The Shortest Path Tree Problem with Bandwidth Constraint in the Computer Networks

Sai Swaroop Madugula, Emad Saleh, Hamzih Alsmadi, Farah Shahata

Abstract: **The communication applications in data networks require to transmit the data from many sources to multiple destinations. In order to make this transmission efficient we need to use the shortest paths during these communications between the source and the destination. Therefore, the shortest paths tree problem is evaluated regarding to the bandwidth constraints on data networks to find minimal cost between the sender and the receiver nodes. To overcome this problem, we proposed the famous optimization tool, that is Genetic Algorithm. This paper focuses on the Genetic Algorithm to control the problem of finding the shortest path tree with minimum cost and bandwidth constraint. Moreover, a connection matrix of a specific network is applied, and the cost, bandwidth matrices are randomly generated by the algorithm. The proposed algorithm aims to find out the group of edges that consists of links to all nodes, value of bandwidth and the summation of the cost's restrictions. The proposed algorithm has been used in different sample networks to illustrate its efficiency.**

Keywords—Shortest Path Problem, Genetic Algorithm, Shortest Paths Tree, Dijkstra's Algorithm.

I. INTRODUCTION

The definition of the shortest path problem is the problem of finding a path between two vertices (or nodes) in a graph such that the sum of the weights of its component edges is minimized. The real-world conventional optimization problem is the shortest problem of several types of networks. The shortest path problem has been applied universally in various applications such as data network, computer networks, engineering management, etc. Suppose we have shortest path tree rooted from the vertex s and the spanning tree T of G. This path distance has been from root v to any other vertex u in T where the shortest path distance from v to u in G. So, to find the optimal shortest path tree, the proposed algorithm will be applied in a scenario of single link failure. To determine the Shortest Best Path Tree (SBPT) the proposed algorithm will be used in the case of a multicast tree. Furthermore, to determine the shortest path trees the labeling techniques are used. The genetic algorithm also is proposed to sort out MCR issues with respect to transmission success and transmission delay ratio.

Other algorithms like heuristic algorithm will not be active especially If state information related to network exists in a dynamic network environment for the real-time applications. In Huang and Wang, [1] they mentioned the genetic algorithm (GA) it will solve the shortest path. Other stated that the Oriented Spanning Tree (OST) based algorithm used for sorting out Multi-criteria Shortest Path Problem (MSPP) and the Multi-criteria Constrained Shortest Path Problems (MCSPP), [4]. Also, the Oriented Spanning Tree (OST) based algorithm used for sorting out Multi-criteria Shortest Path Problem (MSPP) and the Multi-criteria Constrained Shortest Path Problems (MCSPP), [4]. On the other hand, our genetic algorithm (GA) is used to estimate the low-cost of the multicasting tree with bandwidth and delay constraints, [1].

In this paper the genetic algorithm is applied to evaluate shortest path tree with the bandwidth constraint. The network also consists of connection and cost matrices. The genetic operations are used to find the shortest cost path by having the source node s, which can calculate and create a minimum cost path tree with bandwidth constraints.

The purpose of the genetic algorithm is finding the minimal cost shortest paths tree and solve the problem subject to bandwidth constrained and hop limited. Also, the aim from the algorithm is to search the optimal set of edges connecting all nodes such that the costs are minimized, the bandwidth is constrained, and hop is limited.

II. LITERATURE REVIEW AND RELATED WORKS.

The problem at hand is find the shortest path from a given network. In this paper, we are going to discuss the working of an algorithm which can generate the shortest paths tree in a bandwidth and minimum cost scenario. There are a lot of related works and papers which follow different methods to find the shortest paths. Dijkstra's algorithm is one of the most famous algorithms which is used to select the next shortest simple path. Several other papers, like [2] have been implemented to find the shortest path. Hershberger et al. [2] divides the previous shortest paths into different equivalence classes, then the candidate paths in each equivalence class can be found by a replacement path-based algorithm. Finally, the shortest path is selected among the candidate paths. Related studies like [1] have implemented a method by means of random fuzzy simulation and a new

repair-based genetic optimization. They have efficiently searched for shortest paths by using a mix of uncertainty of randomness and fuzziness.

While some papers use the genetic algorithms to find shortest paths of a given network, some studies like [1] use the classical K shortest paths problem, which identifies the k shortest paths in a directed graph. This problem is used to provide alternative paths for vehicle routing services and several other services. Finding the top-k general shortest paths are studies in several paper, where undirected graphs and cycles in paths are allowed. However, some of the studies do not consider the path diversity. There are a lot of studies which have implemented several algorithms for the multi-constrained shortest path problem but very few have good practical performance in case of two or more constraints [4].

Many other studies have also studied the genetic algorithms and have implemented them to solve the minimal cost shortest paths tree problem subject to bandwidth constraint and hop limited network. For instance, in one of the studies, the genetic algorithm (GA) is used to evaluate the low-cost multicasting tree with the bandwidth and the delay constraints. Another study consists of a genetic algorithm which is used to determine the k shortest path with bandwidth constraints from source to multiple destination nodes.

