www.revistageintec.net
ISSN: 2237-0722

revista
GEINTEC
Gestão, Inovação e Tecnologias

Computation Performance Optimization Technique of Shortest Path Routing Algorithm in Networks Using Out-degree

Dr.S. Muthusundari[1]; Dr.R. Jothilakshmi[2]; S. Divya[3]; Dr.A. Kavitha[4]; N. Umamaheswari[5]

[1]Associate Professor, Department of CSE, R.M.D. Engineering College, Kavaraipettai, India.

[2]Associate Professor, Department of IT, R.M.D. Engineering College, Kavaraipettai, India.

[3]M.E., Student, Department of CSE, R.M.D. Engineering College, Kavaraipettai, India.

[4]Assistant Professor, PG Department of Computer Science, D G Vaishnav College, Arumbakkam, Chennai, India.

[5]Assistant Professor, Department of CSE, R.M.K. College of Engineering and Technology, Puduvoyal, India.

Abstract

Objective: *This paper presents a novel technique in Dijkstra's routing algorithm by considering the concept of out degree which will decrease the computational cost and increase the speed of execution.*

Methods: *In this proposed method the time taken by this algorithm that runs as a combination of the input distance is calculated. Conversely, a computation time complexity measures how efficient it is. accuracy of this method. The Shortest Path Problem is the problem of finding a path in a digraph between two vertices or nodes to minimize the sum of the weights of the edges of its constituent. In the existing approach it specifies a number of nodes as the plot unfolds and calculates the quickest route between it and most other nodes. which will complex the speed of execution.*

Findings: *The proposed out degree approach will resolve that drawback by computing two minimum distance from initial table and this operates from the source node and determines the shortest path over the network as a whole. It also guarantees to find a globally optimal solution path with accurate results.*

Novelty: *The algorithm's description and comparison are presented in graphical ways to determine the algorithm's features. The analysis shows that the best route from source to destination will be established which provides the shortest distance. The optimal pathways discovered as a result of the analysis which reduce the distance traveled by the company in shipping products and reduce the time and cost of delivery.*

Key-words: Dijkstra's Algorithm, Out Degree, Shortest Path, Optimal Solution, All-Pairs Shortest Path.

