American Journal of Science, Engineering and Technology
2023; 8(3): 146-151
http://www.sciencepublishinggroup.com/j/ajset
doi: 10.11648/j.ajset.20230803.14
ISSN: 2578-8345 (Print); ISSN: 2578-8353 (Online)

A Modified Approach of Dijkstra's Method for Finding Shortest Path in a Weighted Directed Graph

Md. Mehedi Hassan[*], Md. Asadujjaman, Md. Golam Robbani

Department of Mathematics, University of Dhaka, Dhaka, Bangladesh

Email address:

mehedihassan116328@gmail.com (Md. Mehedi Hassan), asad132@du.ac.bd (Md. Asadujjaman), mdgolamrobbanisk@gmail.com (Md. Golam Robbani)

[*]Corresponding author

To cite this article:

Md. Mehedi Hassan, Md. Asadujjaman, Md. Golam Robbani. A Modified Approach of Dijkstra's Method for Finding Shortest Path in a Weighted Directed Graph. *American Journal of Science, Engineering and Technology*. Vol. 8, No. 3, 2023, pp. 146-151.
doi: 10.11648/j.ajset.20230803.14

Received: June 8, 2023; **Accepted**: June 25, 2023; **Published**: July 31, 2023

Abstract: The shortest route technique is a fundamental problem in various fields, including transportation, logistics, network routing, and robotics. In this paper, we have discussed a prominent algorithm, namely Dijkstra's algorithm, and propose an alternative method for addressing these problems. A thorough comparison is conducted between the proposed algorithm and Dijkstra's algorithm, considering factors such as solution accuracy and computational efficiency. The experimental results indicate that our proposed method yields identical results to the existing method but with significantly reduced computation time. By leveraging advancements in computational power and algorithmic design, our proposed technique addresses the limitations of existing methods and offers new avenues for optimizing route planning processes. We begin by reviewing the classical algorithms commonly used for solving the shortest route problem, such as Dijkstra's algorithm. While this algorithm has proven its effectiveness over the years, it faces challenges when applied to large-scale networks and real-time applications due to its computational complexity. Our approach incorporates advanced data structures and optimization strategies to efficiently handle massive network graphs. Additionally, we integrate machine learning models to learn from historical data, allowing for the prediction of traffic patterns and considering dynamic factors in route planning.

Keywords: Network Routing, Dijkstra's Algorithm, Floyd's Algorithm, Triple Operation, Vehicle Routing Problem

1. Introduction

The introduction of shortest route problems in the field of Operations Research (OR) has been instrumental in solving various optimization challenges related to transportation, logistics, and network planning. The shortest route problem aims to find the most efficient path or route between two locations in a network, considering factors such as distance, time, cost, or other relevant criteria. The development of techniques to address the shortest route problem has significantly impacted industries and sectors that rely on efficient transportation and logistics operations.

The concept of the shortest route problem can be traced back to the early days of OR when researchers recognized the need for mathematical models to optimize transportation and logistics decisions. The study of the shortest route problem gained prominence during World War II, where it played a crucial role in military logistics and strategic planning.

The shortest route problem is typically framed within the framework of graph theory, where a network is represented by a graph consisting of nodes and edges. Nodes represent locations or points of interest, and edges represent their connections or routes. Each edge is assigned a weight or cost, such as distance or travel time, which reflects the effort required to traverse that route. [2]

Various algorithms have been developed to solve the shortest route problem efficiently. One of the most well-known algorithms is Dijkstra's algorithm, proposed by Dutch computer scientist Edsger Dijkstra in 1956. Dijkstra's algorithm finds the shortest path from a specified origin node to all other nodes in a network. It has been widely used in transportation planning, logistics optimization, and network routing.

The introduction of shortest route problems in OR has provided decision-makers with powerful tools to optimize transportation and logistics operations. By finding the most efficient routes, organizations can reduce transportation costs, minimize delivery times, improve customer satisfaction, and enhance overall operational efficiency.

2. Shortest Path Algorithms

2.1. Dijkstra's Method [1]

Suppose we want to find the shortest path from a given node s to other nodes in a network (one-to-all shortest path problem) [4].