III. SHORTEST PATH PROBLEM.

In laymen's terms, the shortest path problem is about computing a path between nodes of a network such that the total sum of edges weights is minimum. However, the approach to the shortest path problem is realized in different forms and therefore several algorithms were introduced to find the shortest path in several different scenarios.

The reason that several shortest path algorithms exists is because there are different types of networks and the several requirements of the said network present during transmission. For example, consider A is a source node and B as a destination node. Now shortest path might mean the shortest path between path between point A and B or alternatively it might also mean the shortest path between the node A and all other points in the graph.

Generally, shortest path problem algorithms are distinguished into two types. The first type is known as the single-source shortest path algorithm. As the name suggest, this type of problem consists of a single source node and the problem is to find the shortest paths from the source node to all other node in the network. In order to solve this problem, several algorithms have been introduced and Dijkstra's algorithm is the most commonly known algorithm for solving the single-source shortest path problem with non-negative edge weights. Dijkstra's algorithm is based on the principle of relaxation, in which approximation to the correct distance is gradually replaced by more accurate values until shortest distance is reached. In this algorithm, we start with a minimum value which is an overestimation of the true distance between source and destination node. After taking this value, it is replaced with the minimum distance of the newly found. It makes use of a priority queue and selects the closest vertex that has not yet been processed and performs the process on all its outgoing edges.

The second type of shortest path problem is known as the All-pairs shortest path problem. In this we must find the shortest paths between every pair of nodes present in the network. The most commonly used algorithm to solve this problem is the Flyd-Warshall algorithm. This algorithm concerns itself with the task of finding the shortest paths with both positive or negative edge weights. The output of this algorithm is the lengths of shortest paths between all pairs of vertices. This algorithm uses the dynamic programming approach to do so.

For our purpose, we are taking a duplex link network in which information is transferred both ways. Each link consists of a specific bandwidth and cost. As mentioned earlier in the paper, our algorithm will be used to determine the Shortest Best path tree in case of multicast tree.

IV. PROPOSED ALGORITHM.

The proposed method will consider the cost, connection and bandwidth matrix of the network. This algorithm will determine the minimum cost path tree with bandwidth constraint at source node. We will make use of 'chromosomes' values in our algorithm to compute the best path. The chromosome can be defined as an element which refers to a specific node present in the network. Each chromosome consists of at least two non-zero elements. A chromosome is a binary string of length N and indicates each path in the network.

In the beginning of computation, we will set a random value as the initial population. This value is used to produce the chromosome which are required to compute shortest path for each node. First, we generate a chromosome randomly and associate it with the candidate path (which is the path to traverse). We repeat the above step until the chromosomes produced are equal to the initial population. Then we use the objective function of the algorithm. This function compares the solutions and evaluates the best one among them. The bandwidth and cost of the candidate path are used as the main parameters for the objective function. The objective function has two main criteria for the computation of the candidate path. The first criteria are that the chromosome should consist of two non-zero elements. The second criteria are that it should consists of connected candidate path and each node connects one another.

After the criteria are met and the objective function computes the candidate paths, we apply a genetic crossover operation on the network. In this, we generate a new offspring from the chromosomes and to generate a new offspring from two parents a single cut point crossover should be applied. Once the crossover ratio (Pc=0.90) is verified, the crossover process will be run, and a cut point will be selected at random.

Finally, the genetic mutation operation is applied in which there a bit by bit operation. We initially set a value for the mutation ratio (Pm). This ratio will be used to verify and run the mutation process. The point to be mutated is also selected randomly.

The algorithm used to compute the shortest paths tree is depicted below:

```
Input: Initialise parameters: Pop_Size, Max_Gen, Crossover Ratio.
Output: Minimum-Cost paths tree with bandwidth constraint.
Step 1: Set j=2, as the destination node.
Step 2: Generate initial population according to the process explained earlier.
Step 3: Gen<-1.
Step 4: While (gen <= max_gen) do {
        P <- 1.
        While (P <= pop_size) do {
        Apply Genetic operations to acquire new population
        Apply Crossover according to Crossover value.
        Apply Mutation according to Mutation ratio.
        Compute the total cost and bandwidth of the candidate path.
        P <- P + 1.}
        Set gen = gen+1
        If gen is greater than Max_Gen then stop}
Step 5: Save the candidate path for the destination j that has the minimum cost and bandwidth constraint.
Step 6: Set j = j+1.
Step 7: Repeat the entire process depicted above until the value of j is equal to number of nodes.
Step 8: Print the shortest paths to destination nodes.
```

Fig. 1. Genetic Algorithm (Pseudocode).

