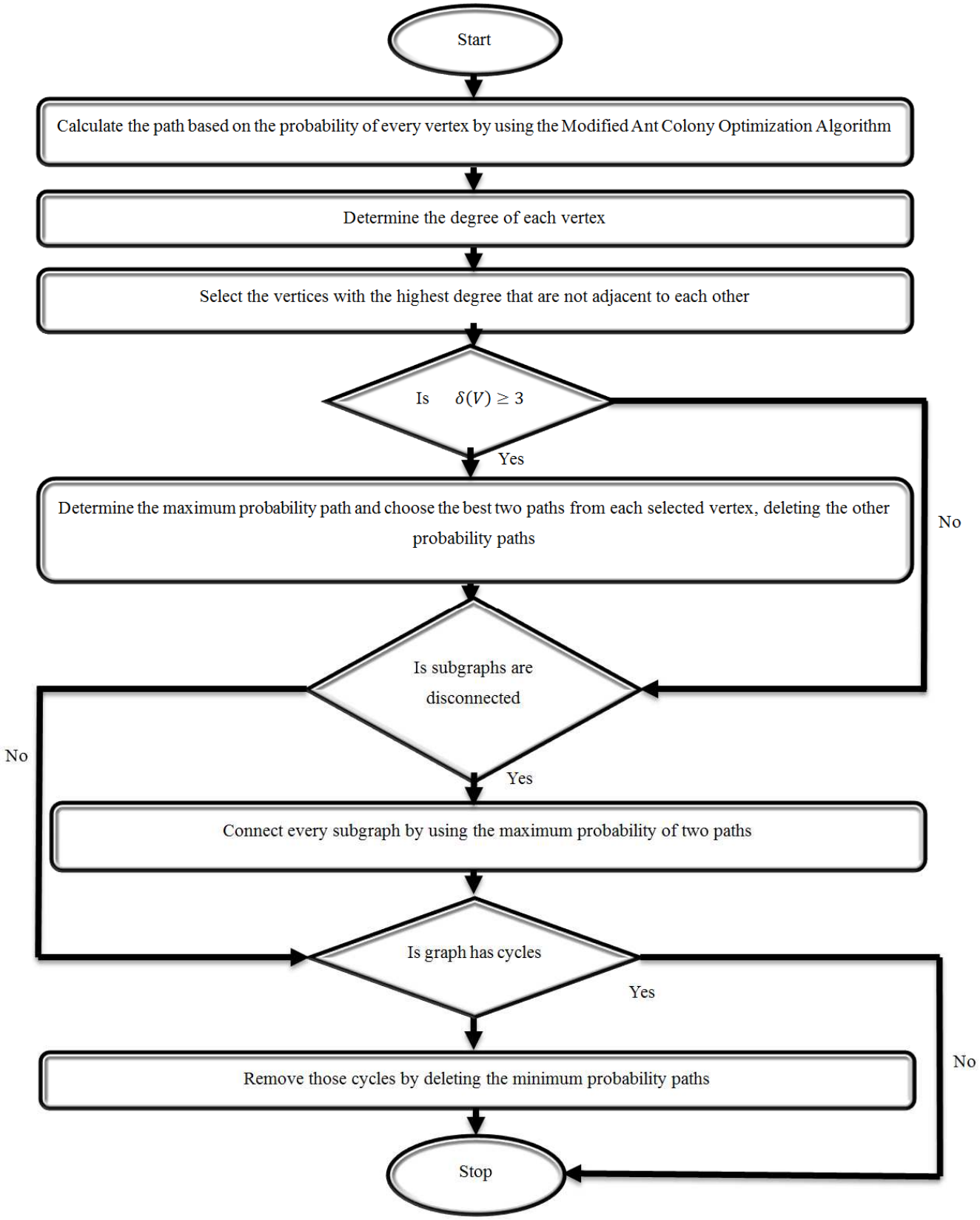

Figure 6. *Flow chart representation of the new algorithm, MACOA.*

Step 2, 3 & 4:

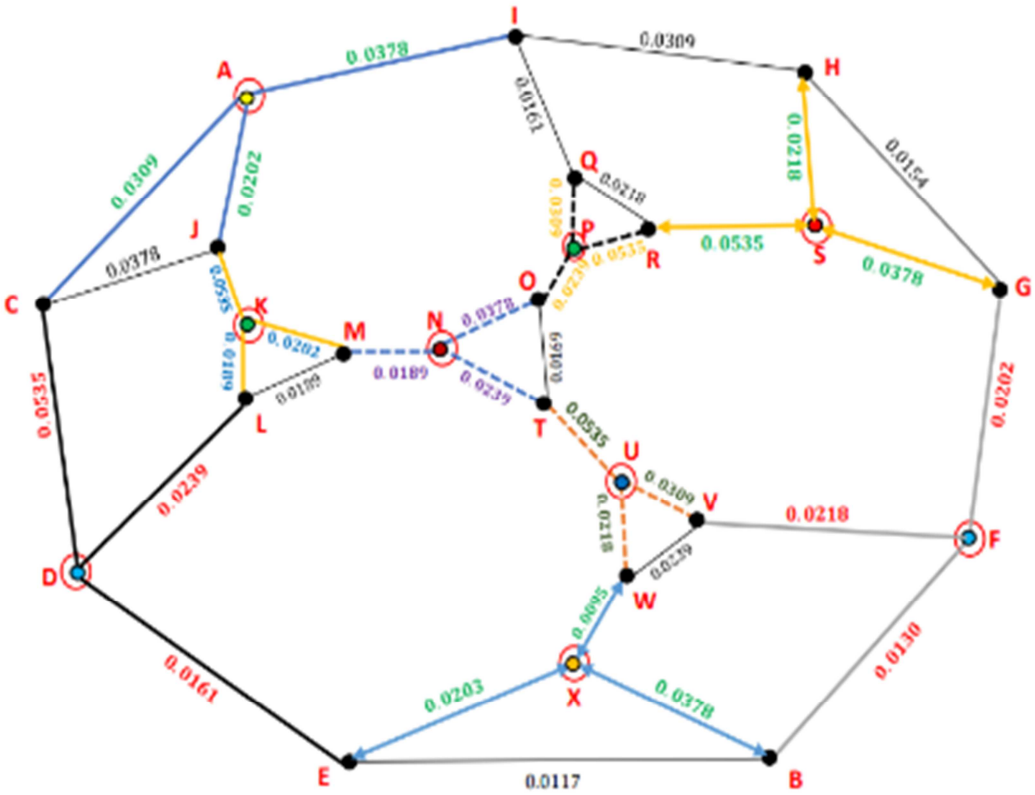

Figure 7. *Reduced Graph 1.1.*

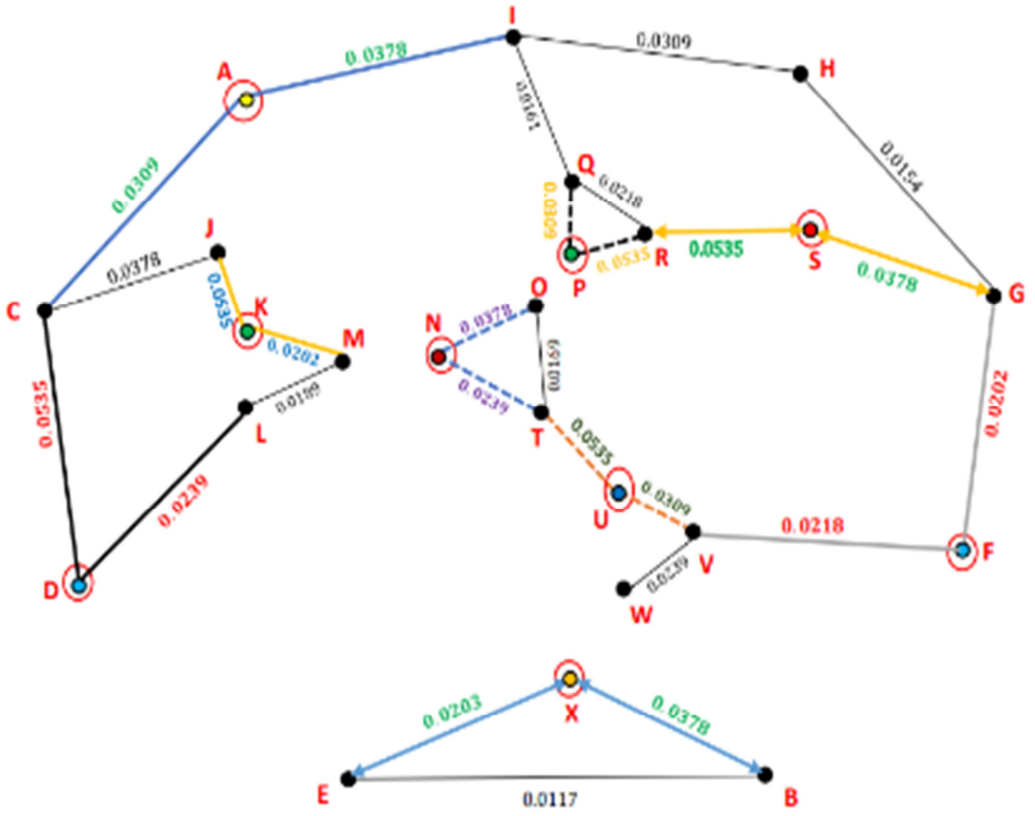

Figure 8. *Reduced Graph 1.2.*

