Souphanouvong University Journal of Multidisciplinary Research and Development
2024,Vol. 10, No. 1, 18-25
https://doi.org/10.69692/SUJMRD100118

Implementation of Shortest Path Finding using Dijkstra's Algorithm

Kalaphath Kounlaxay [1], Southeva Simmavong[1], Vialakone Manivanh[1], Somchith Thongphet[1], Sinlapaxay Sengdala[2], Bounhome Meksavanh[2]

[1]*Department of Computer Engineering, Faculty of Engineering, Souphanouvong University, Laos PDR*
[2] *Information and Technology Center, Souphanouvong University, Laos PDR*

**Correspondence: Kalaphath Kounlaxay, Department of Computer Engineering, Faculty of Engineering, Souphanouvong University, Laos, 020 55672266,* kk_koun@hotmail.com

Submitted: Jun 02, 2023
Revised: Nov 20, 2023
Accepted: Dec 23, 2023

Abstract

Dijkstra's algorithm is one of the algorithms that can be used in the shortest path determination process. Therefore, this research aims to create a simulation to implement how Dijkstra's Algorithm (DA) was used to find the Shortest Path (STP) to travel through any vertex or node by distance vertex and previews vertex and executed or calculated the minimum cost of each edge. The simulation was created based on the shortest route between the source node and the distance node. We have designed the model of pathfinding based on the real location of each city in the northern part of Laos which the cost of each edge is also based on the actual traffic situation and the experiments show that using Dijkstra's algorithm correctly finds the shortest path from the source node to the destination node.

Keywords: *Dijkstra's Algorithm (DA), Pathfinding, Minimum Spanning Tree (MST), Prim's Algorithm (PA), Vertex.*

1. Introduction

Nowadays, research on graph theory is getting a wide range of attention with the development of computer and information technology. Therefore, many various numbers of graph structures and algorithms have been proposed (Sadig, 2019). Dijkstra's STP algorithm has activities such as graphs are used to model connections between objects, people, or entities. There are two main elements: nodes and edges. Nodes represent objects and edges represent the connections between these objects (Amgad et al, 2017). The DA finds the shortest path between a given node which is called the "source node", and all other nodes in a graph. The DA uses the weights of the edges to find the path that minimizes the total distance (weight) between the source node and all other nodes (Wenzheng et al, 2019). Dijkstra STP Algorithm using Prim's algorithm (PA) always starts with a single node and it moves through several adjacent nodes, in order to explore all of the connected edges along the way:

- Choose any initial node in the graph G to a node and let $T = \{u\}$
- Select the connecting route $e = xy$with the smallest distance $x \in T$ and $y \in G - T$ then replace $T \cup \{y\}$
- If $|T| = |V(G)| - 1$ the procedure ends, if not equal, repeat step 2 until equal.

PA method: A minimum spanning tree (MST) is a subset of the edges of a weighted, connected graph that connects all with the minimum possible total edge weight (Suvajit et al, 2014). In other words, it is a tree that spans all the vertices of the original graph and the smallest total weight among all possible spanning trees (Sheikh et al, 2019). There are different algorithms to find the minimum spanning tree of a given graph, but one of the most common ones is PA: Initialize a tree with a single vertex, chosen arbitrarily from the graph (Ayegba et al, 2020). Thus, the method to find the MST of the graph (Eneh et al, 2017), according to Fig 1 which we start by selecting an arbitrary vertex, let's say vertex A. Node A was added in to their tree and mark it as visited. Then, examine all the edges that connect A to other vertices, and it chooses the one with the smallest weight, which is the edge from A to B with weight 2. The edge was added and vertex B to their tree, and mark B as visited. It repeats the process by examining all the edges that connect the vertices in the tree to the vertices not yet in the tree, and choosing the one with the smallest weight. In this case, it is the edge from B to C with weight 6. It was added and vertex C to their tree, and mark C as visited. It continues the process until all vertices are in the tree. The MST has a total weight of 38, which is the smallest possible weight among all spanning trees of this graph.