V. METHODOLOGY AND IMPLEMENTATION

For our implementation, we have used the C++ platform to create a program which calculates multiple shortest paths for a given number of nodes. The program inputs the number of nodes and the connection matrix and it will generate both cost matrix and bandwidth matrix. The costs and bandwidth of the links are generated randomly. Then, by using the above explained algorithm, the program calculates multiple shortest paths to the destination nodes and outputs the path along with the cost and bandwidth of that link. In order to test the efficiency of our algorithm, we have taken three scenarios which consists of different number of nodes. This algorithm will be run in the scenario of single link failure. Each link is a duplex link meaning information can be transferred from both directions.

In our program, the initial values of parameters are size of the population, maximum generation, Pc and Pm. We give each of these parameters an initial value to begin with. The size of the population is set to 25 and maximum generation will be equal to 300. Also, Pc is set to 0.90 and Pm is set to 0.02.

We will consider a network with 3 scenarios:

1) Network with 8 nodes.
2) Network with 11 nodes.
3) Network with 15 nodes.

Let us consider the first case where the number of nodes in the network are equal to eight:

For this network, we have eight nodes and for the program to calculate shortest path, we need to specify the connection matrix first. Connection matrix is nothing but adjacency matrix. If there a link between node 1 and node 2 then the value at (1,2) and (2,1) in the connection matrix is set to 1. If there is no link, then it is set to 0. The program reads the number of nodes and the connection matrix from a text file. The number of nodes must be equivalent to the total size of the matrix i.e. if the network has eight nodes, then the connection matrix is of size 8x8. Then, after reading the connection matrix, our program randomly generates costs for each link within the range of 0-12. Then, the bandwidth matrix is also generated based on the cost matrix, and the destination nodes are set. After the above steps are completed, we execute the genetic algorithm and calculate the shortest paths in the network.

The operation below describes the calculation of the shortest path for a network consisting of eight nodes.

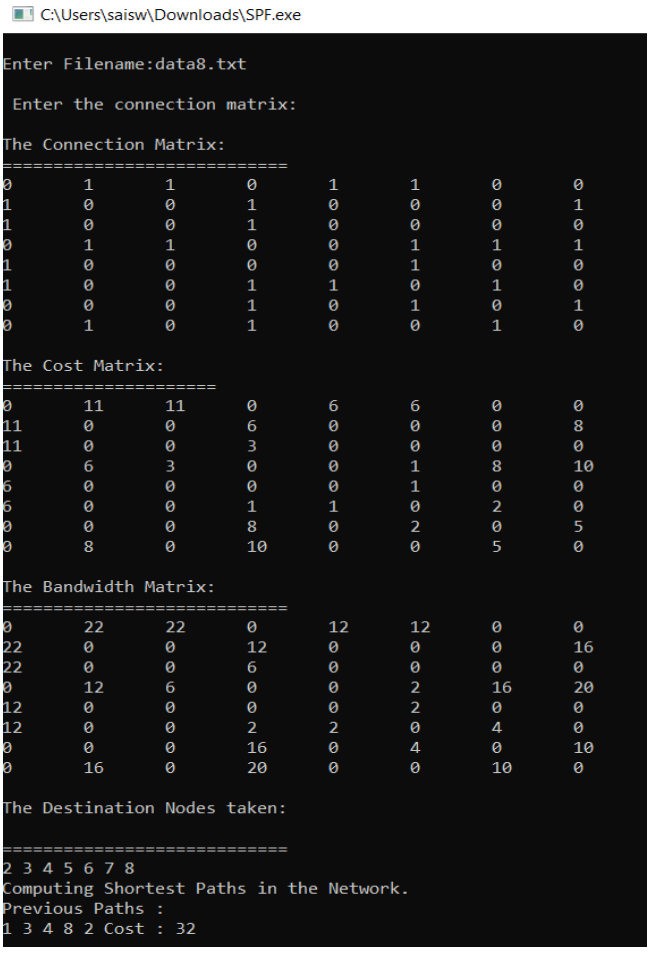

Fig. 2. An eight-node network's connection matrix.

In figure 2, we can observe that the program opens the file "data8.txt" and reads the number of nodes and the connection matrix. Then it generates a random cost matrix and bandwidth matrix. The number of destinations is set to destination nodes = nd-1 where Nd is the number of nodes in the network. Then genetic algorithm is applied to the matrices.

Now consider the case where there are 11 nodes in the network. The same procedure is followed as mentioned earlier. The connection matrix will be of size 11x11. The process of calculating the cost and the bandwidth matrix is the same for any scenario of the network. The bandwidth matrix is generated by multiplying the cost matrix with the value 2. Since there are eleven nodes in the network, the destination nodes are set to 10 and will range from node 2 to node 11 with node 1 being the source node. All the shortest path computations will be done through the source node 1.