1) Dijkstra's algorithm solves such a problem
   a) It finds the shortest path from a given node s to all other nodes in the network
   b) Node s is called a starting node or an initial node
2) How is the algorithm achieving this? [5]
   a) Dijkstra's algorithm starts by assigning some initial values for the distances from node s and to every other node in the network.
   b) It operates in steps, where at each step the algorithm improves the distance values. [6]
   c) At each step, the shortest distance from node s to another node is determined.

2.1.1. Algorithm Steps for Dijkstra's Algorithm [1]

Step 1. (Initialization)

1) Assign the zero-distance value to node s, and label it as Permanent. [The state of node s is $(0, p)$.] [7]
2) Assign to every node a distance value of ∞ and label them as Temporary. [The state of every other node is (∞, t).] [8]
3) Designate the node s as the current node.

Step 2. (Distance Value Update and Current Node Designation Update) [9]

Let i be the index of the current node.

1) Find the set J of nodes with temporary labels that can be reached from the current node i by a link (i, j). Update the distance values of these nodes. For each $j \in J$, the distance value d_j of node j is updated as follows:
   New $d_j = min\{d_j, d_i + c_{ij}\}$; where c_{ij} is the cost of link (i, j), as given in the network problem. [10]
2) Determine a node j that has the smallest distance value d_j among all nodes $j \in J$, find $j*$ such that, $min\, j \in J, d_j = d_{j*}$
3) Change the label of node $j*$ to permanent and designate this node as the current node.

Step 3. (Termination Criterion) [10]

1) If all nodes that can be reached from node s have been permanently labeled, then stop - we are done.
2) If we cannot reach any temporary labeled node from the current node, then all the temporary labels become permanent - we are done.
3) Otherwise, go to Step 2.

2.1.2. Example of Dijkstra's Algorithm

We want to find the shortest path from node 1 to all other nodes using Dijkstra's algorithm of network which is shown in figure 1.

Step 1 (Initialization)

1) Node 1 is designated as the current node
2) The state of node 1 is $(0, p)$.
3) Every other node has state (∞, t)

The associated figure is figure 2.

Step 2 (Distance Value Update and Current Node Designation Update)

1) Nodes 2, 3, and 6 can be reached from the current node 1.
2) Update distance values for these nodes

$$d_2 = min\,\{\infty, 0 + 7\} = 7$$

$$d_3 = min\,\{\infty, 0 + 9\} = 9$$

$$d_6 = min\,\{\infty, 0 + 14\} = 14$$

3) Now, among the nodes 2, 3, and 6, node 2 has the smallest distance value.
4) The status label of node 2 changes to permanent, so its state is $(7, p)$, while the status of nodes 3 and 6 remains temporary.
5) Node 2 becomes the current node.

The associated figure is figure3.

Step 3

From the graph at the end of Step 2 we see that we are not done, not all nodes have been reached from node 1, so we perform another iteration (back to Step 2).

Another Implementation of Step 2

1) Nodes 3 and 4 can be reached from the current node 2.
2) Update distance values for these nodes

$$d_3 = min\,\{9, 7 + 10\} = 9$$

$$d_6 = min\,\{\infty, 7 + 15\} = 22$$

3) Now, between the nodes 3 and 4 node 3 has the smallest distance value
4) The status label of node 3 changes to permanent, while the status of node 6 remains temporary.
5) Node 3 becomes the current node.

The associated figure is figure 4.

We are not done (Step 3 fails), so we perform another Step 2

Another Step 2

1) Nodes 6 and 4 can be reached from the current node 3.
2) Update distance values for them

$$d_4 = min\,\{22, 9 + 11\} = 20$$

$$d_6 = min\,\{14, 9 + 2\} = 11$$

3) Now, between the nodes 6 and 4 node 6 has the smallest distance value.
4) The status label of node 6 changes to permanent, while

the status of node 4 remains temporary.

5) Node 6 becomes the current node.

The associated figure is figure5.

We are not done (Step 3 fails), so we perform another Step 2

Another Step 2

1) Node 5 can be reached from the current node 6
2) Update distance value for node 5

$$d_5 = min\{\infty, 11 + 9\} = 20$$

3) Now, node 5 is the only candidate, so its status changes to permanent.
4) Node 5 becomes the current node.
5) From node 5 we cannot reach any other node. Hence, node 4 gets permanently labeled and we are done.

The final figure is figure6.