Received: 12.04.2021 – Accepted: 04.05.2021

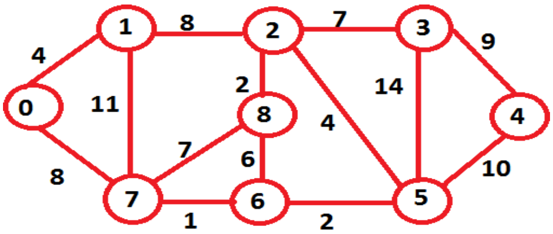
1. Introduction

In Graph theory, among the most frequently discussed subjects and researched by researchers is the Shortest Path Problem(SPP). Thus the core problem would be to discover the quickest distance in a directed graph from a source vertex S to a single destination vertex D and to calculate the corresponding minimum value. Dutch computer scientist Edsger Dutch Dijkstra a network search technique was proposed in 1959 that could be used to solve the shortest single-source path problem for any graph with a non-negative path value. Dijkstra's is a really well strategy for finding the shortest path and also known as "Label Algorithm". It selects repeatedly from the unselected vertices, vertex v closest to source s and announces the distance to be the actual shortest distance between s and v. The edges of v are then verified to determine if they can achieve their goal using v, and then the relevant outward edges are examined. Determining the easiest distance among a goal state and a destination vertex is a difficult problem. Vertex is frequently utilized in a variety of applications, including robot outriders, Routing method for computer networks, user interface design, route guidance, and so on. For nearest neighbor issues, two algorithms are provided. One is to determine the all-pairs shortest path (APSP) for n-vertex m-edge directed networks characteristic of successful feature vectors with O(logn) edges between them in O(n2logn +nm) latency. It's quite another to discover the straight route that is the quickest (SSSP) running in O(n) time for graphs that can be. reduced by simple transformations to the trivial map. For some special classes of graphs, these algorithms are optimally efficient in The earlier reaches O(n2), which really is a faster deadline for finding APSP, though it also achieves O(n), which is a faster time limit for finding SSSP. The latter can be used to find APAS, which also achieves runtime **O(n2).** the problem that occurs during traffic conditions between cities from time to time and there are usually huge quantities of request that need to find a solution quickly so that this issue has been rectified by using the Dijkstra's algorithm in the shortest possible way, the main goals are to find the low cost of implementation. Dijkstra's algorithm was presented to overcome the minimization problem for fixed graphs from a central entity. Noting that on the specified transport network, an upper bound of the distance between two nodes may be determined in advance [1]. A new directed backward version of the Single Source Shortest Path algorithm. This algorithm agrees that the approaching adjacency list of the loads in the specific graph vertex showed up in the request expansion. In addition, the new algorithm's likelihood also takes O(n) time. This is an equally improved version of Wilson's exponential and polynomial low probability. O(n)'s greatest after-effect is the running time of forward-biased strategy algorithm, which is Wilson's most perfect forward-biased SSSP effects [2]. The main objective of testing the general aspects of Dijkstra's Algorithm in solving the shortest path problem. This has some major drawbacks such as being unable to manage negatives edges, which contribute to acyclic graphs, and the shortest path can't be reached most often [3]. There are different shortest path problem algorithm and implemented a method for solving or overcoming difficulties. Although the problem of minimizing the error besides transport systems still exists, they have did significant adjustments and developed a new algorithm, namely modifying Dijkstra's shortest route to use a clustering technique with main memory and comparing it to the adaptive technique to demonstrate its highest performance [4,5-7]. The difficulty of Monarch Paints Nig. Ltd in transporting their commodities at thier manufacturing plant to sales outlets was examined using Dijkstra's algorithm, and the shortest path was determined. A package called TORA (version 2006) was utilized to do the analysis [8]. Many studies have been conducted to improve Dijkstra's algorithms; those studies are primarily focused on finding the shortest path through one citation to one endpoint in major highway networks. This methodology was employed in the Guidance System, which is the most recent technology, to make it more trustworthy and accurate. The majority of the research has been focused on improving Dijkstra's Tool to make it a more adaptable and efficient cheapest circuit method [9].

2. Results and Discussion

This proposed algorithm presents a novel technique in Dijkstra's routing algorithm.

Figure 1 - Dijkstra's Algorithm for Undirected Weighted Graph



By considering the concept of out degree which will decrease the computational cost and increase the speed of execution. An optimization method in which mainly improved the selection of nodes for the shortest path and the structure and organization of data storage. The improved algorithm was obtained through comparison and analysis, which reduced the storage space and increased operational efficiency. We concluded by optimizing the complexity of the system, time complexity and combination of storage reduced storage space, reduced data redundancy and significantly improved running rate [10]. Beginning while iteration takes O seconds to execute (V). Determine out

Min of the heap is logV in each iteration of the while looping. The deepest for loops iterates for each node adjacent to the current node, taking O seconds to complete (E). As a result, the temporal complexity of a particular algorithm is O((V + E)*log(V) = O(E* log(V)) [11]. The main function of the algorithm includes to find the out degree, adjacency matrix and comparing the vertices. The proposed algorithm is given below.

2.1. Proposed Methodology

A. Proposed Algorithm

```
1. Initialize the graph.
2. Find the out degree for all vertices. Find the Max[out degree] and select the max[out degree] as source node.
      a) If more than one max out degree is generated then calculate the cost factor for their adjacent vertex and find the [minimum_cost_factor_vertex]
3. Find adjacency for source node and form the initial table with the corresponding adjacency vertices of source node.
4. Choose two minimum distances from the initial table.
    a) Find the adjacency vertices for those two minimum distance vertices.
    b) Add and compare the weights from source node and create a new_table.
    c) Mark the min_distance vertex as known.
    d) Repeat from Step 4 until all vertexes are known
```

B. Working Principles of the Algorithm

Consider the Undirected Weighted Graph as an example and analyze the working principle of the proposed algorithm with existing shortest path algorithm.

Step1: Initialize the Graph

The undirected weighted graph considered is initialized with 9 vertex starting from 0 to 8 with different weights so that minimum single source shortest path can be opted.

Step2: Find the out degree for all vertices. Find the Max[out degree] and select the max[out degree] as source node.

a) If more than one max out degree is generated then calculate the cost factor for their adjacent vertex and find the [minimum_cost_factor_vertex]

Table 1 - Out Degree for all Vertices

Vertex	Out Degree
0	2
1	3
2	**4**
3	3
4	2
5	**4**
6	3
7	**4**
8	3

From the [Table 1], vertex 2, vertex 5 and vertex 7 has the highest out degree. The cost factor is calculated to choose optimized vertex as minimum one.