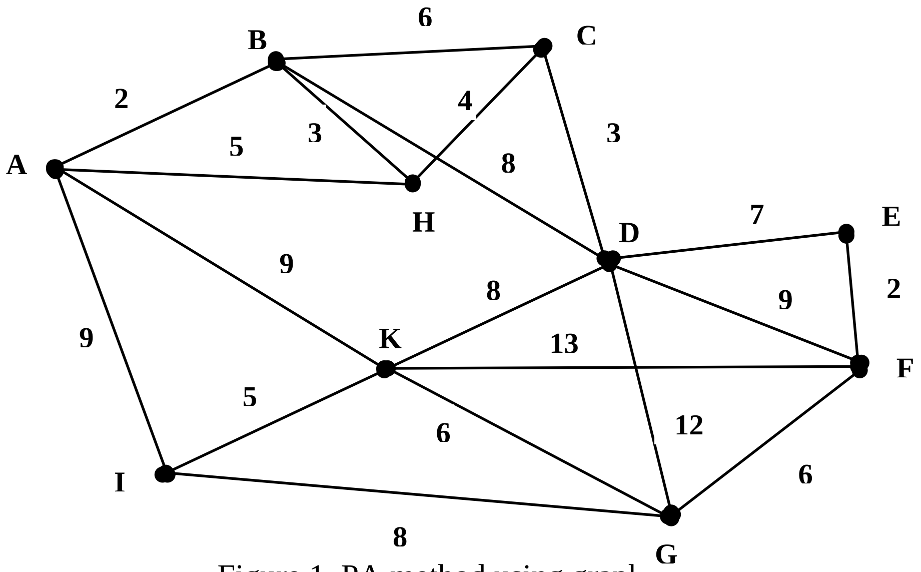


Figure 1. PA method using graph

PA method was calculated as:

1) Choose the starting point, node A is the shortest distance to node B is $T = \{u\} = \{AB\}$

2) Continue to find the shortest path to node H is

Continue to find the shortest path to node C is T = {AB, BH}

Continue to find the shortest path to node D is T = {AB, BH, HC}

Continue to find the shortest path to node E is T = {AB, BH, HC, CD}

Continue to find the shortest path to node F is T = {AB, BH, HC, CD, DE}

Continue to find the shortest path to node G is T = {AB, BH, HC, CD, DE, EF}

Continue to find the shortest path to node K is T = {AB, BH, HC, CD, DE, EF, FG}

Continue to find the shortest path to node I is T = {AB, BH, HC, CD, DE, EF, FG, GK}

3) That mean $|T| = |V(G)| - 1 = 10 - 1 = 9$ equal to number of nodes in step two above

The formula of PA to find the shortest path by distance vertex is:

T = {AB, BH, HC, CD, DE, EF, FG, GK, KI}

$$\begin{aligned} G &= \sum \deg(u_{\min}) = \deg(AB) + \deg(BH) + \deg(HC) + \deg(CD) + \deg(DE) + \deg(EF) \\ &+ \deg(FG) + \deg(GK) + \deg(KI) \\ &= 2+3+4+3+7+2+6+6+5 \\ &= 38 \end{aligned}$$

Therefore, this research aims to design the model or graph of pathfinding based on the real location of each city in the northern part of Laos which the cost of each edge is also based on the actual traffic map, and create the simulation for DA finds the shortest path between a given node as source node and all other nodes in a graph by executed or calculated the minimum cost of each edge.

2. Material and Method

2.1 Material

This research material is the model of

pathfinding based on the real location of each city in the northern part of Laos as the map in Fig 2. which the cost of each edge is also based on the real traffic situation.

