Balkan Journal of Applied Mathematics and Informatics
Volume ?? Number ?
Year 201?

Print ISSN ???
Online ISSN ???
UDK: ???

SHORTEST PATH OPTIMIZATION IN TRANSPORTATION NETWORKS USING DIJKSTRA'S ALGORITHM: A CASE STUDY OF THE MACEDONIAN ROAD NETWORK

NATASHA STOJKOVIKJ, STEFANIJA FILEVA, ALEKSANDRA STOJANOVA ILIEVSKA, LIMONKA KOCEVA LAZAROVA, CVETA MARTINOVSKA BANDE AND VASKO KOKALANOV

Abstract. Efficient route determination in transportation networks represents an important challenge in modern navigation systems, logistics planning, and traffic management. Models based on graph theory provide an appropriate mathematical framework for representing road infrastructure and analysing optimal paths between different locations. This paper examines the application of Dijkstra's algorithm for solving the shortest-path problem in transportation networks modelled as weighted graphs. In the proposed model, cities in the Republic of North Macedonia are represented as vertices, while road connections between them are modelled as edges whose weights correspond to real geographical distances expressed in kilometres. To demonstrate the practical applicability of the proposed approach, a software application implemented in the Java programming language was developed. The developed system enables users to select a starting and a destination location through both command-line and graphical JavaFX interfaces, after which the algorithm automatically computes the shortest route within the modelled transportation network. The developed model illustrates how graph algorithms can be applied in the analysis and optimization of real transportation systems. The results obtained from the analysis of the modelled road network indicate that the algorithm efficiently identifies optimal routes and supports decision-making in route planning. The study confirms that Dijkstra's algorithm represents a reliable and computationally efficient method for solving shortest-path problems in road networks. The proposed approach has potential applications in navigation systems, transportation planning, and logistics optimization, highlighting the practical significance of graph algorithms in modern transportation infrastructures.[1]

1. Introduction

In modern conditions, transportation and mobility represent one of the key elements for the proper functioning of society. From daily travel to work and school to the transportation of goods and services across the country, the efficient movement of people and resources plays an important role in economic growth and social development. With the increasing level of urbanization and the constant growth in the number of vehicles, managing road infrastructure and transportation networks is becoming increasingly complex. Therefore, finding the fastest, safest, or most economical way to reach a

[1] *Date:* May 30, 2026.
Keywords. Dijkstra's algorithm; transportation networks; shortest path; graph theory; route optimization.

particular destination represents a significant challenge not only for everyday road users but also for engineers, researchers, and experts involved in the planning and optimization of transportation systems. Transportation networks form a vital part of modern infrastructure and play a key role in logistics, navigation systems, and regional economic development. Efficient route planning within these networks is important for reducing travel time, minimizing transportation costs, and enhancing traffic management. As transportation systems continue to grow in complexity, the development of computational techniques and optimization algorithms has become an increasingly significant research focus in fields such as computer science, transportation engineering, and operations research.

In computer science and mathematics, such problems are typically described and analysed using the graph theory. In these models, transportation networks are represented as graphs G(V,E) consisting of a set of vertices V and set of edges E .The vertices represent cities or intersections, and the edges represent the roads connecting them. A weight w(u,v) is assigned to each edge (u,v) that represents the "cost" of travel, which may correspond to geographical distance, travel time, or traffic conditions. Within this framework, the task of determining the shortest route between two locations can be formulated as a classical shortest-path problem in weighted graphs G(V,E,w). One of the most widely recognized algorithms for solving the shortest-path problem in weighted graphs is Dijkstra's algorithm, first introduced in 1959 [1]. The algorithm efficiently finds the shortest paths from a given source node to all other nodes in a graph with non-negative edge weights. Due to its simplicity and efficiency, Dijkstra's algorithm is widely considered one of the fundamental algorithms in the graph theory and network optimization [2]. Algorithms for computing the shortest path are widely applied across numerous domains, including transportation planning, communication networks, geographic information systems (GIS), and intelligent transportation systems [3][4]. Modern navigation systems and route-planning applications largely rely on shortest-path algorithms to analyse complex road networks and determine optimal travel routes. Research in transportation network analysis emphasizes the importance of shortest-path computation for modelling traffic flows and supporting informed decision-making in route planning [5]. Additionally, studies in geographic information science demonstrate the value of graph algorithms in spatial network analysis and route optimization [6].