The Out Degree for all the vertex is calculated, here 3 values of the same highest out degree is found so by calculating the cost factor for the opted vertex we can find one optimized vertex. The major objectives for studying Dijkstra's Algorithm, Floyd-Warshall Algorithm, Bellman-Ford Algorithm, and Genetic Algorithm (GA) in solving the shortest path issue with the Genetic Algorithm paradigm are as follows: for finding the perfect solution to the shortest path problem. The results, as well as the communication overhead of studying them the Dijkstra's, Floyd-Warshall and Bellman-Ford algorithm [12]. Global positioning system used to add new features to Dijkstra's algorithm. In this paper the location parameter is applied to the Dijkstra's algorithm using Global Positioning System. From this, at any point the current position is retrieved. This range orientation attribute can be used to compute the location between two nodes using the actual state [13].

Table 2 - Cost Factor

VERTEX:2	**8+7+4+2=21**
VERTEX:5	10+14+4+2=30
VERTEX:7	11+7+8+1=27

From the [Table 2], Vertex 2 is chosen as the optimized vertex. Hence it is declared as the Source Vertex.The source vertex is declared as vertex 2 from vertex 2 a shortest path is calculated to all other vertex in the initialized graph, so that minimal cost for travelling from single source to all other vertex is implemented.

***Step3*:** Find adjacency for Source Vertex and form the initial table with the corresponding adjacency vertices of Source Vertex. The adjacency calculation starts from the source vertex and then it is traversed for all other vertices.

Table 3 - Adjacency of Source Vertex 2

Adjacency Vertex	1	3	5	8
Weight	8	7	4	2

To find the adjacency matrix of source vertex 2 from initialized graph, from the [Table 3], we infer that source vertex 2 has four adjacency vertexes with different weights, with these weights a initial table is constructed. Adjacency vertices that are calculated from source vertex is referred as base for creating the initial table.

Table 4 - Initial Table

Vertex	Known	Distance	Path
0	0	∞	-
1	0	8	2
2	*	*	*
3	0	7	2
4	0	∞	-
5	**0**	**4**	**2**
6	0	∞	-
7	0	∞	-
8	**0**	**2**	**2**

To form the Initial Table by considering the [Table 2], from [Table 4] two minimum cost is identified and the adjacency matrix is calculated. From the initial table vertex 5 and vertex 8 is chosen as minimum cost vertex and further steps for calculating the shortest path is implemented.

Step 4: Choose two minimum distances from the initial table.

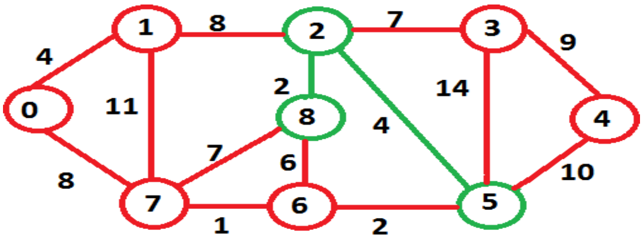
a. Find the adjacency vertices for those two minimum distance vertices.

b. Add and compare the weights from source node and create a new table.

c. Mark the min_distance vertex as known.

d. Repeat from Step 4 until all vertexes are known.

Figure 2 - Two Minimum Distances from the Initial Table



Received: 12.04.2021 – Accepted: 04.05.2021

From initial table the minimum distance is found from source node 2 to vertex 8 with distance 2 and vertex 5 with distance 4. The weights of the respective vertex are correlated with the source vertex so they are added to the original weights of the vertices.

Table 5 - Weights from Source Vertex and Comparing the Weights of the Vertices.

vertex:5	2→Source node	3→4+14=18	4→4+10=14	**6→4+2=6**
vertex:8	2→Source node	6→ 2+6=8	7→2+7=9	-

From [Table 5] The weights of each node from the source vertex is analyzed. The minimum weight is chosen for the further findings of shortest path that are compared.