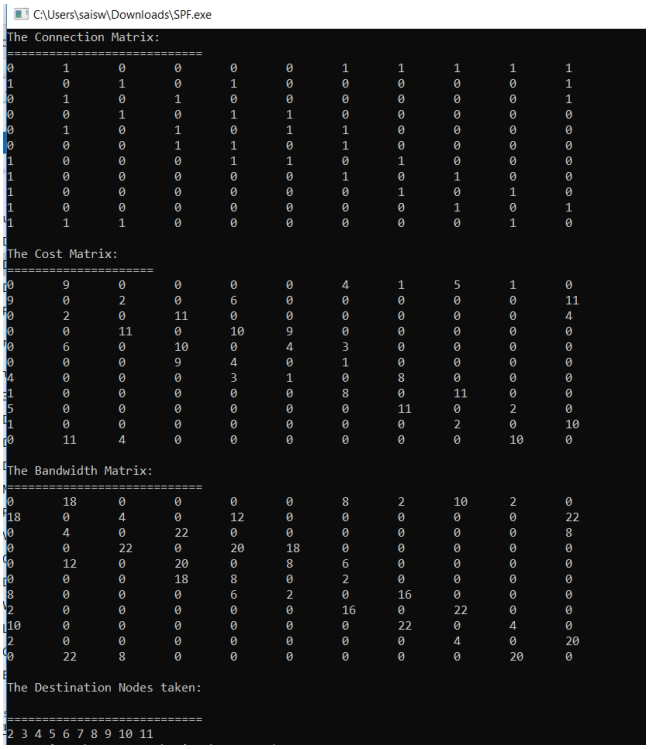

Fig. 3. An eleven-node network's connection matrix

The initial parameters are the same for any number of nodes in the network. The number of destination nodes change whenever the number of nodes in the network are changed. But it should be noted that the number of destination nodes is always less than the total number of nodes present in the network.

Below is the depiction of a 15-node network:

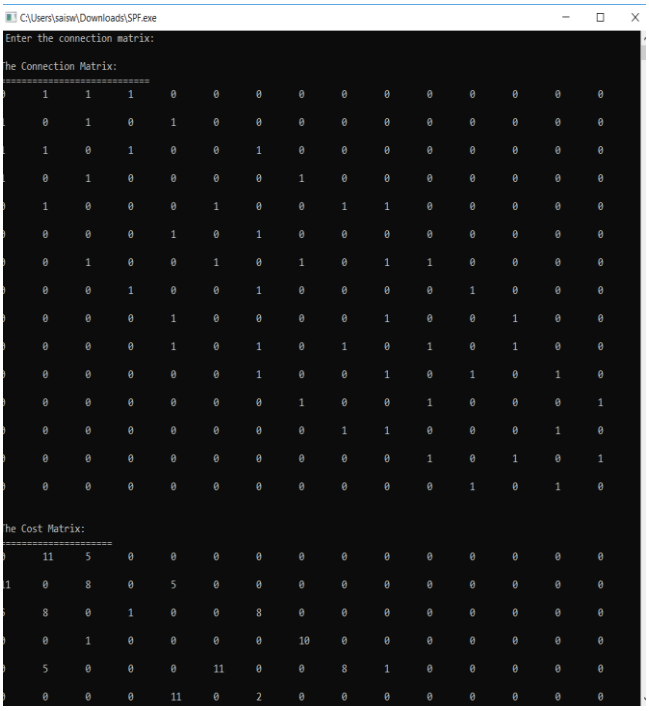

Fig. 4. An fifteen-node network's connection matrix.

VI. SIMULATION RESULTS

We have taken a total of three scenarios to test our algorithm to find the shortest paths tree. The program is coded in C++. And in addition to this, the graphical representation of the shortest paths tree is done on Network Simulator 2 (NS2) and network animator (NAM).

The first scenario consists of eight nodes in the network. The connection matrix is depicted above in figure 2. Keeping the source node as 1, our algorithm computes the shortest paths to every node in the network. The eight-node network is represented as a tree with the help of network simulator 2 tool which supports simulation of several wired and wireless networks. The graphical representation of the tree is below.

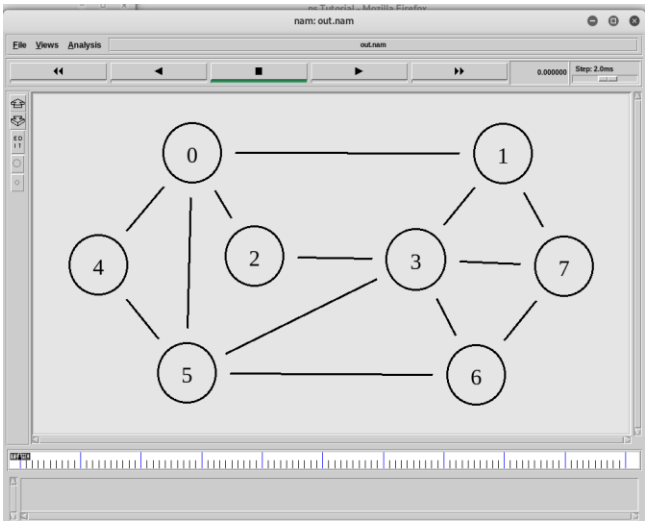

Fig. 5. Eight Node Network.