2.2. Proposed Modified Approach of Dijkstra's Method

For my proposed method, the initial step is similar to Dijkstra's, and I will use the Floyd-Warshall method in the middle of my procedure if needed. For initializing we have to select a node of the network as the initial node from where we go any other node of the network and mark it as permanent. Now select the adjacent node(s) of the permanent node and identify the ingoing and outgoing flow of this (these) adjacent node(s). We have to mark the adjacent node as permanent with its distance value if it has only one ingoing flow, otherwise, mark it as temporary with its distance value. If there is more than one ingoing flow to the adjacent node then we have to apply triple operation to find the shortest distance. The triple operation is as follows (figure 7): [11]

Given three nodes $i, j,$ and k in the above Figure with the connecting distances shown on the three arcs, it is shorter to reach j from i passing through k if

$$d_{ik} + d_{kj} < d_{ij}$$

In this case, it is optimal to replace the direct route from $i \to j$ with the indirect route $i \to k \to j$.

Now let x is the adjacent node to the permanent node which has more than one ingoing flow and let y is the node that is adjacent to node x and is used for performing the triple operation, then it has to be sure that node y is labeled as permanent if not the make it as permanent then apply the triple operation. We have to continue this process until all the nodes are labeled as permanent.

For labeling a nod as permanent we have to choose the minimum distance value for finding the shortest route from starting node to finishing node. After labeling all nodes as permanent we can find the shortest path from the back calculation.

2.2.1. Algorithm for Proposed Method

Step 1

Select the source node and label it as permanent.

Step2

Select the adjacent nodes to the recent permanent node and label it as a permanent node with its distance value if it has only one ingoing flow, otherwise, label it as a temporary node and go to step 3.

Step 3

Apply triple operation for labeling the node as permanent with minimum distance value. It may happen that an adjacent node of the current permanent node is labeled as permanent after a few iterations depending on the distance value of arc and ingoing flow towards the node. [14]

Step 4

Check whether all the nodes are labeled as permanent, if not, go to step 2, otherwise, go to step 5.

Step 5

Stop, our process is done. The shortest distance from the source node to each node can be found in the distances array, and the shortest path from the source node to each node can be reconstructed using the previous node information.

Note: This algorithm assumes that the graph does not contain negative edge weights. If the graph contains negative edge weights, alternative algorithms like the Bellman-Ford algorithm or specialized versions of Dijkstra's algorithm, such as Dijkstra's algorithm with a min-priority queue, should be used. [13]

It's worth mentioning that the algorithm can be optimized using a priority queue data structure to efficiently find the unvisited node with the smallest tentative distance. This helps improve the overall runtime complexity of the algorithm. [12]

2.2.2. Example of Proposed Method

Here we solve the previous problem (figure 1).

Step 1

We select node 1 as the initial node and labeled it as permanent where the network is becomes as figure 8.

Step 2

The adjacent nodes of node 1 is node 6, node 3, and node 2 where node 2 has only one ingoing flow so we label it as permanent with its distance value and label the other nodes as temporary with its distance value as shown in figure 9.

Step 3

For applying triple operation in the set of node $\{1, 3, 6\}$ where we want to go from node 1 to node 6 via node 3 but node 3 is not labeled as permanent, so we have to make node 3 permanent first. For this, we have to apply the triple operation in the set of node $\{1, 2, 3\}$ and we get

$$d_{12} + d_{23} = 7 + 10 = 17 > d_{13} = 9$$

So we label node 3 as permanent with distance value 9. Now for the set $\{1, 3, 6\}$ we get

$$d_{13} + d_{36} = 9 + 2 = 11 < d_{16} = 14$$

So we label node 6 as permanent with distance value 11.

Here the adjacent node of node 6 is node 5 and it has two ingoing flows so label it as temporary with distance value $11 + 9 = 20$ but here we can't apply triple operation because there is no direct link between node 6 and node 4. Node 4 is the adjacent node of node 3 and node 2. Here node 4 has two ingoing flows but here we don't apply triple operation because this is adjacent to both node 3 and node 2 and since the distance value of node 4 from node 3 is smaller than from node 2 so we label node 4 as permanent with distance value 20 as figure 10.

Now since node 4 is permanent so the distance value of node 5 via node 4 is 26 which is greater than 20 so label node 5 as permanent with distance value 20 as figure 11.