In recent years, intelligent transportation systems have increasingly combined graph algorithms with digital maps and traffic data to improve navigation accuracy and overall transportation efficiency [7]. These approaches make it possible to analyse large-scale transportation networks and provide valuable support for decision-making in logistics management and traffic control [8]. Additionally, advanced route-planning techniques have been developed to handle extensive road networks and enhance computational efficiency in real-time navigation systems [9][10]. In addition to traditional models, literature increasingly focuses on more sophisticated approaches that address the analysis of multi-state networks and minimal path vectors. These studies propose methods for identifying minimal paths and applying them to evaluate network reliability and performance [11]. Furthermore, advanced algorithms have been developed to assess the

reliability of multi-state networks without restrictions on link capacities, representing a significant advancement in network analysis [12].

The objective of this research is to investigate the application of Dijkstra's algorithm in transportation network analysis using a case study of the road network in the Republic of North Macedonia. In the proposed model, cities are represented as vertices, while road connections between them are modelled as edges with weights corresponding to real geographical distances. A software implementation developed in the Java programming language is used to simulate the transportation network and compute the shortest routes between selected locations. The results of this study demonstrate the practical applicability of graph algorithms in transportation systems and highlight the effectiveness of Dijkstra's algorithm in route optimization and transportation network analysis. The findings confirm that mathematical models based on the graph theory can provide efficient solutions for real-world transportation problems and contribute to the development of modern navigation and logistics systems.

2. Materials and Methods

2.1 Graph-based modelling of the transportation network

Transportation and road networks can be represented as graphs, which are mathematical structures composed of vertices and edges. In the context of a road network:

- Vertices represent intersections or cities where roads meet.
- Edges represent roads, streets, or transportation connections between the vertices.
- Each connection (edge) has a weight that determines the "cost" of using that road. This weight may represent:
- Time – the time required to travel along the road.
- Distance – the length of the road expressed in kilometres.
- Cost – for example fuel consumption, tolls, or other financial expenses.

By properly modelling vertices and edges, any road network can be represented as a graph, which enables the application of algorithms for computing optimal routes, such as Dijkstra's algorithm. In Figure 1 a graph representation of a part of the transportation network is presented.

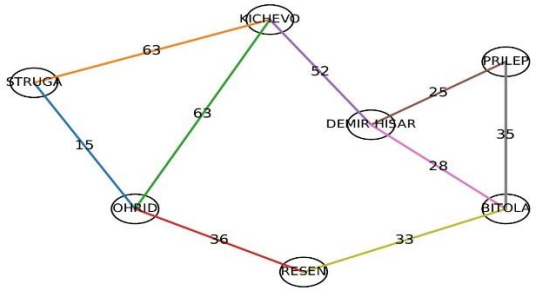


Figure 1. *Graph representation of a part of the transportation network.*

2.2 Dijkstra's algorithm for shortest-path computation

Dijkstra's algorithm is one of the most widely known algorithms for computing shortest paths in graphs with non-negative edge weights. The primary objective of the algorithm

is to determine the shortest path from a single source vertex to all other vertices in the graph. The algorithm operates under the assumption that all edge weights are non-negative, meaning that the values representing the cost of traversal—such as distance, travel time, or transportation cost—must be greater than or equal to zero. The shortest paths are determined through an iterative process in which the algorithm progressively selects the vertex with the smallest tentative distance from the source vertex among all unvisited vertices.

During each iteration, the algorithm examines the neighbouring vertices of the selected vertex. If a shorter path to a neighbouring vertex is found through the currently selected vertex, the corresponding distance value is updated. This procedure, commonly referred to as edge relaxation, ensures that the shortest distances are gradually determined for all vertices in the graph. The process continues iteratively until all vertices have been processed and the shortest paths from the source vertex to every other vertex in the graph have been determined (Figure 2).