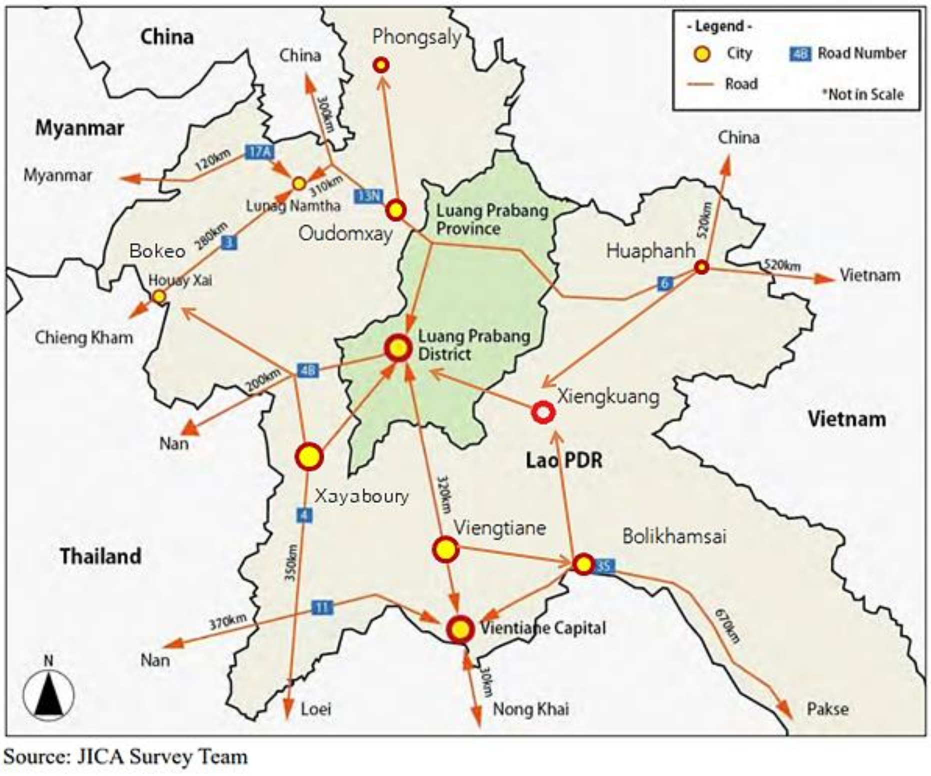


Figure 2. Traffic map of northern part of Laos

The experimental environment for this research is an Intel (R) Core (TM) i5-8400T CPU @ 1.70 GHz, 16 GB RAM, and Windows 10 (version 2019) Personal (64 bit) of the Java development Kit (JDK) and Apache NetBeans IDE 14 to design graphical user interface (GUI) for interaction between user and applications.

2.2 Method

The shortest path problem is an approach to finding the shortest and fastest path or route from a starting point to an ending point (Sathyapriya et al, 2020). Therefore, we have designed a traffic map based on real location of northern part of Laos which consist of 11 provinces as each node and traveling should use to travel to each nearest city by minimum cost or distance vertex. This simulation was used DA which using PA and MST to find the shortest path for this experiment. This simulation was used Java program to simulate the Dijkstra STP finding, as shown in Fig. 3.

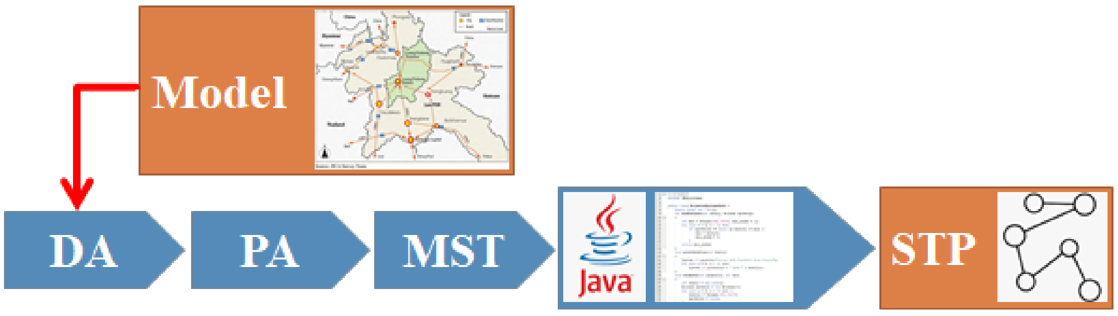
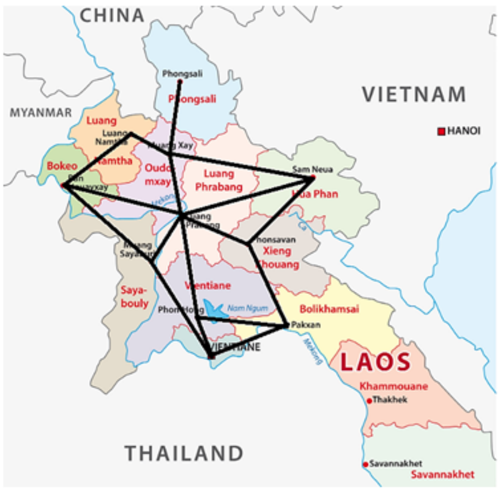


Figure 3. System Overviews

Firstly, we draw vertex, node and edge of each point and indicate start node from number 0 for each province abbreviation, as shown in Fig 4.