And we are done.

Here the shortest route from node 1 to node 5 is 5 ← 6 ← 3 ← 1 and the shortest distance is 20, which is the same as Dijkstra's method.

3. Figures Used in Both Methods

3.1. Figure Used in Dijkstra's Method

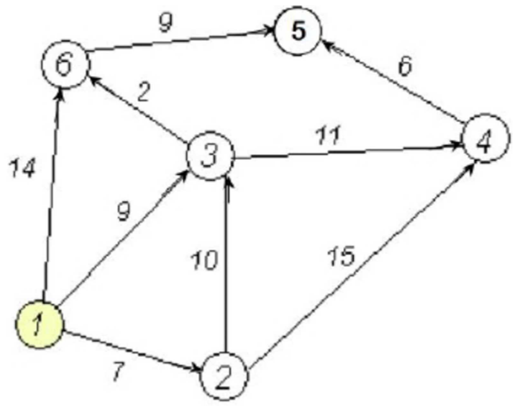

Figure 1. *Network to solve [3].*

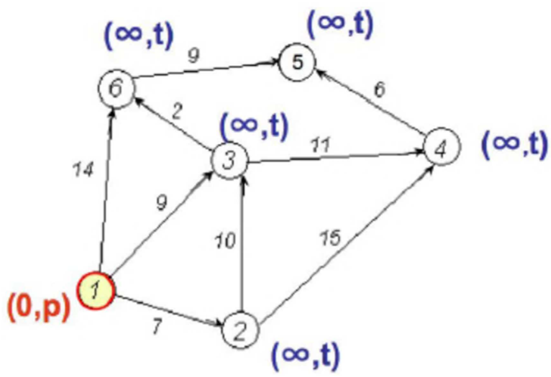

Figure 2. *Network for step 1.*

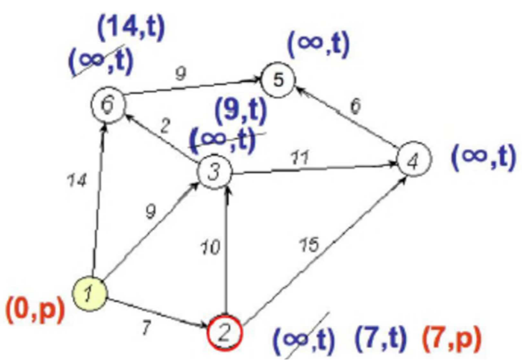

Figure 3. *Network for step 2.*

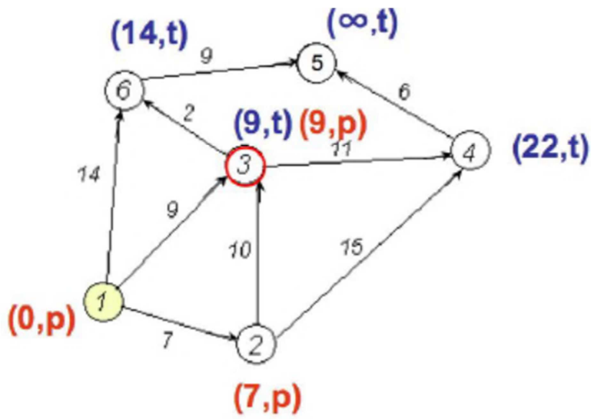

Figure 4. *Network for another implementation of step 2.*

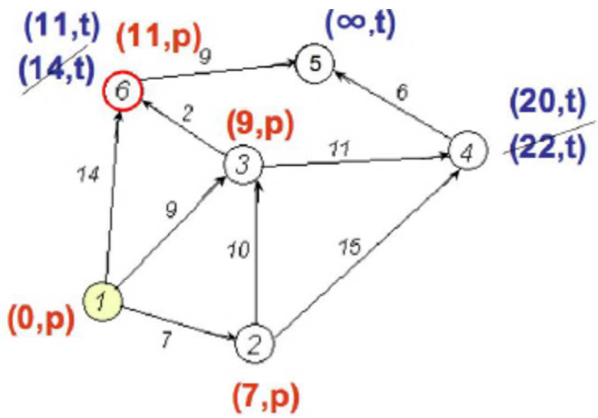

Figure 5. *Network for another step 2.*

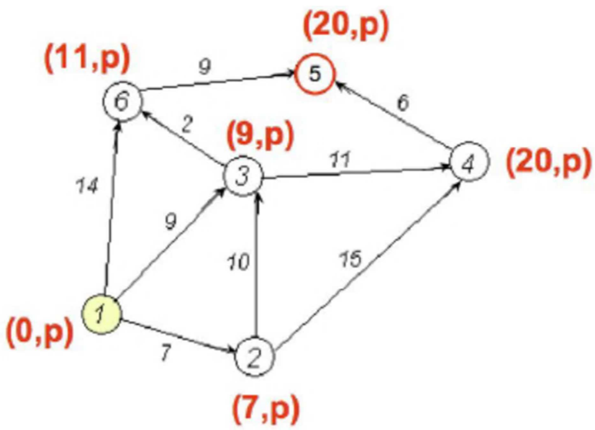

Figure 6. *Final Network*

3.2. Figure Used in Proposed Method