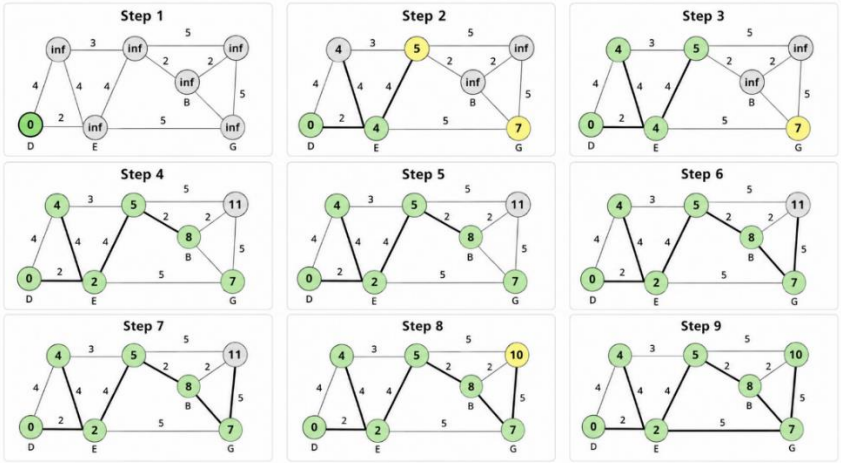


Figure 2. *Step-by-step execution of Dijkstra's algorithm on a weighted graph [13]*

Let the transportation network be represented as a weighted graph G(V,E,w). Each edge (u, v) from E has an associated weight $w(u, v)$, representing the distance between the corresponding cities.

The objective of the algorithm is to compute the shortest distance $d(v)$ from a selected source vertex s to every other vertex v∈V. The shortest-path condition can be expressed as:

$$(v) = \underbrace{min}_{(u,v)\in V} \{d(u) + w(u, v)\}$$

where $d(v)$ represents the shortest known distance from the source vertex to vertex and $w(u, v)$ represents the weight of the edge connecting vertices u and v.

The result is a set of shortest distances from the source vertex to all other vertices, which can be used to compute specific routes and to optimize transportation or traffic networks. The pseudocode of the proposed shortest-path computation procedure is presented in Algorithm 1.

Algorithm 1. Dijkstra's Shortest Path Algorithm

```
Input:
    G(V,E) – weighted graph
    s – source vertex
Output:
    Shortest distances and paths
for each vertex v in V do
    dist[v] ← ∞
    previous[v] ← NULL
end for

dist[s] ← 0
Q ← all vertices in priority queue
while Q is not empty do
    u ← vertex with minimum dist[u]
    remove u from Q
    for each neighbour v of u do
        alt ← dist[u] + weight(u,v)
        if alt < dist[v] then
            dist[v] ← alt
            previous[v] ← u
        end if
    end for
end while
return dist, previous
```

2.3 Application in transportation and road networks

Transportation and road networks represent a natural example of how the graph theory can be applied in practice. In this context, cities or intersections can be considered as vertices, while the roads connecting them are represented as edges. Each road can be assigned a weight, such as distance, travel time, or cost, depending on the objective of the analysis. Once the network is modelled in this way, it becomes much easier to analyse and process. Algorithms can then be applied to find optimal routes and improve the overall functioning of the system. One of the most used algorithms for this purpose is Dijkstra's algorithm, which is designed to find the shortest paths in a weighted graph.

In real transportation networks, this algorithm can be applied in several practical situations:

- Finding the shortest route between two points, which is important for navigation and everyday travel;
- Selecting the fastest path, especially when time is more important than distance;
- Providing alternative routes, for example in cases of traffic congestion or road closures;
- Reducing transportation costs, which is particularly important in logistics and delivery systems;
- Analysing traffic flow, in order to better understand how vehicles move through the network.

However, real-world transportation systems are not static. Traffic conditions change, roads may become temporarily unavailable, and external factors such as weather can affect travel. As a result, the values assigned to edges are not always fixed and often need

to be updated. For this reason, shortest path algorithms are used in a more flexible way, allowing routes to adapt to current conditions. In addition to route planning, they can also be used to identify important parts of the network, such as critical roads or intersections that have a significant impact on traffic flow. Today, these approaches are widely applied in navigation applications, route-planning systems, and traffic management solutions. They help users make better decisions when choosing routes and contribute to improving the overall efficiency of transportation systems. Overall, the application of Dijkstra's algorithm in transportation networks provides a clear example of how theoretical concepts can be successfully applied in real-world scenarios, offering practical solutions to everyday problems.