Table 6 - Iterations are Carried out to Perform the Proposed Algorithm

ITERATION I				ITERATION II			
Vertex	Known	Distance	Path	Vertex	Known	Distance	Path
0	0	∞	-	0	0	∞	-
1	0	8	2	***1***	***0***	***8***	***2***
2	*****	*****	*****	***2***	*****	*****	*****
3	***0***	***7***	***2***	3	1	7	2
4	0	14	5	4	0	14	5
5	1	4	2	5	1	4	2
6	***0***	***6***	***5***	6	1	6	5
7	0	9	8	***7***	***0***	***7***	***6***
8	1	2	2	8	1	2	2
ITERATION III				ITERATION IV			
Vertex	Known	Distance	Path	Vertex	Known	Distance	Path
0	***0***	***12***	***1***	0	***1***	12	1
1	1	8	2	1	***1***	8	2
2	*****	*****	*****	***2***	*****	*****	*****
3	1	7	2	3	***1***	7	2
4	***0***	***14***	***5***	4	***1***	14	5
5	1	4	2	5	***1***	4	2
6	1	6	5	6	***1***	6	5
7	1	7	6	7	***1***	7	6
8	1	2	2	8	***1***	2	2

From the [Table 6- Iteration I] vertex 3 and vertex 6 is chosen as minimum cost vertex and further steps for calculating the shortest path is implemented. The experimental result showing that the three problems were successfully resolved and Dijkstra's "**Label Algorithm"** was evaluated to demonstrate the algorithm's inadequacies and suggested improved methods. This subject put forward the improved algorithm and carried out a series of targeted experiments. Results of experiments suggest that the improved algorithm can not only prove undigraph shortest path problem it can also solve digraph’s shortest path problem [14]. From Iteration I, the minimum distance is found, vertex 3 with distance 7 and vertex 6 with distance 6.

Comparing the Vertices:

VERTEX:3	2→Source node	4→7+9 =16	5→7+14=21
VERTEX:6	5→Already visited	7→6+1 = 7	8→6+6=12

From [Table 6- Iteration II], the minimum distance is found, vertex 1 with distance 8 and vertex 7 with distance 7. Vertex 1 and Vertex 7 is chosen as minimum cost vertex and further steps for calculating the shortest path is implemented. Dijkstra's algorithm has been modified to increase precision and effectiveness in order to find a superior shortest path and potentially reduce intricacy. In order to compare QDA and MDSC, a new technique named Adaptation In Dijkstra's Algorithm And Determine The Shortest Path For ‘N' Nodes Considering Constraint (MDSC) was presented. [15,16-19].

Comparing the Vertices:

Vertex: 1	0→8+4=12	2→Already visited	7→8+11=19	-
Vertex: 7	0→7+8=15	1→7+11=18	6→Already visited	8→7+7=14

From [Table 6- Iteration III], the minimum distance is found, vertex 0 with distance 12 and vertex 4 with distance 14. All the Vertices are known vertex from the [Iteration IV]. The process is finished. The cost calculation for the single source shortest path is given below in [Table 6- Iteration IV].

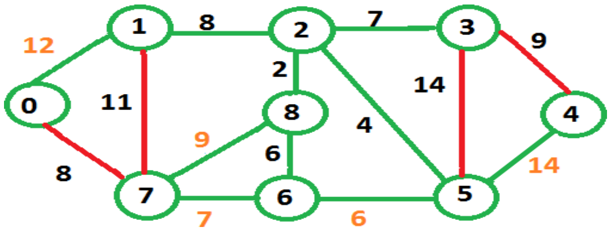
Comparing the Vertices:

Vertex: 0	1→Already visited	7→12+8 =20
Vertex: 4	3→14+9 =23	5→Already visited

Table 7 - Final Distance via Path from Single Source Shortest Path

Vertex	Distance	Path
0	12	1
1	8	2
2	*	*
3	7	2
4	14	5
5	4	2
6	6	5
7	7	6
8	2	2

Figure 3 - Result of Iteration 4



2.2. Result Analysis

This result analysis is designed on the comparison of the following parameters.

1. Number of steps performed.
2. Complexity and efficiency Comparison.
3. Time Complexity.

I. Number of steps performed: For the analysis of the new algorithm's performance, the new method is compared to the existing method using the following characters from table 7.

II. Complexity and efficiency Comparison: The Proposed algorithm is evaluated with existing method, with 50 different Graph Samples with different no of vertices 6,7,9,10,25 and 50, and their result performance was analyzed. From that the complexity and efficiency comparison is given in the Table 8 for performance consideration. Hence the result was shown in the table 8. The Proposed algorithm is much better than the Existing method.

III. Time Complexity: When looking at the temporal complexity of an approach, there are three possibilities:

- Best-Case
- Average-Case
- Worst-Case

Best-case: This is the degree of difficulty in resolving the problem in order to obtain the best feedback. The highest probability in our circumstances would be to look for the fastest route. In the first iteration, the vertex with the min distance would be discovered.