Legend: province abbreviation
0. PL: Phongsaly
1. LT: Luangnamtha
2. BK: Bokeo
3. OX: Oudomxay
4. LB: Luangprabang
5. XB: Xayaboury
6. HP: Huaphan
7. XK: Xiengkhuang
8. VT: Vientiane
9. VC: Vientaine Capital
10. BX: Bolikhamxay

Figure 4. Traffic Sketch Map

Given a set of locations and possible roads to be built between pairs of cities with the associated distances, we need to determine the minimum distance road traffic connecting all the locations. We created a table of distances between each province in which distances from each province are in kilometers, as shown in Table 1.

	PL	LT	BK	OX	LB	XB	HP	XK	VT	VC	BX
PL				230							
LT			173	125							
BK		173			451	315					
OX	230	125			192		220				
LB			451	192		125	387	263	244		
XB			315		125					319	
HP				220	387			243			
XK					263		243				214
VT					244					93	85
VC						319			93		170
BX								214	85	170	

Table 1. Distance between each provinces

We use distance of each province from Table 1. to designed a diagram of vertex, node and edge for DA, as shown in Fi 5. Given a traffic and a starting node 0:PL, we want to determine the shortest path to all the other nodes in the traffic connecting or to a specified destination node.

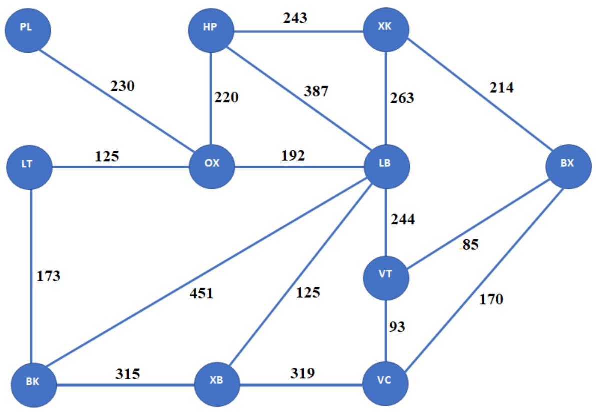


Figure 5. Diagram of vertex/node and edge

MST helps to find a subset of the edges of a connected, edge-weighted undirected graph that connects all the nodes, without any cycles and with the minimum possible total edge weight. Thus, we created MST (Adjacency Matrix), as shown in Table 2.

	PL	LT	BK	OX	LB	XB	HP	XK	VT	VC	BX
PL	0	0	0	230	0	0	0	0	0	0	0
LT	0	0	173	125	0	0	0	0	0	0	0
BK	0	173	0	0	451	315	0	0	0	0	0
OX	230	125	0	0	192	0	220	0	0	0	0
LB	0	0	451	192	0	125	387	263	244	0	0
XB	0	0	315	0	125	0	0	0	0	319	0
HP	0	0	0	220	387	0	0	243	0	0	0
XK	0	0	0	0	263	0	243	0	0	0	214
VT	0	0	0	0	244	0	0	0	0	93	85
VC	0	0	0	0	0	319	0	0	93	0	170
BX	0	0	0	0	0	0	0	214	85	170	0

Table 2. MST

The DA is a greedy algorithm was used to finds the shortest path between a given node (which is called the "source node") and all other nodes in a graph (Chris et el, 2018). This algorithm uses the weights of the edges to find the path that minimizes the total distance (weight) between the source node and all other nodes (Shariff et al, 2020). The activity of Dijkstra's STP algorithm using PA always starts with a single node and it moves through several adjacent nodes, in order to explore all of the connected edges along the way as Fig 6. which DA find the shortest path from vertex 0:PL to every other vertex.

