Finding the Shortest Route for Kosovo Cities Through Dijkstra's Algorithm

Senad Orhani

Department of Education, Faculty of Education, University of Prishtina "Hasan Prishtina", Prishtina, Kosovo

ANNOTATION

Kosovo is a beautiful country with a strong cultural heritage which has a very rich nature. Kosovo is not only mountainous, but also has scenic views and other attractions of high cultural importance. Although a small country, Kosovo has many destinations to see and visit. Therefore, to visit these cities we try to choose the shortest route between these locations. But this is probably very difficult to do manually. For this reason, the purpose of this study is to find the shortest path between the two cities of the Republic of Kosovo through Dijkstra's algorithm. The mentioned cities are in both ends of the country, where the starting point is the city of Gjakova and the destination is the city of Gjilan. Dijkstra's Algorithm basically starts at the node that you choose and it analyzes the graph to find the shortest path between that node and all the other nodes in the graph. The results of this study showed the importance of Dijkstra's Algorithm, where by measuring the road distance through Google Maps, we came to the conclusion in calculating the shortest route between these cities.

KEYWORDS: *Dijkstra's Algorithm, graphs, Kosovo, node, and shortest path.*

1. INTRODUCTION

The long journey in most cases is tedious, we usually look for a shorter route between the two locations. With the advancement of ICT, we have the opportunity to use Google Maps to do our planning of which route we will travel. This is feasible thanks to some algorithms with which a shorter route can be found to travel. These computer applications use representations of the street maps as graphs, with estimated driving times as edge weights. The purpose of this study is to find the shortest path between the two cities of the Republic of Kosovo through Dijkstra's algorithm. The mentioned cities are in both ends of the country, where the starting point is the city of Gjakova and the destination is the city of Gjilan. The road between these cities will pass through other settlements, where by means of the algorithm in question we are able to determine their nodes and their weight. In Dijkstra's algorithm, this means the edge has a large weight--the shortest path tree found by the algorithm will try to avoid edges with larger weights.

Well simply explained, an algorithm that is used for finding the shortest distance, or path, from starting node to target node in a weighted graph is known as Dijkstra's Algorithm. This algorithm makes a tree of the shortest path from the starting node, the source, to all other nodes (points) in the graph. Dijkstra's algorithm is the iterative algorithmic process to provide us with the shortest path from one specific starting node to all other nodes of a graph. It is different from the minimum spanning tree as the shortest distance among two vertices might not involve all the vertices of the graph. It is important to note that Dijkstra's algorithm is only applicable when all weights are positive because, during the execution, the weights of the edges are added to find the shortest path [1]. The purpose of Dijkstra's algorithm is not limited to just getting the shortest distances [2].

2. MATERIALS AND METHODS

In mathematics, we have a saying that the shortest path is a straight line between two points. This may be true in a simple and abstract situation. However, the straight path may be impossible in real life, and a web of networks may be given as alternative routes. How do we determine the shortest or smallest sum of the collective paths from a starting point to its destination? The shortest path is the route that requires the smallest distance travelled. If this entails a web of paths, the shortest path is the one that gives the smallest total distance travelled from the initial to the terminal location [2].

Shortest path algorithms are used to find from the source city to destination city. Shortest path problem is the finding a path between two vertices (or nodes) such that the sum of the weights of its constituent edges is minimized. An example is finding the quickest way to get from one location to another on a road map, the vertices represent locations and of road and are weighted by the time needed to travel that segment [3].

One of the classic and most used algorithms for calculating the shortest path from an origin to a destination is Dijkstra's algorithm, it was first enunciated by Edsger Wybe Dijkstra (1959) and is one of the most used and discussed algorithms in the literature of graphs, the temporal complexity is $O(|E| + |V|log|V|)$, where $|E|$ is the number of edges and $|V|$ is the number of vertices of the graph. However, this algorithm is not efficient for searching shortest path in large graphs [4]. For a given source node in the graph, the algorithm finds the shortest path between that node and every other. It can also be used for finding the shortest paths from a single node to a single destination node by stopping the algorithm once the shortest path to the destination node has been determined. For example, if the nodes of the graph represent cities and edge path costs represent driving distances between pairs of cities connected by a direct road, Dijkstra's algorithm can be used to find the shortest route between one city and all other cities [5].