Below is the final output of the programs which displays multiple shortest paths present in the networks.

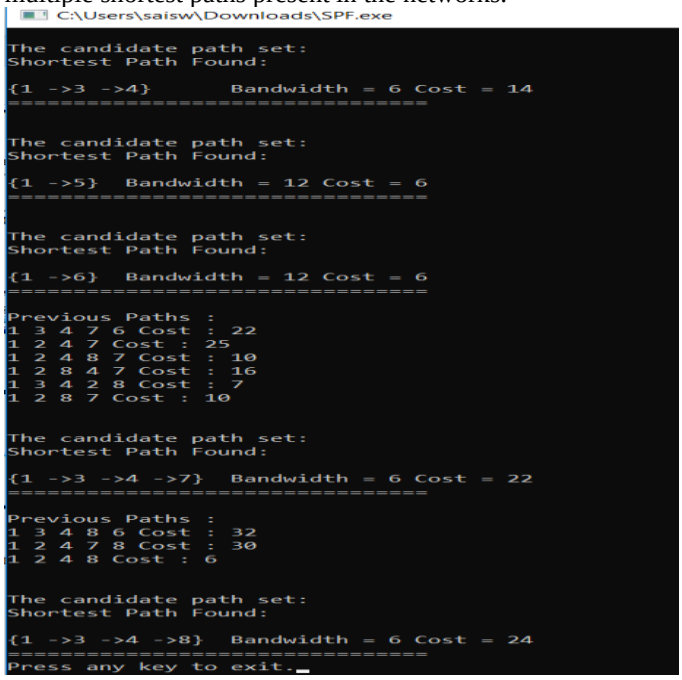

Fig. 6. Printing Shortest paths from source node 1 to multiple destinations.

We can observe in the figure 6 that the program prints all the shortest paths from source node 1. Along with the printed paths, the bandwidth and the cost of the path is also printed. From the final output (Figure 6), which are all the shortest paths, we generate the shortest paths tree (using NS2) which uses the minimum cost in a bandwidth constrained network. NS2 uses TCL scripts to run network animator.

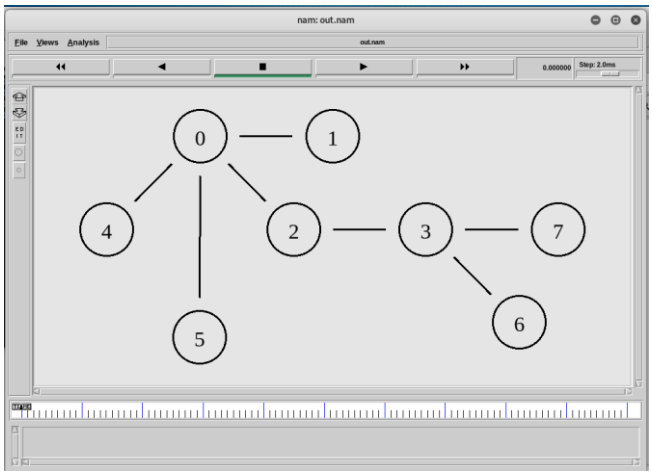

Fig. 7. Shortest Paths Tree (8-node network).

As we can see in figure 7, our genetic algorithm produces the best shortest paths from the source node 0 to every destination node up to node 7.

The second scenario consists of eleven nodes in the network. The genetic algorithm is applied to this network. The connection matrix is the same matrix mentioned in the figure 3.

Now, the graphical representation of the network is depicted below:

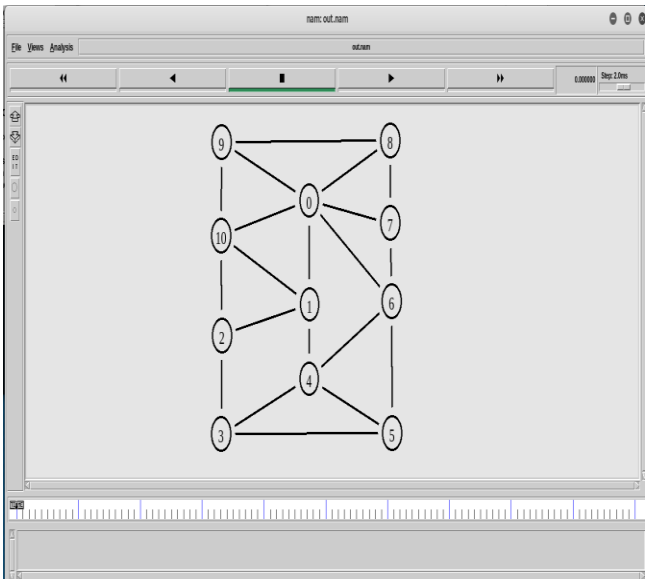

Fig. 8. Shortest Paths Tree (11-node network).

