IOSR Journal of Mathematics (IOSR-JM)
e-ISSN: 2278-5728, p-ISSN: 2319-765X. Volume 13, Issue 3 Ver. 1 (May. - June. 2017), PP 20-32
www.iosrjournals.org

Anapplication of Dijkstra's Algorithm to shortest route problem.

Ojekudo, Nathaniel Akpofure (PhD) [1]& Akpan, Nsikan Paul (PhD) [2],
[1]*Department of Mathematics and Computer science, Ignatius Ajuru University of Education,Port Harcourt, Rivers State, Nigeria.*
[2]*Department of Mathematics and Statistics, University of Port Harcourt, Nigeria.*

Abstract: *Network analysis is an essential tool used in transport sector, information sector and also for the flow of matter and energy. Thus, choosing efficient route is essential for businesses and industries which aid distribution of goods and services optimally. This research addresses the problem of Dominion Paints Nig. Ltd in transporting their products from their production plant to stores of sales by presenting analysis of the shortest path using Dijkstra's Algorithm and it was concluded that the best paths found from the analysis will save the company less distance in transporting the paints and minimize time and cost of fueling their vehicles. The analysis shows that the best route which provides the shortest distance will be from node 1 – 3 – 5 – 8 (Aluu – Rumuosi – Location – Mile 3), when transporting from Aluu (the production location) to Mile 3 the company's major sales point with a total distance of km. A TORA software (version 2006) was used in the analysis.*

Keywords: *Directed Network, node, activity, Dijkstra's Algorithm,Permissible route,*

I. Introduction.

Networks arise in numerous settings and in a variety of ways, ranging from transportation, electrical, and communication networks which pervade our daily lives. Network representations are widely used for problems in diverse areas as oil transportation, production, distribution, project planning, facilities location, resource management, and financial planning, etc. It provides a powerful visual and conceptual aid that is used to portray the relationships between the components of systems in virtually every field of scientific, social, and economic endeavour. One of the most exciting developments in operations research (OR) in recent years has been the unusually rapid advance in both the methodology and application of network optimization models. A number of algorithmic breakthroughs have had a major impact on computer science concerning data structures and efficient data manipulation. Consequently, algorithms and software are now available and are used to solve several problems on a routine basis that would have been completely impossible twoorthreedecadesago.

The model takes in all aspects of the business, helping management to plan and decide differentlevels at the various stages in the industry, e.g. knowing what to pay and what to charge.A network representation is essential in an industry because it helps to determine and monitor the flow of goods from the industry to its final destination. Taking the crude oil as an example, network representation can help determine the various stages, from crude oil purchase, shipping to refineries, refining it, to sending it for storage and distribution for sale purposes.

Network optimization is a special type of linear programming model. Network models have three main advantages over linear programming.

1.1 Background of Study.

Networks are necessary for the movementof people, transportation of goods, communicate information and control of the flow of matter and energy. Network application is quite vast.Phenomena that are represented and analyzed as networks are roads, railways, cables, and pipelines. In networking, the cost, time, and complex nature of network increases in different kinds of network-based systems, e.g. Television cable networks, Telephone networks, Electricity supply networks, Gas pipe network and water supply system. Therefore, the cost, time, and complexity of network are considered greatly in solving networking problems. A graph is a mathematical abstraction that is useful for solving different networking problems. Finding the shortest paths plays an important role in solving network based systems. In graph theory,a number of algorithms can be applied for finding the shortest path in a graph based network system. This reduces the complexity of the network path, the cost, and the time to build and maintain the network based systems.

In recent times, planning efficient routes is very essential for business and industry with applications as varied as product distribution. It is essential for products or services to be delivered on time at the best price using the shortest available route. The shortest route network model is an efficient route that can be used in planning.This network model is applied in telecommunications and transportation planning. The shortest pathproblem involves finding the shortest possible path or route from a starting point to a final point.

Networks are used in general to represent shortest path problem. A graph which is used to solve such problems contains sets of vertices and edges. Pairs of vertices are connected by edges, while movement from one vertex to other vertices can be done along the edges. A graph can be directed or undirected depending on the movement along the edges, either walking on both sides or on only one side.Lengths of edges are often called weights which are normally used to calculate the shortest path from a particular point to another point. In the real world,the graph theory can be applied to different scenarios.A practical example is map representation using a graph, where vertices and edges represent cities and routes that connect the cities respectively. One-way routes are directed graphs, while routes that are not one-way are undirected. There are different types of algorithms that are used to solve shortest path problems. However, only the Dijkstra's algorithm will be discussed in this project.