Average-case: This is the typical difficulty of the problem to solve. The dispersal of the quantities in the raw data is used to define this complication. Perhaps this isn't the best use of the suggested method's min distance traversal.

Worst-case: This is the difficulty of tackling the problem both for smallest n parameter. The worst-case scenario in our suggested strategy is to look for (n-1) min distance, and that's the last component with in sequence. Every facet can be interconnected to (V-1) vertices, resulting in a total of V - 1 nearby edges for each vertex. Assume D stands for the V-1 edges that connect each vertex. The weight of each neighboring vertex in the min heap must be found and updated O(log(V)) + O(1) = O(log(V)). Hence from step1 and step2 above, the time complexity for updating all adjacent vertices of a vertex is D*O(logV).

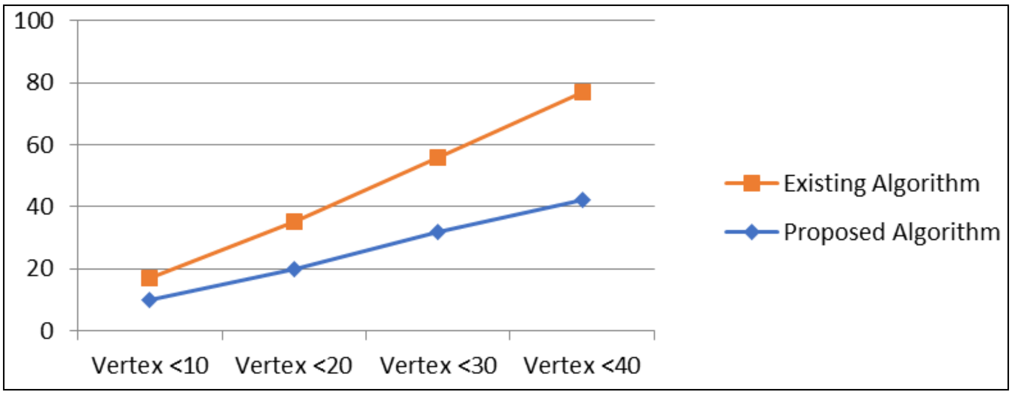
Time complexity for all V vertices is

$$V * (D*\log V) \rightarrow O(VD\log V)$$

Table 8: Comparison of Parameters with Existing & Proposed Algorithm

No.	Parameters	Existing	Proposed
1.	TABLE SIZE	For Given N Vertex, N+1 Table is required.	For Given N Vertex, N-1 and N- 2 Table, In some cases N table is required.
2.	INITIAL TABLE	YES	YES
3.	VERTEX COMPARISON	One minimum Cost is chosen	Two minimum Cost is chosen

Figure 4 - Graph for the Performance of the Proposed Algorithm



From [Figure 4] it is inferred that the Proposed Algorithm calculates less number of iterations when compared to the existing algorithm with respect to the number of vertices. Hence the performance of the proposed algorithm finds the global optimal solution path with accurate results.

3. Future Enhancement

The functionalities used in this proposed algorithm is executed in basic languages and implemented. These functionalities will be simulated as a package for further development.

4. Conclusion

This research paper, The necessity to reduce the distance and time from the start node of the graph to the multitude of separate nodes inside the information given connectivity drove the challenge of determining the shortest. As a result, the suggested Methodology fixes a predefined threshold as a source node and finds the shortest path from source to all other nodes, reducing the complex greater efficiency and effectiveness by evaluating out degree approach, which overcomes the downfalls and also assures to find a best feature highly persuasive with reliable numbers.

References

Md. Almash Alam, Md. Omar Faruq. Finding Shortest Path for Road Network Using Dijkstra's Algorithm. *Bangladesh Journal of Multidisciplinary Scientific Research.* 2019; 1(2). https://doi.org/10.46281/bjmsr.v1i2.366

Jasmine Priskilla, K. Arulanandam. A Backward Single Source Shortest Path Algorithm for Directed Graph. *International Journal of Innovative Technology and Exploring Engineering (IJITEE).* 2019; 8(10). 10.35940/ijitee.J1135.0881019.