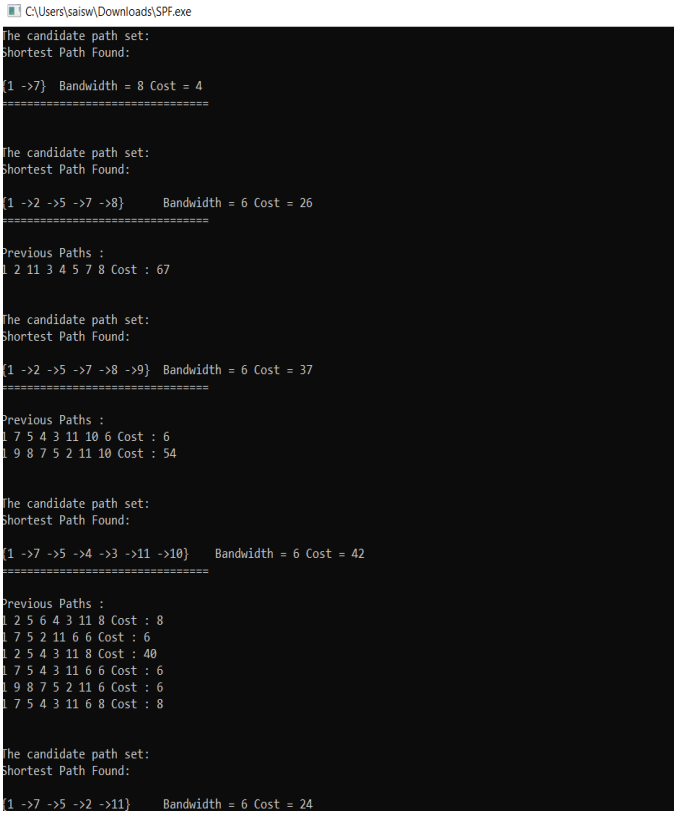

Fig. 9. Printing Shortest paths from source node 1 to multiple destinations in an 11-node network.

As we can observe from figure 9, the program also displays the previous paths which are tested by the chromosomes. These paths are all tested before setting the shortest path which has the minimum cost and bandwidth. From the above output, we can derive the shortest paths tree as depicted below:

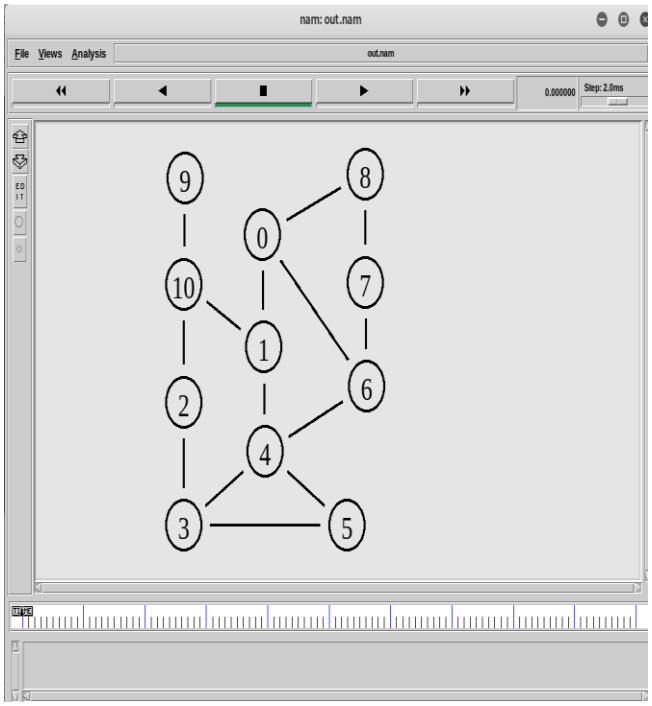

Fig. 10. Shortest Paths Tree for an 11-node network.

The same process is applied for the third scenario also which is a 15-node network.

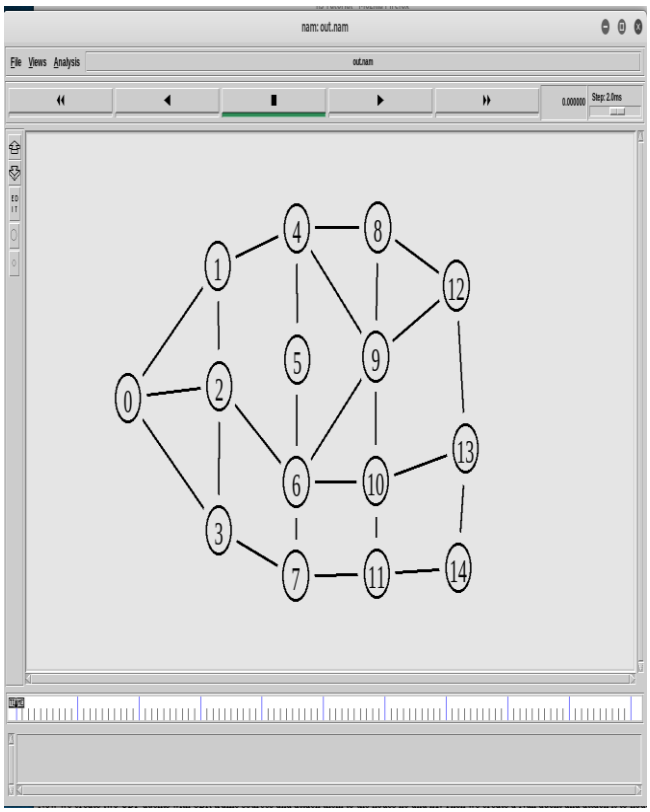

Fig. 11. 15-node network.