1.2 Aim and Objectives of Study

The aim of this project is to determine the shortest route from the production plant of a local paint company(Dominion Paints Nigeria limited) to 7 different dealers in the state with a permissible route.

II. Literature Review

Dijkstra (1959) proposed a graph search algorithm that can be used to solve the single-source shortest path problem for any graph that has a non-negative edge path cost.This graph search algorithm was later modified by Lee in 2006 and was applied to the vehicle guidance system. Thisvehicle guidance system is divided into two paths; namely, the shortest path and the fastest path algorithms(Chen et al., 2009).While the shortest path algorithm focuses on route length parameter and calculates the shortest route between each OD pair, the fastest path algorithm focuses on the path with minimum travel time. The future travel time can be predicted based on prediction models using historical data for link travel time information which can be daily, weekly or even a session.

Meghanathan (2012) reviewed Dijkstra's algorithm and Bellman-Ford algorithm for finding the shortest path ina graph. He concluded that the time complexity of Dijkstra's algorithm is O ($|E|*log\ |V|$) while the time complexity of the Bellman-Ford algorithm is O ($|V||E|$).

Lili Cao et al (2005)concluded that the search for the shortest path is an essential primitive for a variety of graph-based applications, particularly those on online social networks. An example is the LinkedIn platform where users perform queries to find the shortest path "social links" connecting them to a particular user to facilitate introductions. This type of graph query is challenging for moderately sized graphs but becomes computationally intractable for graphs underlying today's social networks, most of which contain millions of nodes and billions of edges. They propose *Atlas*, a novel approach to scalable approximate shortest paths between graph nodes using a collection of spanning trees. Spanning trees are easy to generate, compact relative to original graphs, and can be distributed across machines to parallelize queries. They demonstrate its scalability and effectiveness using 6 large social graphs from Facebook, Orkut, and Renren, the largest of which includes 43 million nodes and 1 billion edges. They describe techniques to incrementally update Atlas as social graphs change over time. They capture graph dynamics using 35 daily snapshots of a Facebook network and show that Atlas can amortize the cost of tree updates over time. Finally, they apply Atlas to several graph applications and show that they produce results that closely approximate ideal results.

Wadhwa *(*2000)stated that researchers have targeted a Network Design Problem (Cable and Trench Problem), which involves a trade-off between utilization costs and capital costs for network construction. A larger network, (the shortest path tree) may cost more to build but may reduce utilization costs by including more attractive origin-destination paths. Conversely, a smaller network, (minimum spanning tree) may increase the utilization costs. A heuristic has been provided which gives us optimal or near optimal solutions. This heuristic is an adaptation of the Savings algorithm given by Clarke and Wright in 1964, for solving a vehicle routing problem. The heuristic provides us good solutions which can be used as upper bounds for branch and bound methods, giving us the optimal solutions in lesser times than that given by branch and bound without the upper bounds.

Pallottino and Scutella (1997)reported on Shortest Path Algorithms in Transportation models: classical and innovative aspects. They reviewed theshortest path algorithms in transportationsin two parts. The first part includes classical primal and dual algorithms which are the most interesting in transportation, either as a result of theoretical considerations or as a result of their efficiencies, and in view of their practical use in transportation models. They discussed the Promising re-optimization approaches involved. The second part includes dynamic shortest path problems that arise frequently in the transportation field. They analyzed the main features of the problems present under suitable conditions on travel time and cost functions, a general "chronological" algorithmic paradigm,called *Chrono-SPT*.

Curtin (2007)reported that the network is a compelling research paradigm because its form can intuitively represent complex systems. The ability to comprehend the complex systems around us

i.e.transportation, communication,or interactions at the cellular level, is of great importance in an increasingly complex world. The spatial nature of Networks gives opportunities for research in network analysis that could prove valuable across a wide range of disciplines.

Ahmat (2005)studied extensively in association with complex communication networks. The study described basic concepts of graph theory and their relation to communication networks. The study also presented some optimization problems that are related to routing protocols and network monitoring and showed that many of the optimization problems are NP-Complete or NP-Hard. Finally, it described some of the common tools used to generate network topologies based on graph theory.