Jogamohan Medak, Partha Pratim Gogoi. Review and Analysis of Single-Source Shortest Path Problem Using Dijkstra's Algorithm, *IOSR Journal of Computer Engineering (IOSR-JCE),* 2018; 20(2), Ver. II, 10-15. www.iosrjournals.org

Mazhar Iqbal, Kun Zhang, Sami Iqbal, Irfan Tariq. A Fast and Reliable Dijkstra Algorithm for Online Shortest Path. *SSRG International Journal of Computer Science and Engineering by SSRG - IJCSE Journal.* 2018; 5(12). 10.14445/23488387/IJCSE-V5I12P106

Dilip Kumar Adhwarjee. Finding the shortest path by Adhwarjee's algorithm and comparison of this powerful method with Dijkstra's algorithm. *International Journal of Mathematics in Operational Research, Inderscience Enterprises Ltd.* 2018; vol.13(2). 10.1504/IJMOR.2018.094058

Mohan, S. Prabhavathi, R. Muthukumar, N. Sangeetha, C. Arockiyadass, S. Dickson. Minimum spanning with dijkstra's algorithm. *International Journal of Mathematical Archive- 9(8).* 2018; ISSN 2229-5046. www.ijma.info ISSN 2229–5046

Singh Y., Sharma S., Sutton R., Hatton D. Towards use of Dijkstra Algorithm for Optimal Navigation of an Unmanned Surface Vehicle in a Real-Time Marine Environment with results from Artificial Potential Field. TransNav, *The International Journal on Marine Navigation and Safety of Sea Transportation.* 2018; Volume 12, Number 1. 10.12716/1001.12.01.14

Ojekudo, Nathaniel Akpofure & Akpan, Nsikan Paul. Anapplication of Dijkstra's Algorithm to shortest route problem. *IOSR Journal of Mathematics.* 2017; 13(3)Ver.1. 20-32 www.iosrjournals.org

Anjali Jain, U. Datta, Neelam Joshi. Implemented modification in Dijkstra's Algorithm to find the shortest path for N nodes with constraint. *International Journal of Scientific Engineering and Applied Science (IJSEAS).* 2016; Volume-2, Issue 2. www.ijseas.com

Arjun RK, Pooja Reddy, Shama, M. Yamuna. Research on the Optimization of Dijkstra’s Algorithm and its Applications. *International Journal of Science, Technology & Management*. 2015; Volume No 04, Special Issue No. 01. 10.1109/FSKD.2010.5569452

Vishal Bharte, Kaustubh Patil, Lalit Jadhav, Dhaval Joshi. Bus Monitoring System Using Polyline Algorithm. *International Journal of Scientific and Research Publications*. 2014; 4(4). http://www.ijsrp.org/research-paper-0414/ijsrp-p2829.pdf

Kairanbay Magzhan, Hajar Mat Jani. A Review and Evaluations of Shortest Path Algorithms. *International Journal of Scientific & Technology Research*, 2013; 2(6): 99-104. https://www.researchgate.net/publication/310594546

Pankaj Verma, J.S Bhatia. Design And Development Of GPS-GSM Based Tracking System with Google Map Based Monitoring. *International Journal of Computer Science, Engineering and Applications (IJCSEA).* 2013; Vol.3, No.3. 10.5121/ijcsea.2013.3304

Wang Shu-Xi*. The Improved Dijkstra's Shortest Path Algorithm and Its Application. *Procedia Engineering.* 2012; 29:1186-1190. 10.1016/j.proeng.2012.01.110

Guo, L., Yang, Q., & Yan, W. Intelligent path planning for automated guided vehicles system based on topological map. *IEEE Conference on Control, Systems & Industrial Informatics.* 2013; INSPEC Accession Number: 13372008. 10.1109/CCSII.2012.6470476

Dobrilovic, D., Jevtic, V., Beker, I., & Stojanov, Z. Shortest-path based model for warehouse inner transportation optimization. *7th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI).* 2012; 10.1109/SACI.2012.6249977;12907435

Ravi, N., Sireesha, V. Using Modified Dijkstra’s Algorithm for Critical Path Method in a Project Network. *International Journal of Computational and Applied Mathematics*. 2010; Volume 5 number2, 217-225. Research India Publications, http://www.ripublication.com/ijcam.htm

James B. Orlin, Kamesh Madduri, K. Subramani, M. Williamson. A Faster Algorithm for the Single Source Shortest Path Problem with few distinct positive lengths, *Journal of Discrete Algorithms.* 2010; Volume 8, Issue 2, Pages 189-198. https://doi.org/10.1016/j.jda.2009.03.001

Xiao, J.X., & Lu, F.L. An improvement of the shortest path algorithm based on Dijkstra algorithm. *In 2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE).* 2010; INSPEC Accession Number: 11259341. 10.1109/ICCAE.2010.5451564, 11259341