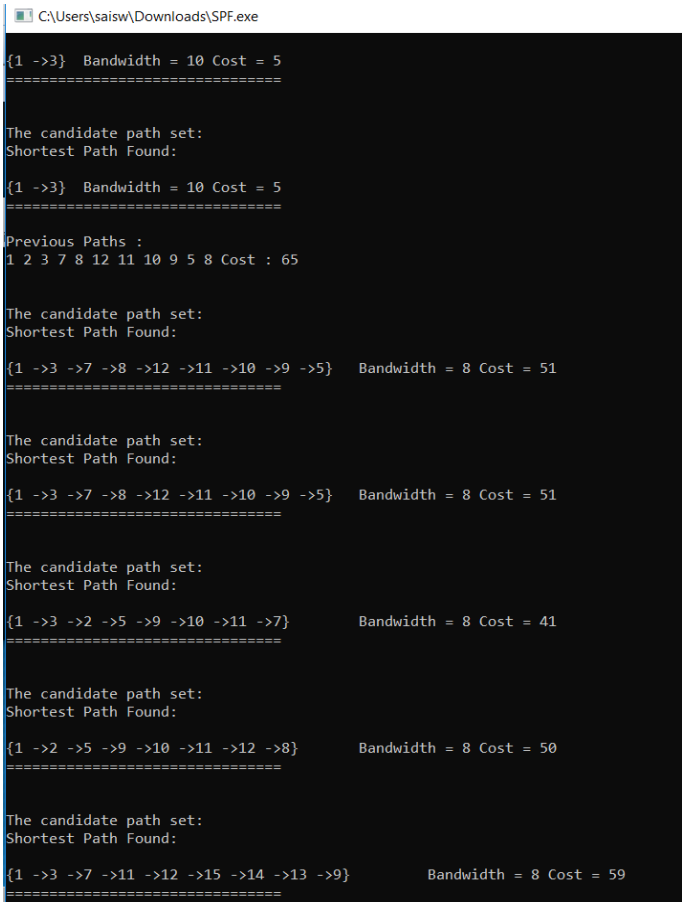

Fig. 12. Printing Shortest paths from source node 1 to multiple destinations in a 15-node network.

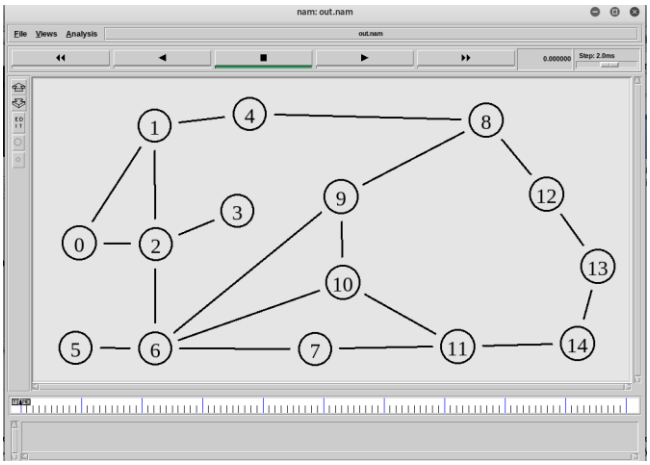

Fig. 13. Shortest Paths Tree for a 15-node network.

We have also compared the efficiency of our algorithm against the Dijkstra's Shortest path problem Algorithm. The scenario here is an eight-node network for both algorithms. The connection matrix or the adjacency matrix of both the algorithms are same and is equal to connection matrix in figure 2. The costs between the links are also derived from the cost matrix of figure 2. The Dijkstra's algorithm is run, and the output is as obtained below:

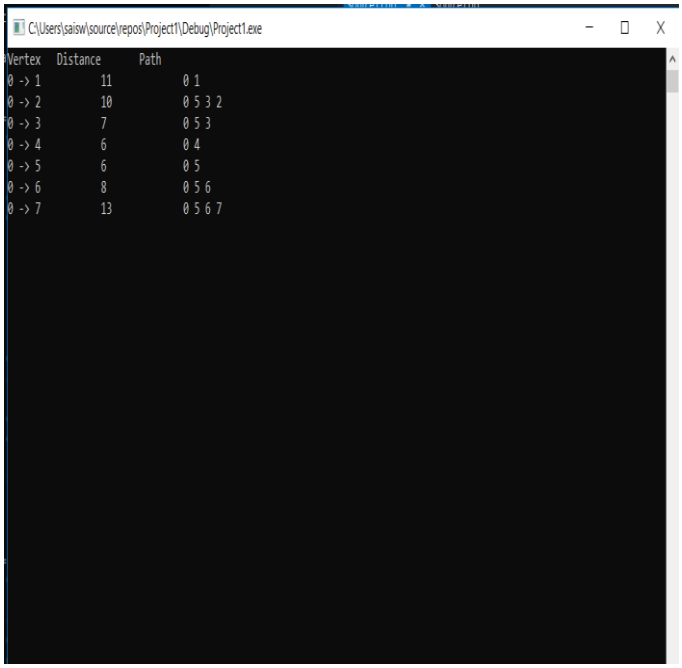

Fig. 14. Output of shortest paths for Dijkstra's Algorithm.

We can observe in the above figure 14 that the source node 0 is taken and the shortest distance is computed from source to every other destination node. The cost or the distance is calculated and displayed. Now if we compare this figure to the figure 7 of the paper, we can observe that the costs of the paths are almost the same value. But we must keep in mind that the Dijkstra's algorithm does not consider the bandwidth constraint whereas the genetic algorithm computes the shortest paths with the both minimum cost and bandwidth constraint.

Below is the computed shortest paths tree of the figure 13 using ns2 and Nam:

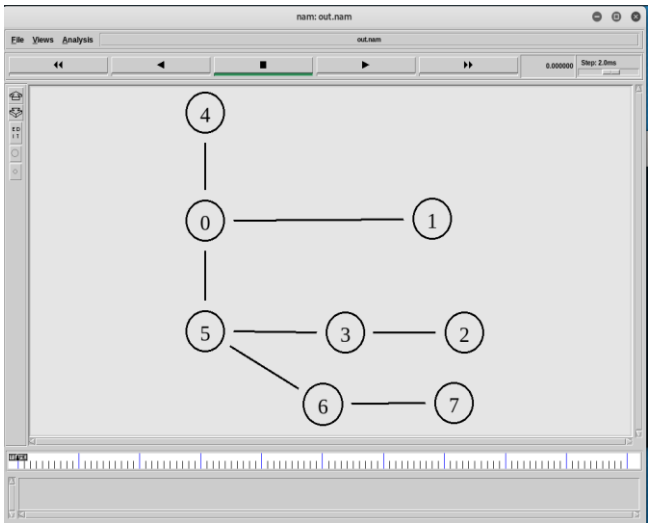

Fig. 15. Output of shortest paths for Dijkstra's Algorithm.

VII. CONCLUSION

The scope of the paper is to solve the optimal minimum cost path tree problem and the Genetic Algorithm optimization tool will address this issue. The proposed algorithm considers the cost and connection matrixes of the network. Furthermore, it can compute the minimum cost paths from root node s. The algorithm has been applied in three sample networks which have different number of nodes. The results also show the performance and efficiency of the proposed Genetic algorithm over the conventional Dijkstra's algorithm. The future work of this paper would be to solve the multi constrained paths tree problem.

VIII. REFERENCES

[1] W. Huang and J. Wang, "The Shortest Path Problem on a Time-Dependent Network with Mixed Uncertainty of Randomness and Fuzziness," IEEE Transactions on Intelligent Transportation Systems, vol. 17, no. 11, pp. 3194–3204, 2016.

[2] H. Liu, C. Jin, B. Yang, A. Zhou, "Finding Top-k Shortest Paths with Diversity." IEEE Transactions on Knowledge and Data Engineering, Vol. 30(3), pp.488-502, 2018.

[3] A. Frechette, F. B. Shepherd, M. K. Thottan, and P. J. Winzer, "Shortest Path Versus Multihub Routing in Networks with Uncertain Demand," IEEE/ACM Transactions on Networking, vol. 23, no. 6, pp. 1931–1943, 2015.

[4] G. Feng and T. Korkmaz, "Finding Multi-Constrained Multiple Shortest Paths," IEEE Transactions on Computers, vol. 64, no. 9, pp. 2559–2572, 2015.

[5] J. Cota-Ruiz, P. Rivas-Perea, E. Sifuentes, and R. Gonzalez-Landaeta, "A Recursive Shortest Path Routing Algorithm with Application for Wireless Sensor Network Localization," IEEE Sensors Journal, vol. 16, no. 11, pp. 4631–4637, 2016.

[6] M. S. Talebi, Z. Zou, R. Combes, A. Proutiere, and M. Johansson, "Stochastic Online Shortest Path Routing: The Value of Feedback," IEEE Transactions on Automatic Control, vol. 63, no. 4, pp. 915–930, 2018.

[7] A. Younes, Usama A. Badawi, T. H. Farag, A. Ben Salah and Fahad. A. Alghamdi, "The Shortest-Path Broadcast Problem", International Journal of Applied Engineering Research, Vol. 13, Number 10, 2018, pp. 7580-7584.