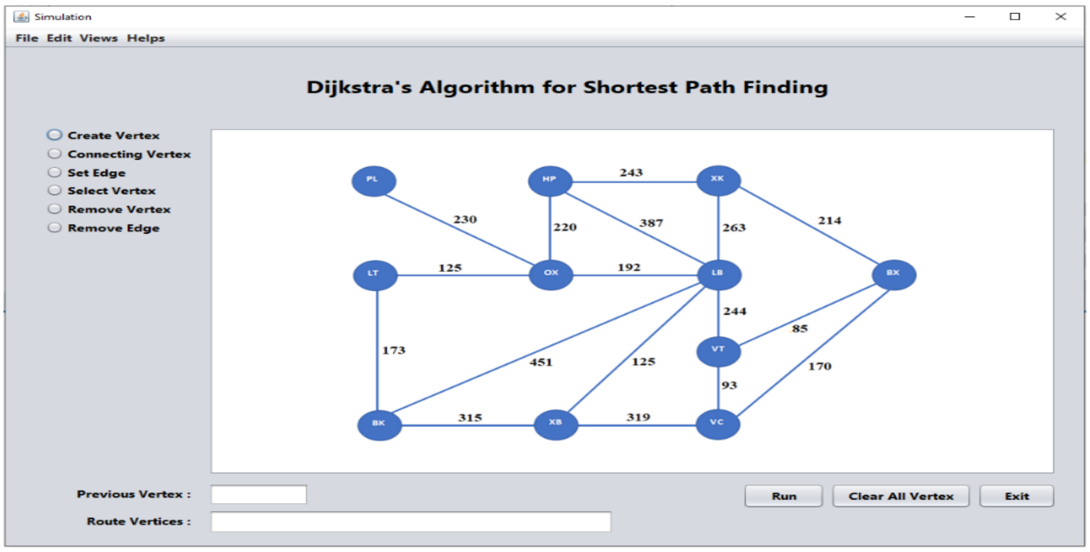
```
Let distance of start vertex from start vertex = 0
Let distance of all other vertices from start = ∞ (infinity)
WHILE vertices remain unvisited
          Visit unvisited vertex with smallest know distance from start vertex (call this "current
          vertex")
          FOR each unvisited neighbor of the current vertex
            Calculate the distance from start vertex
            If the calculated distance of this vertex is less than the know distance
                   Update shortest distance to this vertex
                   Update the previous vertex with the current vertex
            end if
           NEXT unvisited neighbor
           Add the current vertex 0 the list of visited vertices
END WHILE
```