Xiaotakes a problem of online answering shortest path queries by exploiting rich symmetry in graphs. The Dijkstra is the most famous and widely used algorithm to solve the shortest path problem because it is fast and uses heap data structures for priority queues shortest path queries which are required in many applications.
Steinhardt (2006)concludes that Dijkstra's Algorithm traversal algorithms are specialized for finding the shortest paths between vertices on the graph.

Andrew V. Goldberg (2008)studies Point-to-Point (P2P) Shortest Path Algorithms. In recent times, good development has taken place on the Point-to-Point shortest path algorithms with pre-processing. The algorithms proved to be efficient in practice on road networks and some other kinds of graphs. There are some questions, particularly theoretical, that remain open. Therefore, atheoretical justification for these algorithms is necessary. Two possible directions are proving good worst-case bounds for the algorithms on special graph types or proving average-case bounds on graph distributions. For the latter, random grids are interesting candidates. Computing reaches is another set of open question. One can modify a standard all-pairs shortest path algorithm to compute reaches in the same time bound, which is $O^*(n2)$ for sparse graphs. Since the size of the output for the all-pairs problem is $(n2)$, there is limited room for improvement. As reaches need only one value per vertex, this argument does not apply to the problem of computing reaches. An interesting open question is the existence of an algorithm that computes reaches – or provably good upper bounds on reaches – in $O(n2)$ time.

Borgwardt and Kriegel (2005)defined graph kernels based on shortest paths, which are polynomial to compute, positive definite and retain expressivity while avoiding the phenomenon of "tottering". In experiments on classifying graphs model of proteins into functional classes, they outperformed kernels based on random walks significantly. The shortest-path kernels prevent tottering. The definition of a path would be violated if the same edge appears twice in the same shortest path. Subsequently, artificially high similarity score caused by repeated visiting of the same cycle of nodes are prohibited in our graph kernel. The shortest-path kernel as described in this article is applicable to all graphs on which Floyd-Warshall can be performed. Floyd-Warshall requires that cycles with negative weight do not exist. This condition generally holds if edge labels represent a distance which is the case in most molecular classification tasks.

Shirinivas et al (2010)presented the importance of graph theoretical ideas in various areas of computer applications like Shortest path algorithm in a network, Finding a minimum spanning tree, Finding graph planarity, Algorithms to find adjacency matrices, Algorithms to find the connectedness, Algorithms to find the cycles in a graph, Algorithms for searching an element in a data structure (DFS, BFS).

Sommer (2010)investigated shortest path query processing in networks both from a theoretical and a practical point of view. An experimental study was performed using road transportation network. The study revealed a simple and general method based on Voronoi duals to efficiently support the shortest path queries in undirected graphs with very low pre-processing overheads and competitive query times, at the cost of exactness. This method was proved to be effective on a variety of graph types while remaining a reasonable alternative to existing exact methods specifically designed for transportation networks.
Abbasi et al (2011)considered the dynamic shortest path problem, motivated by its applications in dynamic minimum cost flows in transformation problem. The study showed that this problem is equivalent to a classical shortest path problem in a so-called time-expanded network. Although our approach allows us to apply any standard technique on the time-expanded network, the size of this network is typically very large for realistic problems and it may be beneficial to avoid such explicit expansion. The study applied the Label Correcting Algorithm for solving this problem that the time complexity of the algorithm is $O(|nT||mT|)$.
Hung (2003) analyze the inverse shortest path length problem (ISPL) in transportation network improvement and bandwidth pricing.

Borgwardt et al (2005)researched on the shortest-path kernel as applicable to all graphs on which Floyd-Warshall can be performed. The requirement of Floyd-Warshall is that cycles that have negative weight do not exist. This condition generally holds if distances are represented by edge labels.

Li et al (2008) proposed an efficient algorithm named Li-Qi (LQ) for the SSSP problem with the objective of finding a simple path of the smallest total weights from a specific initial or source vertex to every other vertex within the graph. This algorithm is formed from the ideas of the queue and relaxation; the vertices may be queuedseveraltimes, and furthermore, only the source vertex and relaxed vertices are being queued.

III. Methodology

The Shortest Route Problem

This particular problem determines the route of minimum weight that connects two vertices namely a source and a destination in a weighted graph in a transportation network. Other situation can be represented by the same model like the Very Large Scale Integrated (VLSI) design, equipment replacement, and others. Different types of shortest path algorithm are used to determine the shortest path of a graph. The most frequently encountered path are the shortest path between two specified vertices, the shortest path between all pairs of vertices, and the shortest path from a specified vertex to all others.