2.4 **Computational Complexity Analysis**

The computational complexity of shortest-path algorithms is an important factor in transportation network analysis, particularly when dealing with large-scale road infrastructures. When Dijkstra's algorithm is implemented using a priority queue based on a binary heap, its time complexity is: $O((V + E) \log V)$ where V denotes the number of vertices and E denotes the number of edges in the graph.

The space complexity is: $O(V)$ since the algorithm maintains distance and predecessor information for each vertex. For sparse transportation networks, which are common in road-network modelling, Dijkstra's algorithm provides efficient performance and can compute optimal routes within a relatively short execution time. This makes it suitable for practical route-planning applications and navigation systems.

Although more advanced approaches such as Contraction Hierarchies and Transit Node Routing can achieve faster query times on very large networks, Dijkstra's algorithm remains one of the most widely used shortest-path algorithms due to its simplicity, reliability, and ease of implementation [17,18].

3. **Application: Real Application of Dijkstra's Algorithm**

To demonstrate the practical applicability of Dijkstra's algorithm in transportation networks, a case study based on the road network of the Republic of North Macedonia was developed. The objective of this section is to illustrate how the previously introduced theoretical concepts can be applied in a real-world context and to analyse the results obtained from the implemented model. In the proposed approach, cities are represented as vertices, while the road connections between them are modelled as edges. Each edge is assigned a weight corresponding to the actual geographical distance between the cities, expressed in kilometres. This representation enables the transportation network to be treated as a weighted graph, allowing the application of shortest-path algorithms for route optimization.

The resulting graph-based model of the transportation network is presented in Figure 3, where the structure of the network and the relationships between the cities are clearly illustrated.

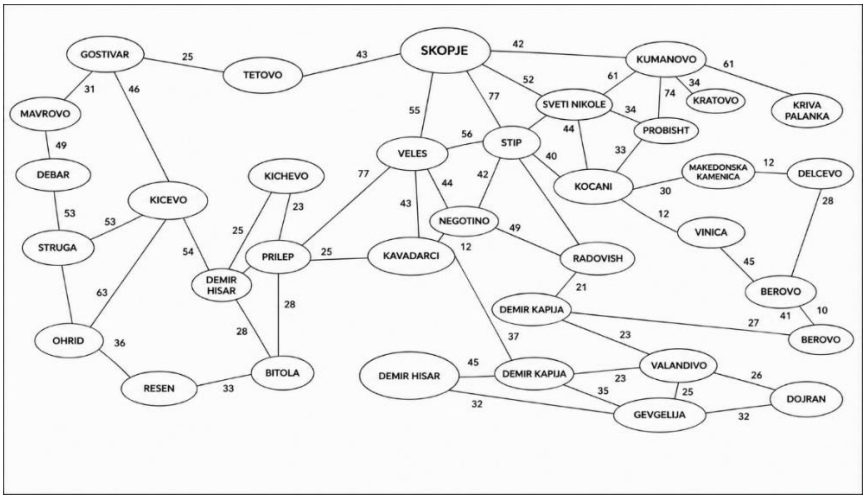


Figure 3. *Graph representation of the road network in the Republic of North Macedonia.*

Following the construction of the graph-based model, a software application was developed in Java to demonstrate the practical implementation of Dijkstra's algorithm for shortest-path computation in the modelled transportation network.

The developed system operates based on several key steps:

- Network representation – Each city is modelled as a vertex, while each direct road connection between two cities is represented as an edge with an assigned weight corresponding to the actual distance in kilometres. For example, the distance between Stip and Skopje is defined as 77 km.
- User input – The application requires the user to specify a source and a destination city (e.g., Stip and Skopje), which serve as input parameters for the algorithm.
- Shortest path computation – Using Dijkstra's algorithm, the system computes the shortest distances from the selected source city to all other cities in the network and determines the optimal route to the destination.
- Result visualization – The output is presented as a sequence of cities forming the shortest path, along with the total distance of the route, providing a clear and interpretable result for the user.