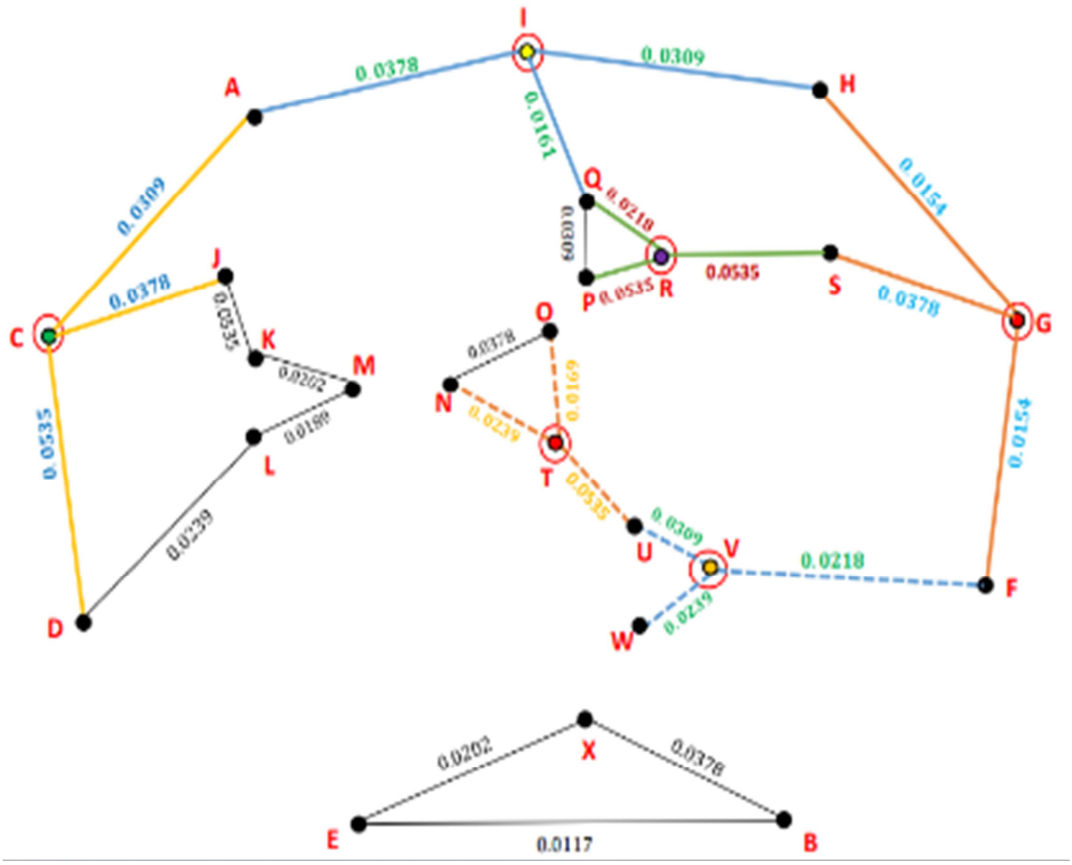

Figure 9. *Reduced Graph 1.3.*

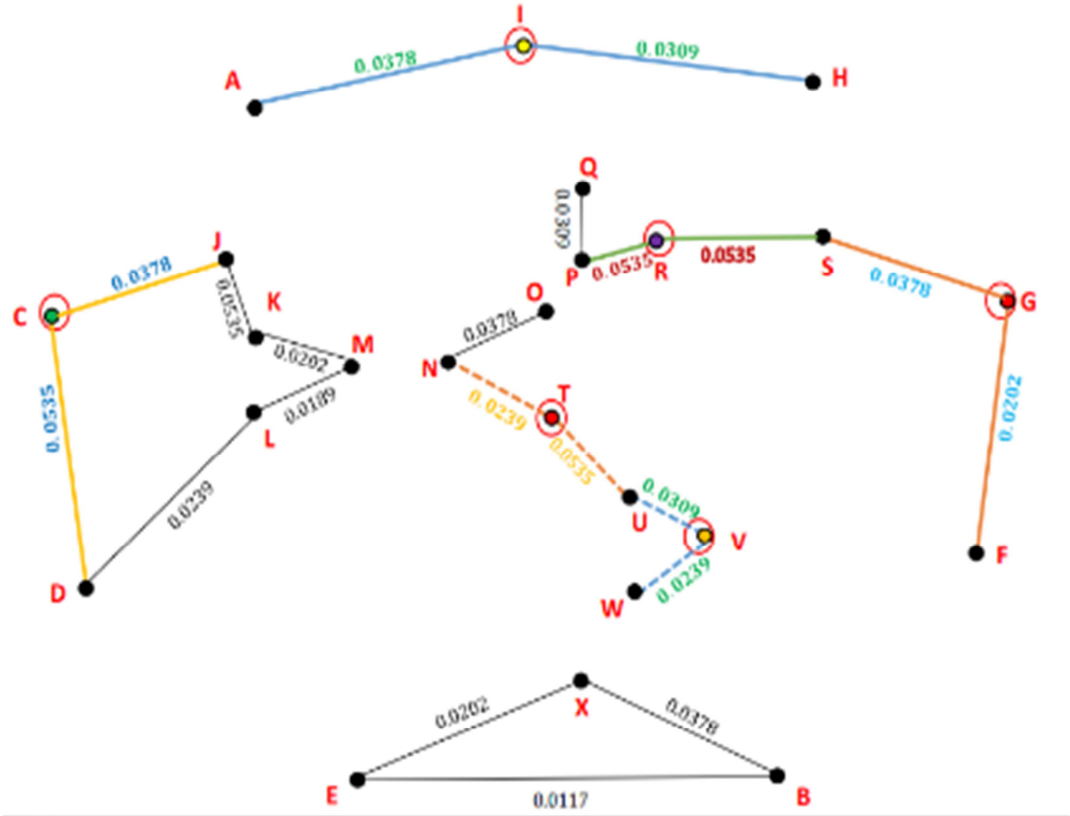

Figure 10. *Reduced Graph 1.4.*

Step 6:

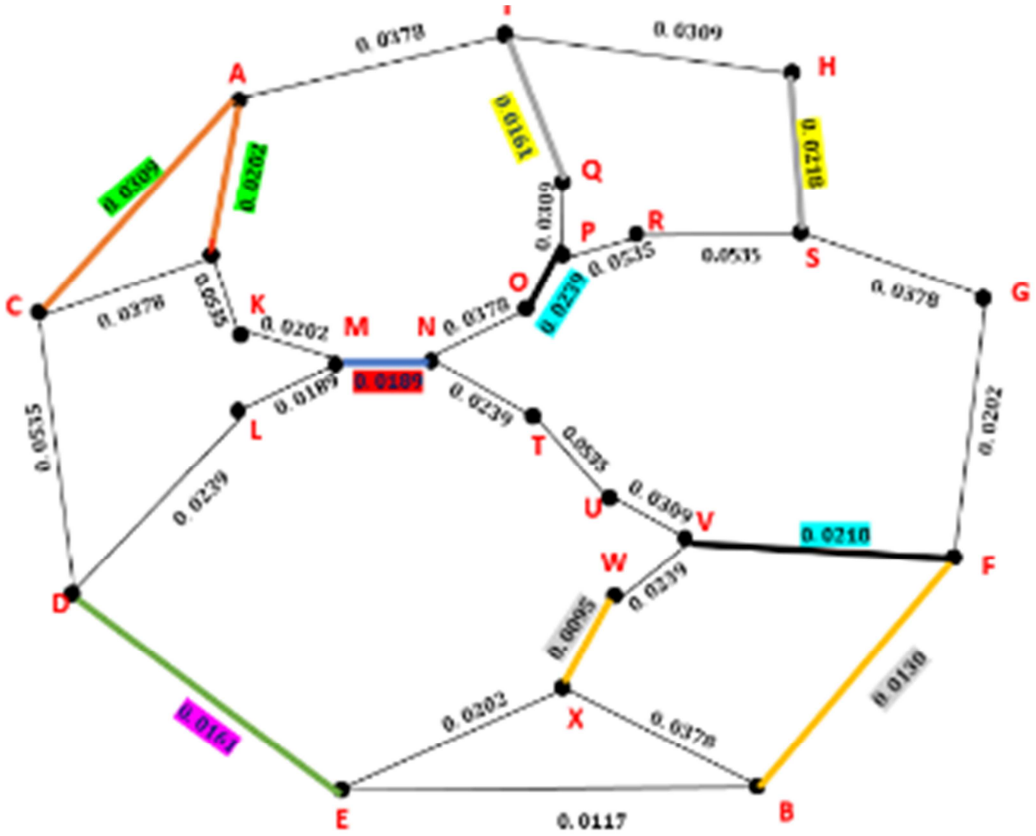

Figure 11. *Graph of Subgraph Connecting.*

Step 7:

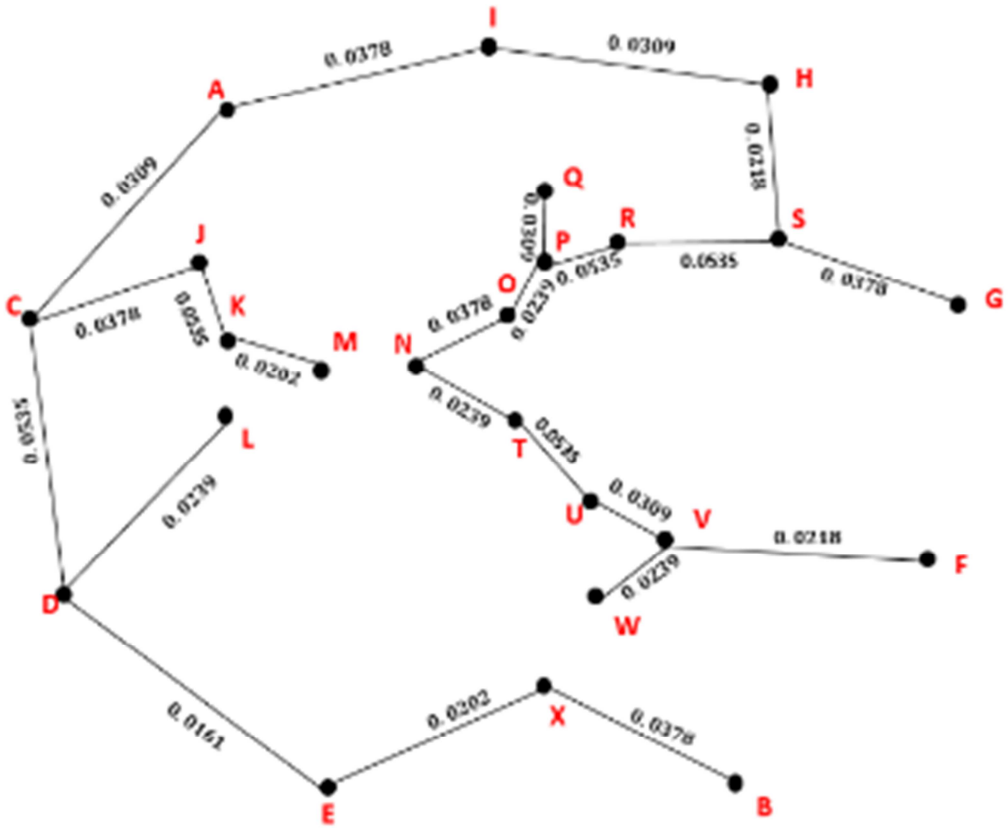

Figure 12. *MST with Probabilities.*

The Minimum Spanning Tree (MST).