Pseudocode [5]***:***

```
function Dijkstra(Graph, source):

    for each vertex v in Graph.Vertices:
        dist[v] ← INFINITY
        prev[v] ← UNDEFINED
        add v to Q
    dist[source] ← 0

    while Q is not empty:
        u ← vertex in Q with min dist[u]
        remove u from Q

        for each neighbor v of u still in Q:
            alt ← dist[u] + Graph.Edges(u, v)
            if alt < dist[v] and dist[u] is not INFINITY:
                dist[v] ← alt
```

```
prev[v] ← u

return dist[], prev[]
```

Dijkstra's Algorithm procedure [6]:

- Dijkstra's Algorithm basically starts at the node that you choose (the source node) and it analyzes the graph to find the shortest path between that node and all the other nodes in the graph.
- The algorithm keeps track of the currently known shortest distance from each node to the source node and it updates these values if it finds a shorter path.
- Once the algorithm has found the shortest path between the source node and another node, that node is marked as "visited" and added to the path.
- The process continues until all the nodes in the graph have been added to the path. This way, we have a path that connects the source node to all other nodes following the shortest path possible to reach each node.

Dijkstra’s algorithm is an optimal algorithm, meaning that it always produces the actual shortest path, not just a path that is pretty short, provided one exists. This algorithm is also efficient, meaning that it can be implemented in a reasonable amount of time. Dijkstra’s algorithm takes around V2 calculations, where V is the number of vertices in a graph. A graph with 100 vertices would take around 10,000 calculations. While that would be a lot to do by hand, it is not a lot for computer to handle. It is because of this efficiency that your car’s GPS unit can compute driving directions in only a few seconds [7].

With Dijkstra's Algorithm, you can find the shortest path between nodes in a graph. Particularly, you can find the shortest path from a node (called the "source node") to all other nodes in the graph, producing a shortest-path tree. This algorithm is used in GPS devices to find the shortest path between the current location and the destination. It has broad applications in industry, specially in domains that require modeling networks [6].

One of the main advantages of it is its little complexity which is almost linear. It can be used to calculate the shortest path between a single node to all other nodes and a single source node to a single destination node by stopping the algorithm once the shortest distance is achieved for the destination node. It only works for directed-, weighted graphs and all edges should have non-negative values [1].

Dijkstra's algorithm has many applications [8]:

Digital Mapping Services like Google Maps - Suppose yo want to travel from on city to another city. You use Google maps to find the shortest route. How will it work? Assume the city you are in to be the source vertex and your destination to be another vertex.

Designate Servers: In a network, Dijkstra's Algorithm can be used to find the shortest path to a server for transmitting files amongst the nodes on a network.

IP Routing: Dijkstra's Algorithm can be used by link state routing protocols to find the best path to route data between routers.

3. RESULTS

3.1. Road Gjakova – Gjilan with Dijkstra’s algorithm

In this section we want to find the shortest path between the cities of the Republic of Kosovo from the map obtained with Google Map with a starting point and a destination. Dijkstra’s Algorithm initially marks the distance (from the starting point) to any other city on the map with infinity. This is not done to imply that there is an infinite distance, but to point out that those cities have not been visited yet.

The problem of finding the way from Gjakova to Gjilan, the cost of the search is the time taken from the search and finding the way as short as possible. To solve the route as short as possible we will use the above algorithm presented in the fourth level of finding the required city. First, we present the map of Kosovo:

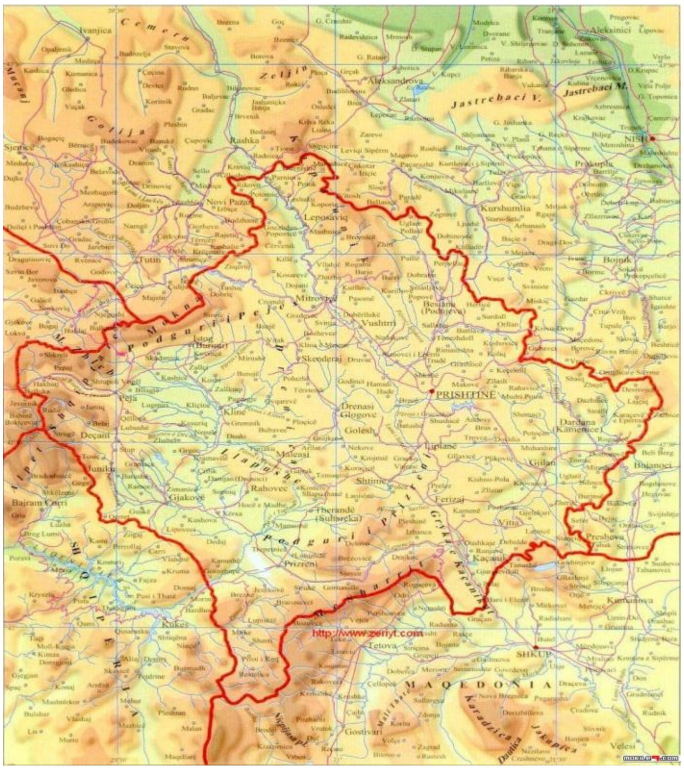

Figure 1. Kosovo Map

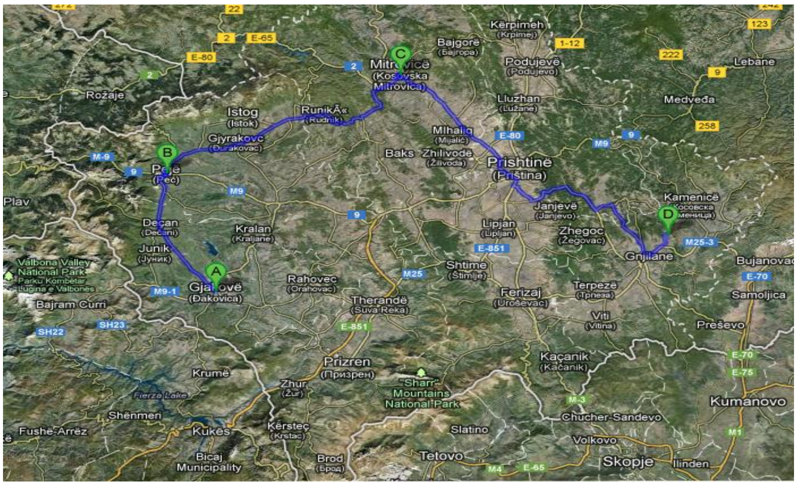

Figure 2. Road Gjakova - Peja - Mitrovica - Gjilan

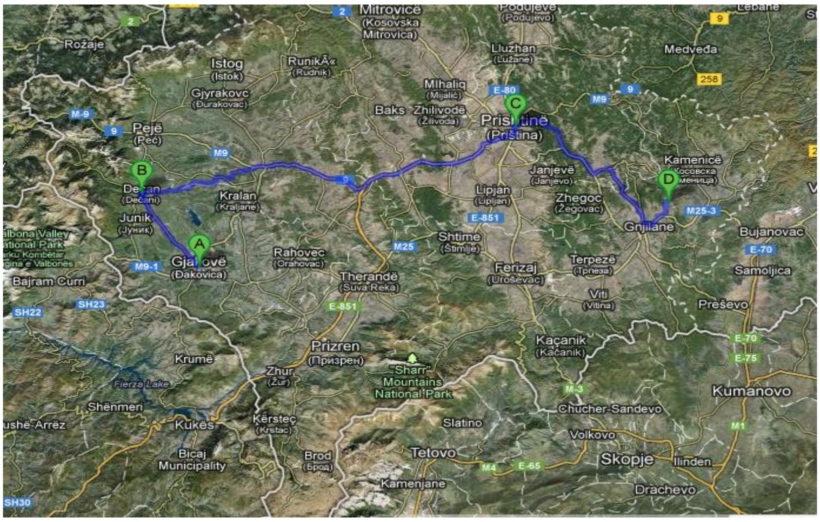

Figure 3. Road Gjakova - Deçan - Prishtina - Gjilan

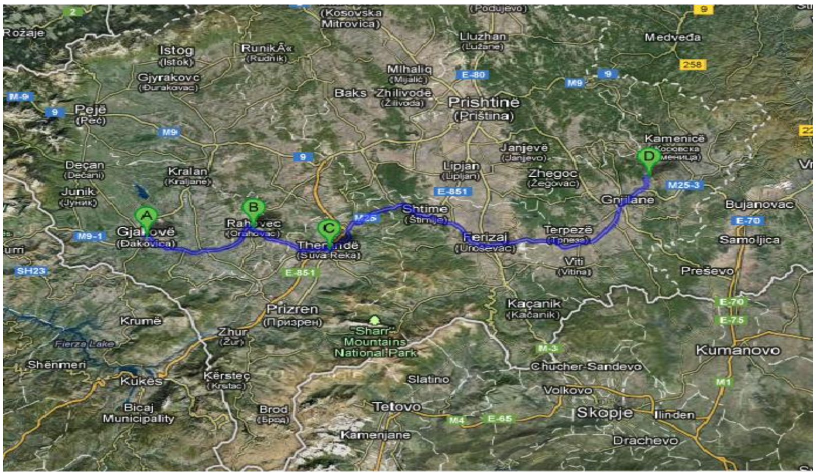