An example of the program execution is presented in Fig. 4.

```
C:\Users\Media\Desktop>java RoadNetworkMacedonia

Enter starting city:
Stip

Enter destination city:
Skopje

Shortest path: Stip -> Skopje
Total distance: 77 km
```

Figure 4. *Example of program execution showing the computed shortest path between selected cities*.

The presented result indicates that travel from Stip to Skopje can be performed directly, without passing through intermediate cities, as a direct connection exists within the network. This confirms the correctness of the algorithm in identifying the optimal route when such a path is available. The developed application represents a practical implementation of Dijkstra's algorithm in real transportation networks. By incorporating all relevant cities and their direct connections with accurate distance values, the proposed algorithm enables efficient and reliable route planning, shortest-path identification, and comprehensive analysis of transportation infrastructure. Furthermore, this type of system can be extended by integrating additional parameters such as travel time, traffic congestion, or road restrictions. Such enhancements would increase the applicability of the model in real-world navigation and transportation systems, allowing more advanced and realistic route optimization.

3.1 **Implementation details**

The implementation of the proposed system was carried out in Java, following a modular and structured design approach in order to ensure clarity and scalability. The transportation network is represented as a graph using an adjacency list structure, where each city is associated with a collection of its neighbouring cities and corresponding distances. In this context, each connection is modelled as an object that encapsulates the target node and the associated weight, representing the distance in kilometres. The graph structure allows efficient representation of multiple direct connections between cities and supports bidirectional relationships when required. All nodes are initialized prior to computation to ensure consistency and avoid runtime errors during graph traversal. The shortest-path computation is performed using Dijkstra's algorithm, supported by several key data structures. A distance mapping is maintained to store the shortest known distances from the source city to all other cities. Additionally, a predecessor mapping is used to reconstruct the optimal route once the computation is complete. A priority queue is employed to efficiently select the next node with the smallest tentative distance at each step of the algorithm. During execution, the algorithm iteratively applies edge relaxation, updating distance values whenever a shorter path is identified. This process continues until all nodes are processed and the optimal distances are determined. Following the computation, the shortest path is reconstructed by tracing the predecessor relationships from the destination node back to the source. The final output is presented as an ordered sequence of cities forming the optimal route, along with the total distance. The modular structure of the implementation enables straightforward extension of the model, allowing the inclusion of additional parameters such as travel time, traffic conditions, or transportation costs.

3.2 **Visualization of the Transportation Network**

In addition to the algorithmic implementation, a visual representation of the transportation network was developed in order to enhance the interpretation of the results and provide a more intuitive understanding of the computed shortest paths. The visualization was implemented using the JavaFX framework, enabling graphical representation of the network structure and dynamic animation of the algorithm's execution. The system was developed using JDK 26 and JavaFX SDK 26, ensuring compatibility with modern Java environments and efficient rendering of graphical

components. In the visual model, cities are displayed as nodes, while the road connections between them are represented as edges. The layout provides a clear overview of the network structure, allowing users to observe how cities are interconnected and how distances influence route selection. Furthermore, the visualization supports animation of the shortest-path computation process, illustrating how Dijkstra's algorithm explores the network and updates distances step by step. This dynamic representation improves the interpretability of the algorithm and facilitates better understanding of its behaviour. An example of the graphical representation of the transportation network during program execution is shown in Figure 5.

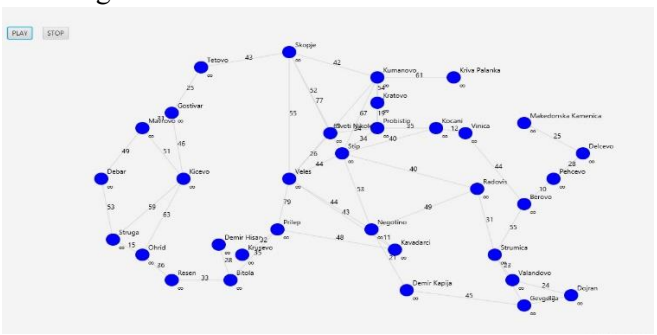
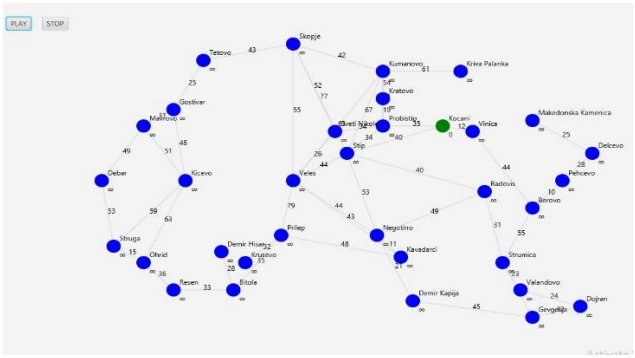