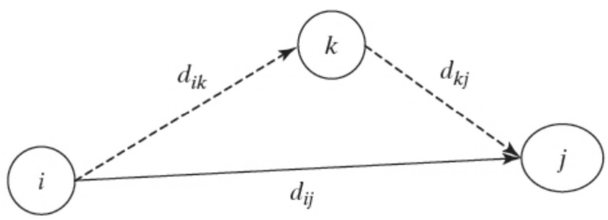

Figure 7. *Figure for Triple Operation.*

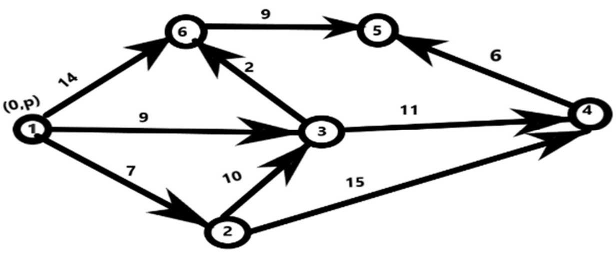

Figure 8. *Network for step 1.*

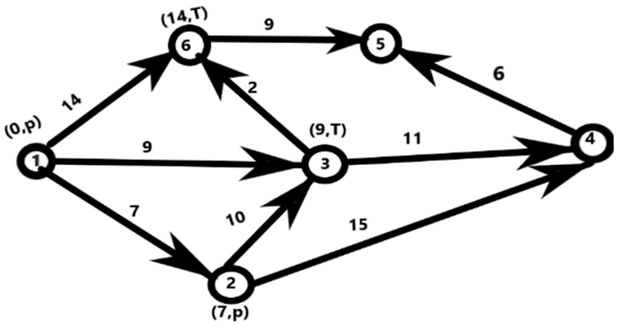

Figure 9. *Network for step 2.*

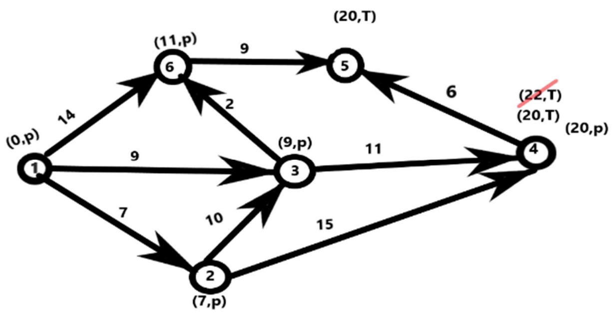

Figure 10. *Network for step 3.*

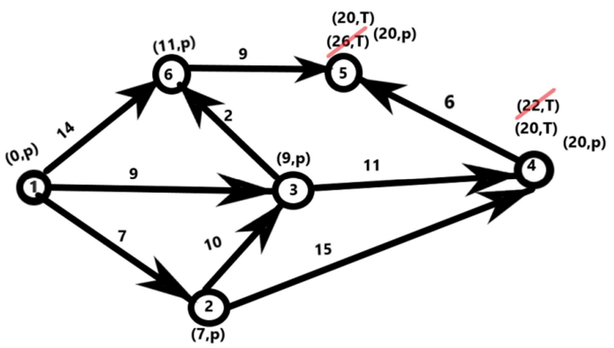

Figure 11. *Final Network.*

4. Conclusion

In Dijkstra's method node which has a minimum weight from the current permanent node is made permanent and take this node as a current node where the unvisited nodes are labeled as (∞, T). On the other hand, in my proposed method nodes which are adjacent to the current permanent node are made permanent (where possible) and for this purpose, we use triple operation if necessary.