Figure 4. Road Gjakova - Rahovec - Theranda - Gjilan

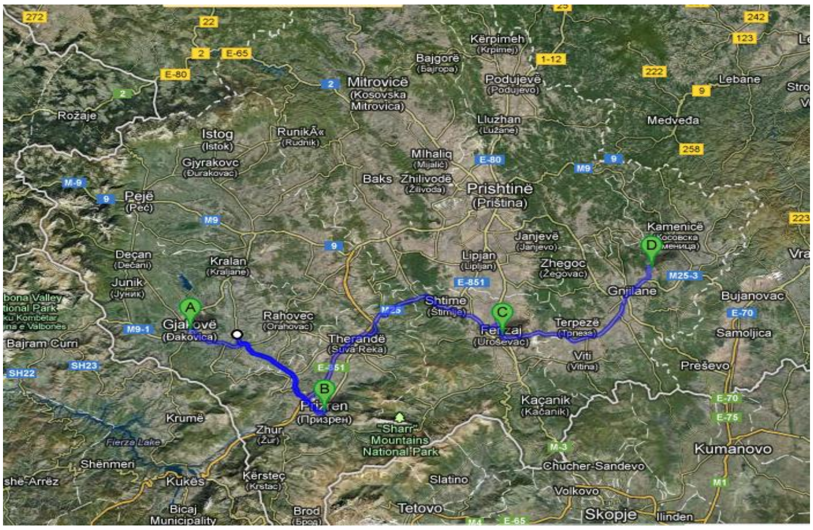

Figure 5. Road Gjakova - Prizren - Ferizaj – Gjilan

3.2. Distance between cities of Kosovo

The distance between the cities from the starting point from Gjakova to Gjilan are presented in the figure:

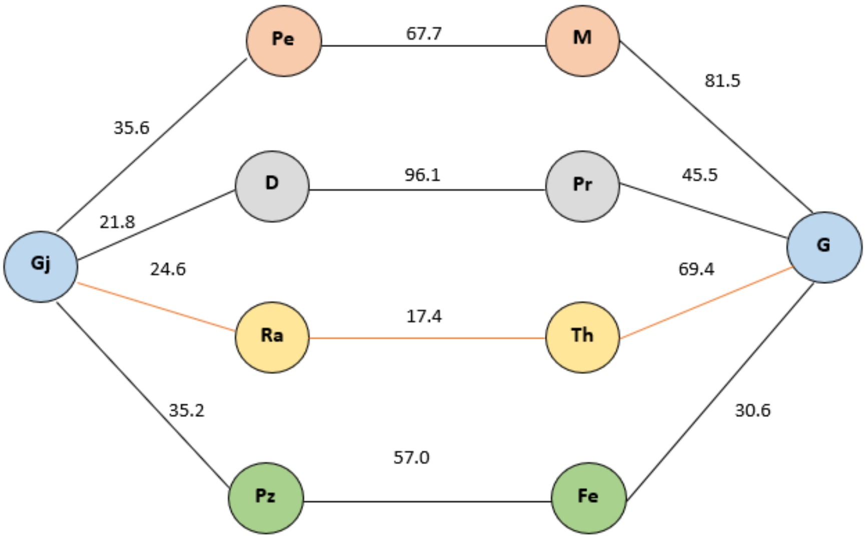

Figure 6. Distance between cities of Kosovo according to Google Maps

- Gjakova --- Gj
- Gjilani --- G
- Prizreni --- Pz
- Peja --- Pe
- Prishtina --- Pr
- Mitrovica --- M
- Ferizaji --- Fe
- Deçani --- D
- Rahoveci --- Ra
- Theranda --- Th

The shortest route calculation:

1. $\text{Gj} \xrightarrow{35.2} \text{Pz} \xrightarrow{57.0} \text{Fe} \xrightarrow{30.6} \text{G} = 122.8$ km
2. $\text{Gj} \xrightarrow{24.6} \text{Ra} \xrightarrow{17.4} \text{Th} \xrightarrow{69.4} \text{G} = 111.4$ km
3. $\text{Gj} \xrightarrow{21.8} \text{D} \xrightarrow{96.1} \text{Pr} \xrightarrow{45.5} \text{G} = 163.4$ km
4. $\text{Gj} \xrightarrow{35.8} \text{Pe} \xrightarrow{67.7} \text{M} \xrightarrow{81.5} \text{G} = 185.0$ km

As the above results show according to the first road from Gjakova to Gjilan is 122.8 km, passing through Prizren, Ferizaj to the destination. The second road shows the distance between the cities of 111.4 km, passing through Rahovec, Theranda reaching Gjilan. Whereas, the results of the third road show that the road from Gjakova to Gjilan is 163.4 km long, passing through Deçan and Prishtina. On the other hand, the road from Gjakova to Gjilan passing through the city of Peja and Mitrovica to Gjilan is 185.0 km long.

Therefore, we are noticing that the shortest road from Gjakova to Gjilan is the road passing the road Gjakova-Rahovec which is 24.6 km, then to continue Rahovec-Theranda which is 17.4 km and ending from Theranda-Gjilan which is 69.4 km. And in total the distance travelled is equal to 111.4 km.

The search algorithm will start from the road Gjakova to Gjilan and will calculate the distances between the two cities and comparing with the final distance, if the distance between the two cities is greater than the total distance then stops the search to continue with the other way.

Below we present what the search algorithm looks like:

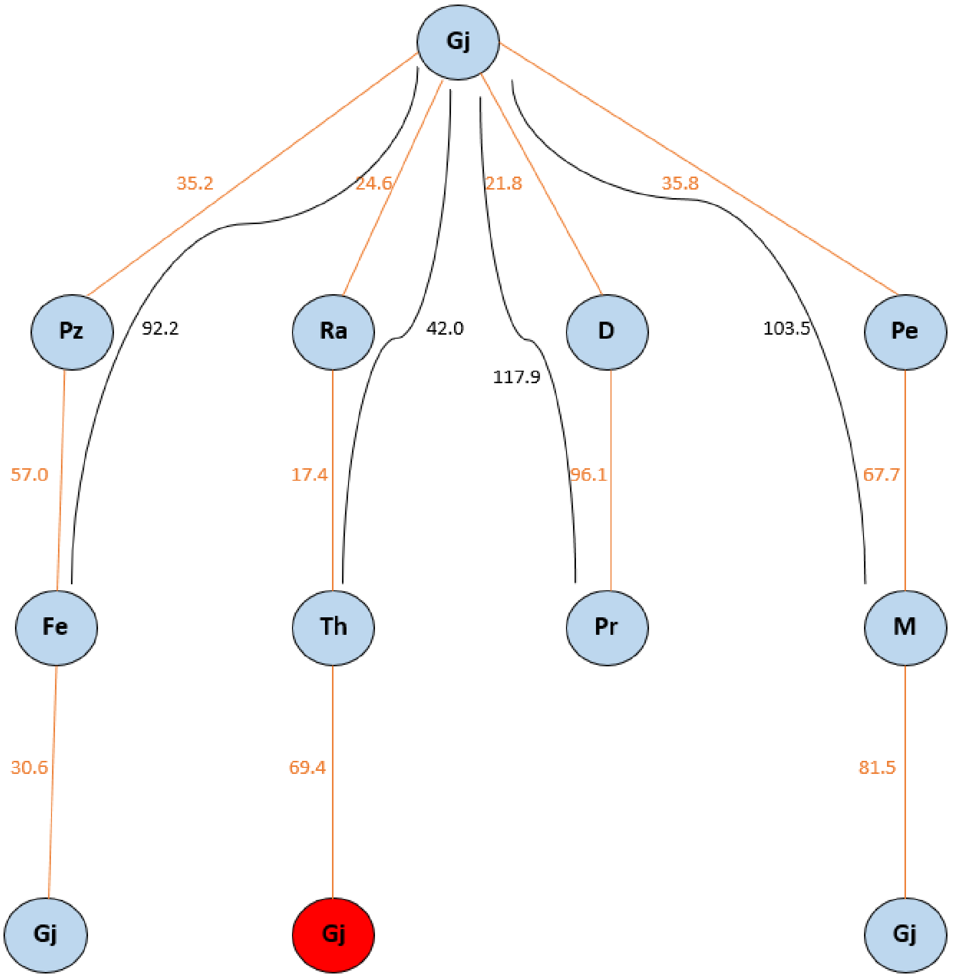

Figure 7. Search algorithm for the road Gjilan – Gjakova

4. CONCLUSION

Nowadays we have route planners such as Google Maps to make driving to destination a lot simpler. Using your GPS location and your target address, they compute an optimal route in the road network and show you in real-time which directions to take. The results of the Dijkstra's Algorithm analysis and the application implemented in this study were verified and validated using the shortest path determination approach. In this study we explained finding the shortest path between the two cities of the Republic of Kosovo through Dijkstra's Algorithm. The results of the study showed that the shortest road from Gjilan to Gjakova passes through the junctions which represent the city of Rahovec, Theranda to Gjilan. This road compared to alternative roads was the shortest road by 111.4 km. Therefore, we can conclude that Dijkstra's Algorithm is one of the main reasons for the popularity of Dijkstra's Algorithm is that it is one of the most important and useful algorithms available for optimal solutions to a large class of shortest path problem.

REFERENCES

1. N. Tyagi, Dijkstra’s Algorithm: The Shortest Path Algorithm, Analytic Steps, 2020. https://www.analyticssteps.com/blogs/dijkstras-algorithm-shortest-path-algorithm.
2. S. Catherine, Dijkstra's Shortest Path Algorithm: Overview and Examples, Study.com, 2003. www.classroom.google.com/?authuser=0.
3. T. Wai dhe H. Thaung, Shortest-Path Finding System using Dijkstra’s Algorithm, University of Computer Studies, Maubin, Myanmar, 2020.
4. Z. Fuhao dhe L. Jiping , An algorithm of shortest path based on dijkstra for huge data, International Conference on Fuzzy Systems and Knowledge Discovery, 2009. FSKD ’09, 4, 244-247, 2009.
5. K. Mehlhorn dhe P. Sanders, Algorithms and Data Structures: The Basic Toolbox., Springer. doi:10.1007/978-3-540-77978-0, 2008.
6. E. C. Navone, «Learn to Code,» Free Code Camp, 2022. https://www.freecodecamp.org/news/dijkstras-shortest-path-algorithm-visual-introduction/.
7. Lumen, «Graph Theory,» Mathematics for the Liberal Arts, 2022. https://courses.lumenlearning.com/waymakermath4libarts/chapter/shortest-path/.
8. M. Pandey, Dijkstra Algorithm, Scalet Topics, 20 12 2021. https://www.scaler.com/topics/data-structures/dijkstra-algorithm/.