Figure 6. Dijkstra's Algorithm

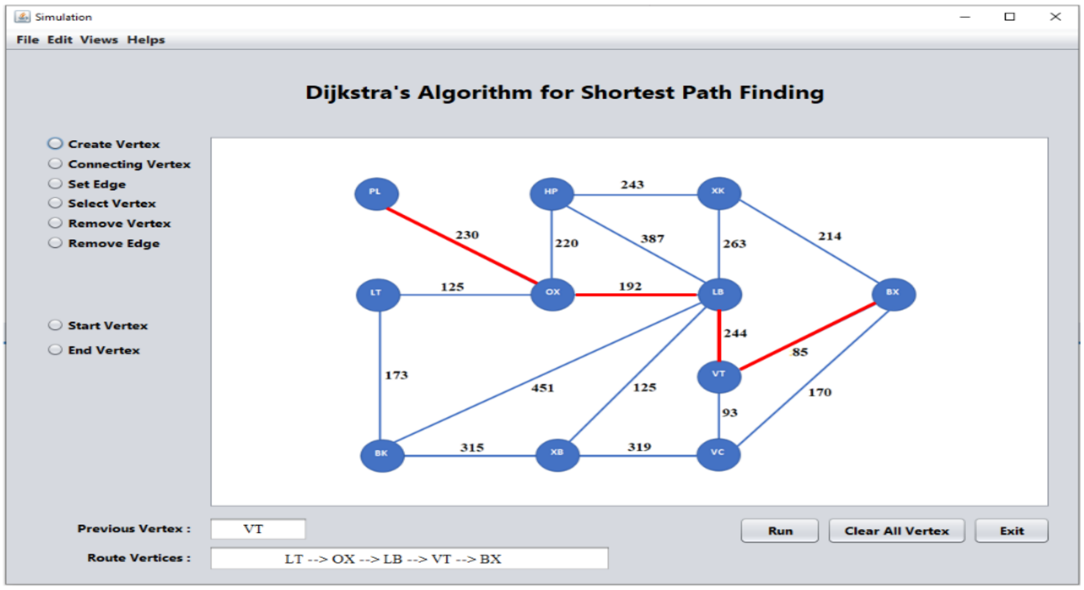
2.3 Experiment and Result

In this section, experiment and result as shown in Fig. 8 where (a) DA simulation form was created by Java program. (b) Find the shortest path from vertex or node 0:PL to every 10:BX vertex. (c) shows shortest path from vertex 0:PL to every vertex in console mode.

(a)



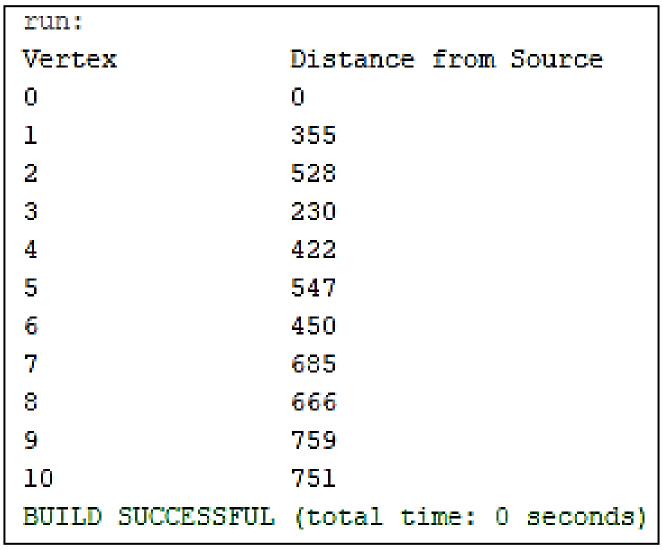
(b)



Previous Vertex : VT

Route Vertices : LT --> OX --> LB --> VT --> BX

(c)



```
run:
Vertex              Distance from Source
0                   0
1                   355
2                   528
3                   230
4                   422
5                   547
6                   450
7                   685
8                   666
9                   759
10                  751
BUILD SUCCESSFUL (total time: 0 seconds)
```

Figure 6. Results

4. Discussion

The result obtained for this project is based on the specifications given such as creating 11 nodes from source to destination in order to travel through each node by using DA, PA and MST. After designing the algorithms and simulation as user interface by using DA, short distance is calculated between two nodes and the shortest distance is return as index value as consistent with Fadhil et al. (2020) reviewed and validated that the source routing protocols have a longer delay because their route discovery takes more time as every intermediate node tries to extract information before forwarding the reply at the same time same thing happens when a data packet is forwarded hop by hop. The limitations of DA like falling into an infinite loop or negative weight can be overcome by improvement as consistent with Niranjane et al. (2020) reviewed and validated that DA was inspired by other algorithm of optimality and so technically it should be viewed as a dynamic programming successive approximation procedure. Therefore, DA can be applied to solve dynamic programming problem with modification to some extent. Search problems can be classified by the amount of information that is available to the search process as consistent with Wint et al, (2019) reviewed that Shortest Path Algorithm is an important problem in graph theory and has applications in communications, transportation, and electronics problems. In graph theory, used many algorithm that solve the shortest path algorithms. According to Wayahdi et al. (2021) who reviewed that DA is a greedy algorithm can be a solution in determining the shortest path in a path or graph with different results. The Greedy algorithm is fast in finding solutions but tends not to find the optimal solution. The DA not only determines the shortest path between the start and end points, but it also determines the shortest paths from the starting point to the other points on a map as consistent with Nurhasanah et al. (2021) demonstrated that factors that must be taken into account in finding the shortest path are road conditions, the length of the path to be traversed, the volume of vehicles, the presence of markets, schools, or other public spaces, and traffic lights. For real conditions, the application of DA must be accompanied by other algorithms so that what is measured can be even more complex.