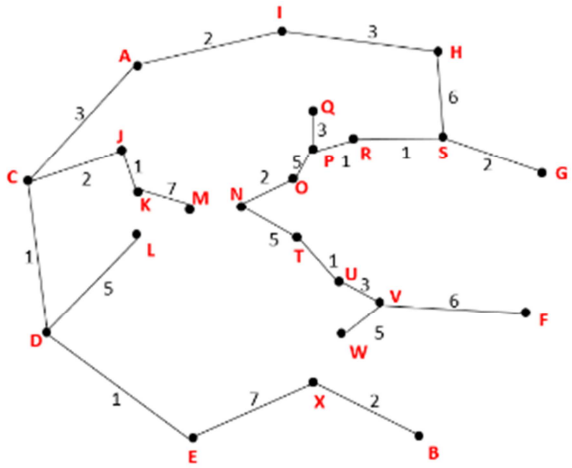

Figure 13. *Minimum Weight Spanning Tree.*

This connected graph has 24 nodes (vertices) and 36 edges. The minimum spanning tree has 23 edges and the total weight of the minimum spanning tree (MST) is 84 units.

4.2. Example 2

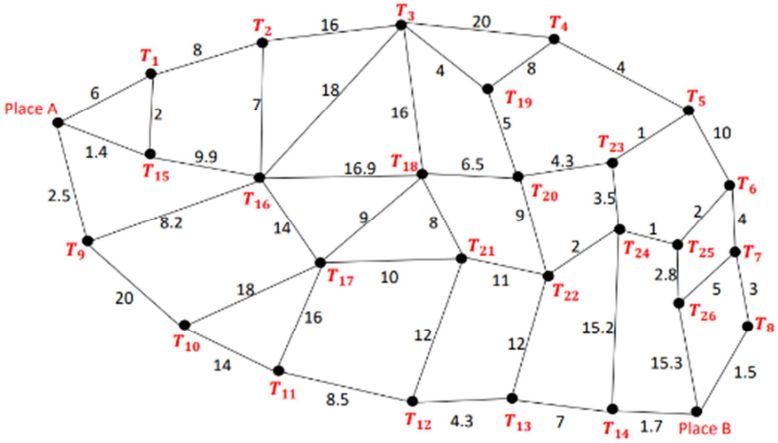

Figure 14. *Connected Undirected Graph 2.*

Step 1:

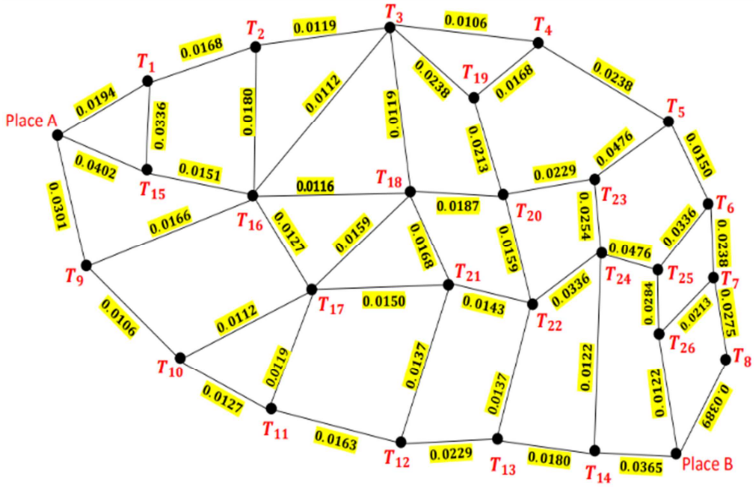

Figure 15. *Graph with Probabilities.*

Step 2, 3 & 4:

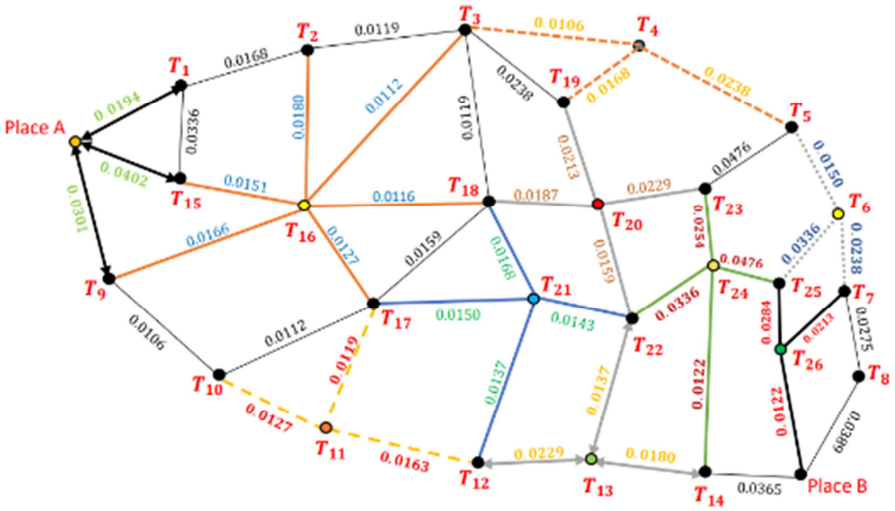

Figure 16. *Reduced Graph 2.1.*

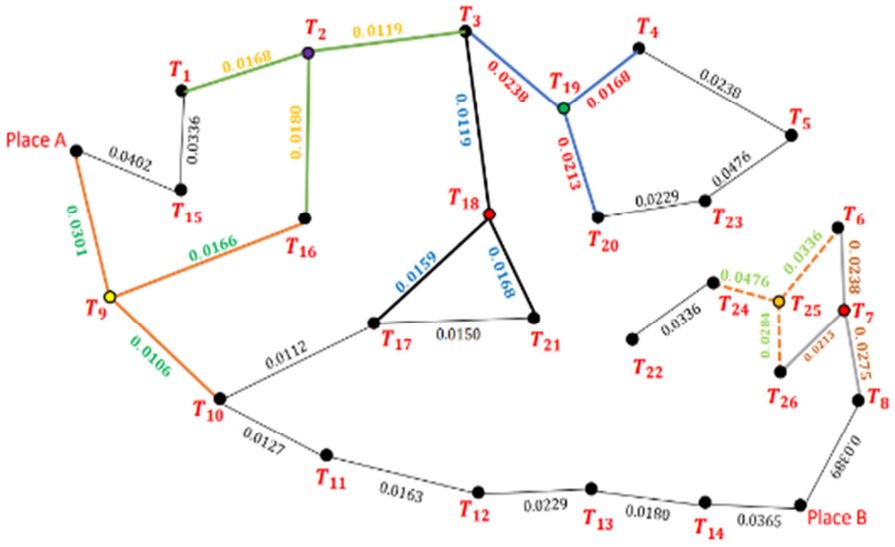

Figure 17. *Reduced Graph 2.2.*

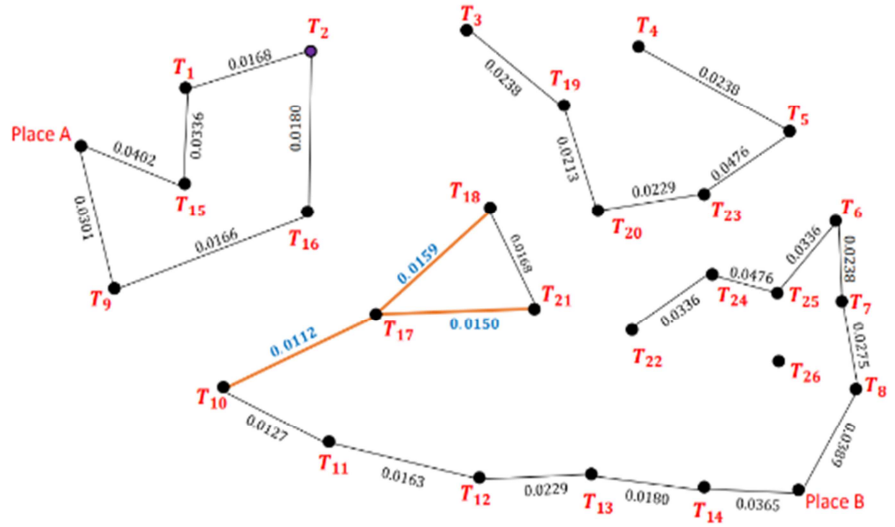

Figure 18. *Reduced Graph 2.3.*

Step 6:

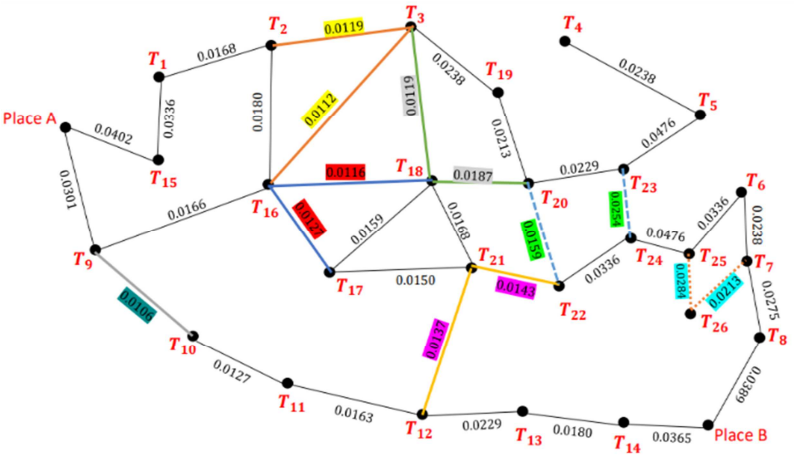

Figure 19. *Graph of Sub graph Connected.*

Step 7:

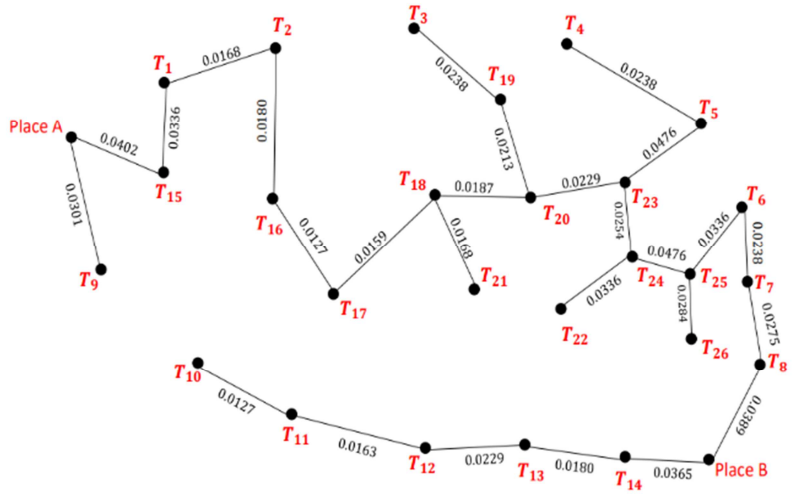

Figure 20. *MST with Probabilities.*

The Minimum Spanning Tree (MST)

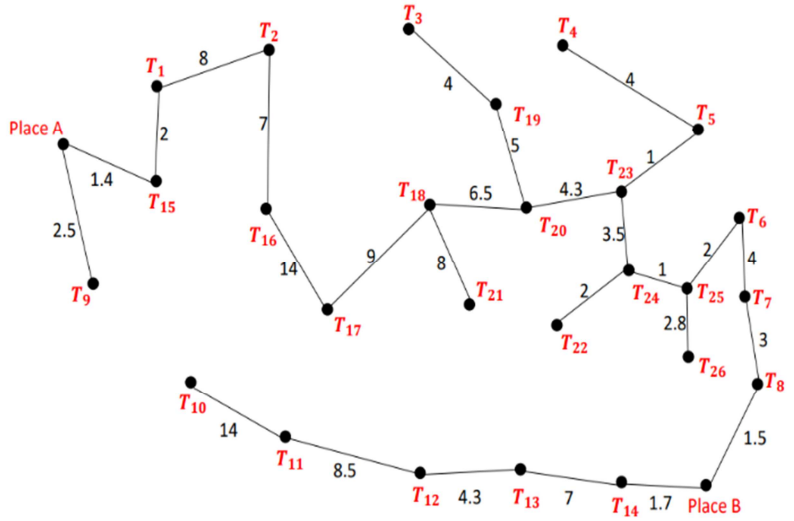

Figure 21. *Minimum Weight Spanning Tree.*

This connected graph has 28 nodes (vertices) and 48 edges. The minimum spanning tree has 27 edges and the total weight of the minimum spanning tree (MST) is 132 units. The results obtained in the Modified Ant Colony Optimization (MACO) Algorithm of examples 1 and 2 can compare with Prim's and Kruskal's algorithms.

Table 1. *Comparative Analysis of example 1 and 2.*

Example No.	Prim's Algorithm	Kruskal's Algorithm	MACO Algorithm
1	84	84	84
2	132	132	132

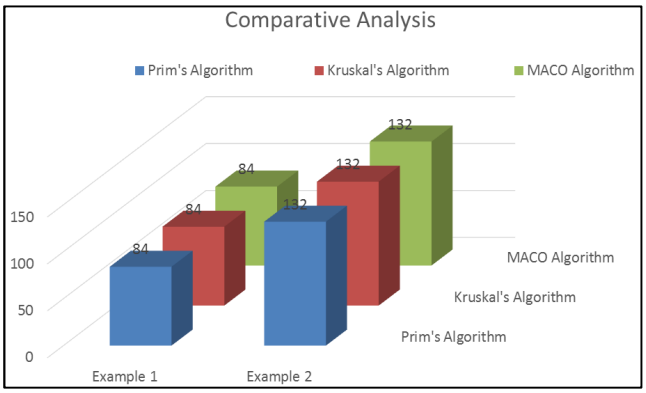

Figure 22. *Comparative Analysis of Example 1 and 2 with Prim's, Kruskal's, and MACO Algorithm.*

In above Table 1 and Figure 22 shows how this new method is efficient. Table 1 represents the comparative analysis of examples 1 and 2 with well-known Prim's and Kruskal's algorithms. When compared to Prim's and Kruskal's algorithms same results can obtain. The total weight of example 1 minimum spanning tree is 84 units and example 2 is 132 units. The graphical depiction of the comparison study is shown in Figure 22. The two examples above use different methodologies, but the heights of the bar charts are the same. An identical result may be produced using the algorithms of Prim and Kruskal.

5. Conclusion

This paper proposed a new definite algorithm for finding the MST of the weighted undirected graph G by using the ant colony algorithm on a given connected graph. The Minimum Weight Spanning Tree (MWST) begins with a single node and identifies all of its accessible nodes as well as the set of connections that link them with the least amount of weight. Prim's and Kruskal's algorithms are well-known greedy algorithms for determining the shortest-spanning tree of a given connected graph G.

The ACOA is a population-based metaheuristic for estimating solutions to complex optimization problems. By modifying ACOA and including the Transition Rule, the suggested technique has been developed. This new algorithm is easier and less complicated in its implementation compared to other existing algorithms when the graph has a lot of nodes.

We represented the newly proposed method using two illustrative examples. We can obtain the same results when compared to Prim's and Kruskal's algorithms, and the paper presented a new simple, and efficient technique to find the minimum cost-spanning tree of an undirected connected weight graph with less iteration and to examine the large-scale of the problem.