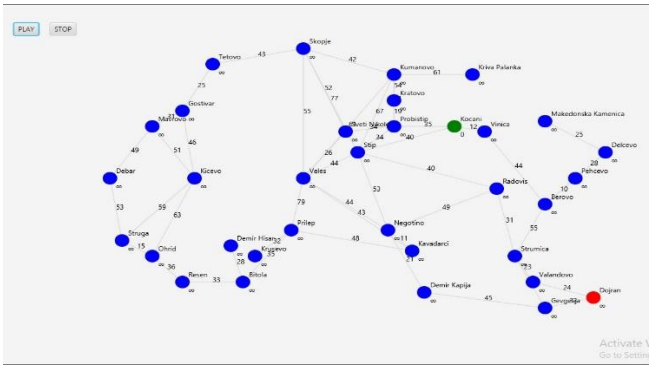
Figure 5. *Dynamic visualization of Dijkstra's algorithm execution on the transportation network using JavaFX.*

The developed system also supports user interaction, allowing intuitive selection of the source and destination nodes. Specifically, the user can select the starting city by clicking on a node, followed by selecting the destination city in the same manner. After selecting the desired nodes, the execution of the algorithm is initiated by pressing the "Play" button. The system then starts the animation of Dijkstra's algorithm, visually demonstrating how the shortest path is explored and constructed. The animation can be paused at any time using the "Stop" button and resumed by pressing "Play", allowing step-by-step observation of the algorithm's behaviour.

An example of node selection and initialization of the algorithm is presented in Figure 6.



a)



b)

Figure 6. *Selection of source and destination nodes in the visualization system*

During execution, the algorithm dynamically updates node states and highlights intermediate steps of the computation process. The final shortest path is emphasized using a distinct colour, clearly indicating the optimal route between the selected cities.
An example of the computed shortest path is shown in Figure 7.

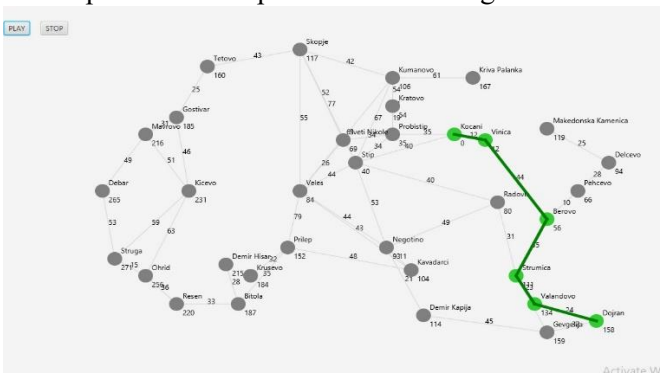


Figure 7. *Visualization of the computed shortest path using Dijkstra's algorithm.*

The animation and interactive features provide significant advantages in terms of usability and analysis. They allow users to better understand the internal behaviour of the algorithm, observe intermediate steps, and visually interpret the results.

Such an approach demonstrates the practical value of integrating algorithmic solutions with graphical interfaces, making the system suitable not only for research purposes but also for educational applications and real-world transportation analysis.