The Dijkstra's algorithm is the most efficient algorithm used to find the shortest path between a known vertex to other vertices. Some improvements on Dijkstra's algorithm are done in terms of efficient implementation and cost matrix. In this project, we propose to implement the Dijkstra's algorithm to determine the shortest route from the production plant of the company to any of the other location in the network.

Dijkstra's algorithm

The problem of finding the shortest path from a specified vertex *s* to mother *t* can be stated as follows:
A simple weighted digraph G of n vertices is described by an n by n matrix D = $[d_{ij}]$, where d_{ij} = length (or distance or weight) of the directed edge from vertex *i* to vertex *j*:
Dijkstra's algorithm labels the vertices of the given digraph, at each stage in the algorithm some vertices have permanent labels and others temporary labels. The algorithm begins by assigning a permanent label 0 to the starting vertex s, and temporary label infinity to the remaining *n-1* vertices. Then, another vertex sets a permanent labelin each iteration, according to the following rules:

a. Every vertex *j* that is not yet permanently labelled gets a new temporary label whose value is given by min [old label of *j*, (old label of $i + d_{ij}$)], where *i* is the latest vertex permanently labelled, in the previous iteration, and $d_{ij}$is the direct distance between vertices *i* and *j*. if *i* and *j* are not joined by an edge, the d_{ij} = infinity.
b. The smallest value of all the temporary labels is found, and this becomes the permanent label of the corresponding vertex. In a case of more than one shortest path, select any one of the candidates for permanent labelling. Steps **a** and **b** are repeated alternately until the destination vertex t gets a permanent label. The first vertex to be permanently labelled is at a distance of zero from *s*. The second vertex to get a permanent label (out of the remaining *n-1* vertices) is the vertex closest to *s* from the remaining *n-2* vertices, the next one to be permanently labelled is the second closest vertex to *s*. And so on. The permanent label of each vertex is the shortest distance of that vertex from *s*.

   Simply, the Dijkstra's Algorithm can be stated as: Let u_i be the shortest distance from source node 1 to node *i*, and define d_{ij} (> 0) as the length of the arc (*i, j*). Then the algorithm defines the label for an immediately succeeding node *j* as

   $[u_j , i] = [u_i + d_{ij}, i], d_{ij} > 0$

That is the label for the node is [0, --], indicating that the node has no predecessor.
Node labels in Dijkstra's algorithm are of two types: *temporary* and *permanent.* A temporary label is modified if a shorter route to a node can be found. If no better route can be found, the status of the temporary label is changed to permanent.

Step 0: Label the source node (node 1) with the permanent label [0, --], set i = 1.

Step i:

a) Compute the temporary labels $[u_i + d_{ij}, i]$ for each node j that can be reached from node *i*, provided *j* is not permanently labelled. If node *j* is already labelled with $[u_j + k]$ through another node *k* and if $u_i + d_{ij} < u_j$, replace $[u_j, k]$ with $[u_i + d_{ij}, i]$.
b) If all the nodes have permanent labels, stop. Otherwise, select the label $[u_r, s]$ having the shortest distance $(=u_r)$ among all the temporary labels (break tie arbitrarily). Set $i = r$ and repeat step i.

IV. Analysis Of Data

This section introduces the data and analytical approach of the methodology used in the project research. The locations used for this project work was obtained from Dominion Paints Nig. Ltd. and also the distance in kilometre (km) between each location in the graph was measured using Google maps location.