The problem that I solved recently with the help of my proposed method takes 4 steps whereas Dijkstra's method takes 6 steps. Moreover, in my proposed method calculation is comparatively easy and convenient which minimizes the labor and processing time.

Our proposed method offers significant improvements in computation time and solution quality, making it highly suitable for real-time applications in transportation, logistics, and other related domains. Future research directions include further optimization and integration with emerging technologies to enhance the applicability and scalability of our technique.

Acknowledgements

First, I would like to convey my gratitude to the almighty Allah for giving me the opportunity to complete this thesis paper.

I would like to convey my heartfelt thanks and gratitude to my supervisor, Md. Asadujjaman, Assistant Professor, Department of Mathematics, Faculty of Science, University of Dhaka, Dhaka-1000, Bangladesh, who instructed me to prepare this thesis paper and provided me with his all-out efforts despite being busy with his daily schedule. I am very much grateful to him for his cordial contribution.

Now I must expose my grant honor to the most respectable Chairman, Department of Mathematics, University of Dhaka to give me all types of opportunities to take over the thesis. I am also grateful to the other teachers in my department. Whenever I remind them of their attempt, kindness, and sincerity, then it seems that I am indebted to them.

Furthermore, we extend our thanks to the academic community and institutions that have provided access to resources, libraries, and datasets necessary for conducting our research. The availability of these resources has significantly enhanced the scope and depth of our analysis.

This research was made possible through the financial assistance provided by the scholarship, which greatly contributed to the successful completion of this work. We extend our deepest gratitude to "*NATIONAL SCIENCE AND TECHNOLOGY (NST) FELLOWSHIP 2022-23*" for their commitment to fostering academic excellence and their investment in our research endeavors.

Lastly, we would like to acknowledge the support and understanding of our families and friends, whose encouragement and patience have been crucial throughout this research endeavor. Their unwavering support has provided us with the motivation and inspiration to overcome challenges and strive for excellence.

References

[1] Taha H. A., Operations Research An Introduction, Prentice Hall of India Pvt. Ltd, New Delhi, 1999.

[2] Winston, W. L., J. B. Goldberg, 2004, "Operations Research: Applications and Algorithms", Thomson Brooks.

[3] https://mathinsight.org/network_introduction.

[4] https://www.chegg.com/homework-help/questions-and-answers/d-10-10-use-dijkstra-s-algorithm-findlength-shortest-path-vertices-z-following-weighted-g-q69930813

[5] https://studylib.net/doc/18121752/lecture-18-solving-shortest-path-problem--dijkstra-s-algo...

[6] https://www.uobabylon.edu.iq/eprints/publication_4_9401_1410.pdf

[7] https://www.ques10.com/p/41050/single-source-shortest-path-1

[8] http://mdcollege.in/wp-content/uploads/2019/08/djikstras-algorithm.pdf

[9] https://medium.com/@letslearnsomething95/dijkstras-shortest-path-finding-algorithm-with-a-solvedexample-df892915b0ca

[10] https://www.coursehero.com/file/p67feerr/Distance-Value-Update-and-Current-Node-Designation-Update-Let-i-be-the-index-of

[11] Aardal, K. I., van der Veen, J. A., & Weismantel, R. (2018). The Traveling Salesman Problem. In 50 Years of Integer Programming 1958-2008 (pp. 53-84). Springer.

[12] B. Korte and J. Vygen, "*Network Flows*," pp. 153–184, 2000, doi: 10.1007/978-3-662-21708-5_8.

[13] R. Danchick, "*An Exact Ranked Linear Assignments Solution to the Symmetric Traveling Salesman Problem*," *OAlib*, vol. 07, no. 03, pp. 1–8, 2020, doi: 10.4236/OALIB.1106159.

[14] Z. Sun *et al.*, "*Optimal Path Finding Method Study Based on Stochastic Travel Time,*" *J Transp Technol*, vol. 3, no. 4, pp. 260–265, Oct. 2013, doi: 10.4236/JTTS.2013.34027.