3.3 **Comparative Analysis with Related Studies**

Several studies have investigated shortest-path computation and route optimization in transportation networks. Zhan and Noon [15] evaluated multiple shortest-path algorithms using real road-network data and demonstrated that Dijkstra's algorithm provides reliable performance for static transportation environments with non-negative edge weights. Hart et al. [14] introduced the A* algorithm, which improves route-search efficiency by incorporating heuristic information. Although A* can reduce the search space, its

performance depends on the quality of the heuristic function. Geisberger et al. [16] proposed the Contraction Hierarchies approach, which significantly accelerates route computation in large-scale road networks through graph preprocessing. However, such methods require additional preprocessing steps and more complex implementation. Delling et al. [17] and Bast et al. [18] investigated advanced route-planning techniques designed for large transportation networks and real-time navigation systems. Their studies emphasize scalability and computational efficiency for extensive road infrastructures. More recently, Li et al. [19] presented a comprehensive survey of route-planning approaches, highlighting the importance of shortest-path algorithms in intelligent transportation systems and smart-city applications. Similarly, Grujić and Grujić [21] demonstrated the practical applicability of Dijkstra's algorithm in urban road-network optimization.

Compared with these studies, the proposed work focuses on the transportation network of the Republic of North Macedonia and combines shortest-path computation with an interactive JavaFX visualization environment. Unlike advanced hierarchical routing approaches, the developed system prioritizes simplicity, educational value, and practical demonstration of graph-based transportation analysis while maintaining reliable shortest-path computation using Dijkstra's algorithm. To better position the proposed system within the existing literature, a comparison with several representative shortest-path and route-planning approaches is presented in Table 1.

Table 1. *Comparison of Related Studies and the Proposed System*

Study	**Algorithm**	**Real Road Network**	**Visualization**	**Complexity**
Hart et al. [14]	A*	Yes	No	Depends on heuristic
Zhan & Noon [15]	Multiple Algorithms	Yes	No	Comparative
Geisberger et al. [16]	Contraction Hierarchies	Yes	No	Very Fast Queries
Bast et al. [18]	Advanced Routing Methods	Yes	No	Highly Optimized
Grujić et al. [21]	Dijkstra	Yes	No	O((V+E)logV)
Proposed System	Dijkstra	Yes (Macedonian Network)	JavaFX Interactive Visualization	O((V+E)logV)

As shown in Table 1, the comparison demonstrates that although advanced routing techniques provide superior performance on very large transportation networks, the proposed system offers a balance between computational efficiency, implementation simplicity, and visualization capabilities. Furthermore, the use of real-world data from the Macedonian road network increases the practical relevance of the developed solution.

4. Conclusion

Dijkstra's algorithm represents an efficient and robust method for computing shortest paths in graphs with non-negative edge weights. Its application in transportation and road networks enables accurate and efficient route planning, identification of optimal paths, and analysis of connectivity between locations.

Through the developed Java-based model of the Macedonian road network, it has been demonstrated how real-world data, such as geographical distances and direct road connections, can be effectively represented within a graph structure. The application of Dijkstra's algorithm on this model allows for reliable computation of the shortest routes between any pair of cities, confirming its practical relevance beyond theoretical contexts. The results highlight that Dijkstra's algorithm can serve as a fundamental component in navigation systems, transportation planning, and traffic management applications. By providing precise and computationally efficient solutions, the algorithm contributes to improved decision-making and optimization of transportation processes.

Furthermore, with appropriate modelling of the network and the use of accurate input data, the algorithm can support the identification of the fastest and most efficient routes. This makes it a valuable tool in modern transportation and logistics systems. The developed system demonstrates that graph-based models can effectively represent real transportation infrastructures and support shortest-path optimization in practical navigation scenarios. Future work may include extending the model by incorporating dynamic parameters such as traffic conditions, travel time variability, and road constraints, which would enhance the applicability of the approach in real-time and large-scale transportation environments.

References

[1] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. Numerische Mathematik, 1, 269–271. https://doi.org/10.1007/BF01386390

[2] Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C. (2009). Introduction to Algorithms. MIT Press.

[3] Ahuja, R. K., Magnanti, T. L., Orlin, J. B. (1993). Network Flows: Theory, Algorithms, and Applications. Prentice Hall.

[4] Diestel, R. (2017). Graph Theory (5th ed.). Springer.

[5] Sheffi, Y. (1985). Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods. Prentice Hall.

[6] Longley, P., Goodchild, M., Maguire, D., Rhind, D. (2015). Geographic Information Science and Systems. Wiley.

[7] Bell, M. G. H., Iida, Y. (1997). Transportation Network Analysis. Wiley.