5. Conclusion

In this paper, we have implemented DA which uses PA and MST to find the STP to travel through any vertex or node by distance vertex and previews vertex and executed or calculated the minimum cost of each edge. This simulation used Java programming and Java Net Bean IDE 14 to simulate the Dijkstra STP finding and the experiments show that using the DA correctly finds the STP from the source node to the destination node. However, for real conditions of DA must be accompanied by other algorithms so that what is measured can be even more complex. For further research, we will extend the research objects such as use the DA to find the STP in a big city by using the more complex Google Map traffic simulator.

6. Conflicts of Interest

We certify that there is no conflict of interest with any financial organization regarding the material discussed in the manuscript.

7. References

Sadig, E. (2019). Parallel Calculation to find Shortest Path by applying Dijkstra algorithm. *Institute of Computer Science University of Tartu.*

Amgad, M., Walid, G., Faizan, U. R., Mohamed, A. R. & Saleh, B. (2017). A Survey of Shortest-Path Algorithms. *Cornell University, Cited as: arXiv: 1705.02044v1 [cs.DS].*

Wenzheng, L., Junjun, L. & Shunli, Y. (2019). *An Improved Dijkstra's Algorithm for Shortest Path Planning on 2D Grid Maps, In: IEEE.*

Suvajit, D., Debasish, P. & Prahakar, A. V. (2014). Development of GIS tool for the solution of minimum spanning tree problem using prim's algorithm. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XL-8.*

Sheikh, I. A & Shahid, U. I. (2019). Minimum Spanning Tree Algorithms and Techniques. *IJCIRAS,* ISSN(O) 2581-5334, Vol 1, Issue 8.

Ayegba, P., Ayoola, J., Asani, E. & Okeyinka, A. (2020). A Comparative Study of Minimal Spanning Tree Algorithms. *International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS)*, pp. 1-4.

Eneh, A. H. & Arize, U. C. (2017). Comparative Analysis and Implementation of Dijkstra's Shortest Path Algorithm for Emergency Response and Logistic Planning. *Nigerian Journal of Technology (NIJOTECH)* Vol. 36, No. 3, pp. 876 – 888.

Sathyapriya, S., Kavin, M. K. & Mythreye, R. S. (2020). Implementation of Dijkstra's Algorithm to Find the Shortest Path in Google Maps. *IJCRT,* Vol. 8, ISSN: 2320-2882.

Chris, K., Alexandre, C., Paulo, X. F. & Miranda, A. V. (2018). Path Value Functions for Which Dijkstra's Algorithm Returns Optimal Mapping. *In: Journal of Mathematical Imaging and Vision Springer Nature.*

Shariff, U. S. & Ganeshan, M. (2020). A Path Finding Visualization Using a Star Algorithm and Dijkstra's Algorithm. *International Journal of Trend in Scientific Research and Development (IJTSRD),* Vol. 5, Issue 1, ISSN: 2456-6470.

Fadhil, M. & Addu, S. (2020). Comparative Study on Bellman-Ford and Dijkstra Algorithm. *International Conference on Communication, Electrical and Computer Networks (ICCECN) Kuala Lumpur, Malaysia.*

Niranjane, P. B. & Amdani, S. Y. (2020). A Survey of Recent Applications of Improved Dijkstra's Shortest Path Algorithm. *International Research Journal of Engineering and Technology (IRJET).* Vol. 07, Issue 11. ISSN: 2395-0056.

Wint, A. K., Kyi, Z. N., Thida, W. & Ei, E. M. (2019). Study of Informed Searching Algorithm for Finding the Shortest Path. *International Journal of Research*, Volume 06 Issue 10, ISSN: 2348-6848.

Wayahdi, M. R., Subhan, H. N. G. & Dinur, S. (2021). "Greedy, A-Star, and Dijkstra's Algorithms in Finding Shortest Path". *International Journal of Advances in Data and Information Systems*, ISSN: 2721-3056.

Nurhasanah, F. Y., Gata, W., Riana, W., Jamil, M. & Saputra, S. F. (2021). "Shortest Path Finding Using Dijkstra's Algorithm". *Penelitian Ilmu Komputer, Sistem Embedded and Logic,* p- ISSN: 2303-3304, e-ISSN: 2620-3553 Vol. 9 (1): 89 – 102.