References

[1] J. Nešetřil, E. Milková, and H. Nešetřilová, "Otakar Borůvka on minimum spanning tree problem: Translation of both the 1926 papers, comments, history," *Discrete Math.*, vol. 233, no. 1–3, pp. 3–36, Apr. 2001, doi: 10.1016/S0012-365X(00)00224-7.

[2] Dey, S. Broumi, L. H. Son, A. Bakali, M. Talea, and F. Smarandache, "A new algorithm for finding minimum spanning trees with undirected neutrosophic graphs," *Granul. Comput.*, vol. 4, no. 1, pp. 63–69, 2019, doi: 10.1007/s41066-018-0084-7.

[3] Khan, A. A. Aesha, and J. Sarker, "A new algorithmic approach to finding minimum spanning tree," *4th Int. Conf. Electr. Eng. Inf. Commun. Technol. iCEEiCT 2018*, pp. 590–594, Jan. 2019, doi: 10.1109/CEEICT.2018.8628095.

[4] P. Biswas, M. Goel, H. Negi, and M. Datta, "An Efficient Greedy Minimum Spanning Tree Algorithm Based on Vertex Associative Cycle Detection Method," *Procedia Comput. Sci.*, vol. 92, pp. 513–519, Jan. 2016, doi: 10.1016/J.PROCS.2016.07.376.

[5] U. Ekanayake, W. Daundasekara, and P. Perera, "Research An Approach for Solving Minimum Spanning Tree Problem and Transportation Problem Using Modified Ant Colony Algorithm," *North Am. Acad. Res.*, vol. 3, no. 9, pp. 28–45, 2020, doi: 10.5281/zenodo.4072472.

[6] P. Ayegba, J. Ayoola, E. Asani, and A. Okeyinka, "A Comparative Study of Minimal Spanning Tree Algorithms," *2020 Int. Conf. Math. Comput. Eng. Comput. Sci. ICMCECS 2020*, no. May 2020, doi: 10.1109/ICMCECS47690.2020.240900.

[7] H. N. Gabow, Z. Galil, T. Spencer, and R. E. Tarjan, "Efficient algorithms for finding minimum spanning trees in undirected and directed graphs," *Combinatorica*, vol. 6, no. 2, pp. 109–122, Jun. 1986, doi: 10.1007/BF02579168.

[8] M.-B. Choi and S.-U. Lee, "An Efficient Implementation of Kruskal's and Reverse-Delete Minimum Spanning Tree Algorithm," *J. Inst. Webcasting, Internet Telecommun.*, vol. 13, no. 2, pp. 103–114, Apr. 2013, doi: 10.7236/JIIBC.2013.13.2.103.

[9] Colorni, A., Dorigo, M. and Maniezzo, V. (1991) Distributed Optimization by Ant Colonies. In Varela, F. and Bourgine, P., Eds., Proceedings of the European Conference on Artificial Life, ECAL'91, Paris, Elsevier Publishing, Amsterdam, 134-142. - Reference.

[10] A. Alsawy and H. A. Hefny, "Fuzzy-based ant colony optimization algorithm," *ICCTD 2010 - 2010 2nd Int. Conf. Comput. Technol. Dev. Proc.*, pp. 530–534, 2010, doi: 10.1109/ICCTD.2010.5645952.

[11] U. Jaiswal and S. Aggarwal, "Ant Colony Optimization," *Int. J. Sci. Eng. Res.*, vol. 2, no. 7, 2011.

[12] John Clark and Derek Allan Holton (1995)'A First Look At Graph Theory', published by Allied Publishers Limited, 1995 - Google Search.

[13] SAYLI and J. H. S. Alkhalissi, "Negligence Minimum Spanning Tree Algorithm," *Eur. J. Sci. Technol.*, no. November, pp. 70–76, 2018, doi: 10.31590/ejosat.386716.

[14] Paryati and K. Salahddine, "The Implementation of Kruskal's Algorithm for Minimum Spanning Tree in a Graph," *MATEC Web Conf.*, vol. 348, p. 01001, 2021, doi: 10.1051/matecconf/202134801001.

[15] E. M. U. S. B. Ekanayake, S. P. C. Perera, W. B. Daundasekara, and Z. A. M. S. Juman, "A Modified Ant Colony Optimization Algorithm for Solving a Transportation Problem," *J. Adv. Math. Comput. Sci.*, no. August, pp. 83–101, 2020, doi: 10.9734/jamcs/2020/v35i530284.

[16] S. Li, Y. Wei, X. Liu, H. Zhu, and Z. Yu, "A New Fast Ant Colony Optimization Algorithm: The Saltatory Evolution Ant Colony Optimization Algorithm," *Mathematics*, vol. 10, no. 6, p. 925, 2022, doi: 10.3390/math10060925.

[17] R. Likaj, A. Shala, M. Mehmetaj, P. Hyseni, and X. Bajrami, "Application of graph theory to find optimal paths for the transportation problem," *IFAC Proc. Vol.*, vol. 15, no. PART 1, pp. 235–240, 2013, doi: 10.3182/20130606-3-XK-4037.00031.

[18] P. S and M. K. M, "Application Of Graph Theory To Find Optimal Path And Minimized Time For The Transportation Problem," *Int. J. Sci. Res. Publ.*, vol. 11, no. 1, pp. 278–285, 2020, doi: 10.29322/ijsrp.11.01.2021.p10929.