[8] Rodrigue, J. P., Comtois, C., Slack, B. (2020). The Geography of Transport Systems. Routledge.

[9] Delling, D., Sanders, P., Schultes, D., Wagner, D. (2009). Engineering route planning algorithms. Algorithmics of Large and Complex Networks, Springer.

[10] Bast, H., Delling, D., Goldberg, A., Müller-Hannemann, M., Pajor, T., Sanders, P., Wagner, D., Werneck, R. (2016). Route planning in transportation networks. ACM Computing Surveys, 46(4), 1–36. https://doi.org/10.1145/2886843

[11] M. Mihova, N. Maksimova, "Estimation of minimal path vectors of multi state two terminal networks with cycles control," Mathematica Balkanica, New Series, Vol. 25, Fasc. 4, pp. 437–446, 2011.

[12] Mihova, M., Maksimova, N., Popeska, Z. (2008). An algorithm for calculating multi-state network reliability with arbitrary capacities of the links. International Scientific Conference Computer Science'2008, pp. 170–175.

[13] W3Schools. Dijkstra's Algorithm Tutorial. Available at: https://www.w3schools.com/dsa/dsa_algo_graphs_dijkstra.php (Accessed: May 2026).

[14] Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, *4*(2), 100-107.

[15] Zhan, F. B., & Noon, C. E. (1998). Shortest path algorithms: an evaluation using real road networks. *Transportation science*, *32*(1), 65-73.

[16] Geisberger, R., Sanders, P., Schultes, D., & Delling, D. (2008, May). Contraction hierarchies: Faster and simpler hierarchical routing in road networks. In *International workshop on experimental and efficient algorithms* (pp. 319-333). Berlin, Heidelberg: Springer Berlin Heidelberg.

[17] Delling, D., Sanders, P., Schultes, D., & Wagner, D. (2009). Engineering route planning algorithms. In *Algorithmics of large and complex networks: design, analysis, and simulation* (pp. 117-139). Berlin, Heidelberg: Springer Berlin Heidelberg.

[18] Bast, H., Delling, D., Goldberg, A., Müller-Hannemann, M., Pajor, T., Sanders, P., ... & Werneck, R. F. (2016). Route planning in transportation networks. In *Algorithm engineering: Selected results and surveys* (pp. 19-80). Cham: Springer International Publishing.

[19] Li, K., Rao, X., Pang, X., Chen, L., & Fan, S. (2021). Route search and planning: A survey. *Big data research*, *26*, 100246.

[20] Strasser, B., Wagner, D., & Zeitz, T. (2021). Space-efficient, fast and exact routing in time-dependent road networks. *Algorithms*, *14*(3), 90.

[21] Grujic, Z., & Grujic, B. (2025). Optimal routing in urban road networks: a graph-based approach using Dijkstra's algorithm. *Applied Sciences*, *15*(8), 4162.

Natasha Stojkovikj
Goce Delcev University,
Faculty of Computer Science,
Address Krste Misirkov, 10A,
2000, Stip, North Macedonia
E-mail address: natasa.stojkovik@ugd.edu.mk

Stefanija Fileva
Goce Delcev University,
Faculty of Computer Science,
Address Krste Misirkov, 10A,
2000, Stip, North Macedonia
E-mail address: stefanija.102848@student.ugd.edu.mk

Aleksandra Stojanova Ilievska
Goce Delcev University,
Faculty of Computer Science,
Address Krste Misirkov, 10A,
2000, Stip, North Macedonia
E-mail address: aleksandra.stojanova@ugd.edu.mk

Limonka Koceva Lazarova
Goce Delcev University,
Faculty of Computer Science,
Address Krste Misirkov, 10A,
2000, Stip, North Macedonia
E-mail address: limonka.lazarova@ugd.edu.mk

Cveta Martionvska Bande
Goce Delcev University,
Faculty of Computer Science,
Address Krste Misirkov, 10A,
2000, Stip, North Macedonia
E-mail address: cveta.martinovska@ugd.edu.mk

Vasko Kokalanov
Goce Delcev University,
Faculty of Computer Science,
Address Krste Misirkov, 10A,
2000, Stip, North Macedonia
E-mail address: vasko.kokalanov@ugd.edu.mk