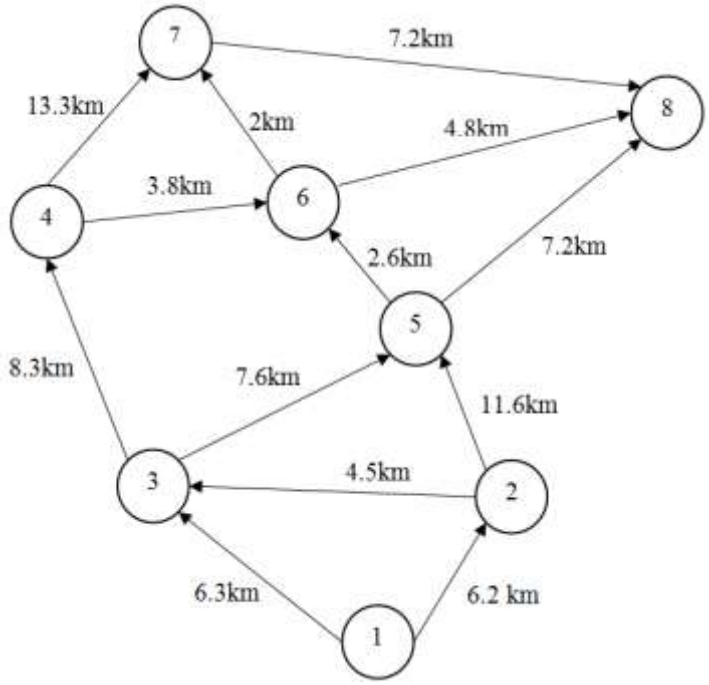


Fig.1. Permissible Route of the Road Network from Aluu through Mile 3.

LOCATION	NODES
Aluu (Production Plant)	1
Choba market	2
Rumuosi market	3
Rumuokoro	4
Location	5
Rumuokuta	6
Rumuola	7
Mile 3	8

Table 4.0. Names of location and Nodes identification.

4.1 Analytical Solution to the Problem.

Iteration 0: Assign the permanent label [0, ∞].

NODES	LABEL	STATUS
1	[0, ∞]	Permanent

Table .1

Iteration 1: Node 2 and node 3 can be reached from (the last permanent labelled) node 1, thus the list of labelled nodes becomes (temporary and permanent).

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[0+6.2, 1]	Temporary
3	[0+6.3, 1]	Temporary

Table 2.

For the two temporary labels [6.2, 1] and [6.3, 1], node 2 yields the smaller distance (u_2 = 6.2).thus the status of node 2 is changed to permanent.

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Temporary

Table .3a

Iteration 2: the new starting node is node 2. Node 3 and node 5 can be reached from node 2. Thus the list of labelled nodes becomes:

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Temporary
3	[6.2+4.5, 2]	Temporary
5	[6.2+11.6, 2]	Temporary

Table .3b

Node 3 temporary label [6.3, 1] obtained in iteration 1 remains the same because in iteration 2 node 3 holds another label [10.7, 2] and the shorter distance found is that of label [6.3, 1].

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Permanent
5	[17.8, 2]	Temporary

Table 4

Node 3 yields the shorter distance (u_3 = 6.3), thus the status of node 3 changed to permanent.

Iteration 3: Node 3 becomes the new start point. Node 3 connects to node 4 and node 5. Thus the list of labelled nodes is updated as:

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Permanent
5	[17.8, 2]	Temporary
4	[6.3+8.3, 3]	Temporary
5	[6.3+7.6, 3]	Temporary

Table5

Node 5 temporary label [17.8, 2] obtained from iteration 2, is thus changed to [13.9, 3] obtained in iteration 3 to indicate a shorter route through node 3. Thus the shorter distance is node 5 (u_5 = 13.9) labelled [13.9, 3], thus we change the status of node 5 permanent. The list of labelled node now becomes;

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Permanent
5	[13.9, 3]	Permanent
4	[14.6, 3]	Temporary

Table 6

Iteration 4: Node 6 and node 8 can be reached from node 5. The list of labelled nodes thus becomes:

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Permanent
5	[13.9, 3]	Permanent
4	[14.6,3]	Temporary
6	[13.9+2.6, 5]	Temporary
8	[13.9+7.2, 5]	Temporary

Table 7

Node 6 yields the shorter distance from node 5, thus the status of node 6 (u_6 = 16.5) becomes permanent. The list is now updated to become;

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Permanent
5	[13.9, 3]	Permanent
4	[14.6,3]	Temporary
6	[16.5, 5]	Permanent
8	[21.1,5]	Temporary

Table 8

Iteration 5: From node 6, node 7 and node 8 can be reached.

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Permanent
5	[13.9, 3]	Permanent
4	[14.6,3]	Temporary
6	[16.5, 5]	Permanent
8	[21.1,5]	Temporary
7	[16.5+2, 6]	Temporary
8	[16.5+4.8, 6]	Temporary

Table 9

Node 8 temporary label [21.3, 6] obtained from iteration 5, is thus changed to [21.1, 5] obtained in iteration 4 to indicate a shorter route through node 5. Thus the shorter distance is node 7 (u_7 = 18.5) labelled [18.5, 6], thus we change the status of node 7 permanent. The list of labelled nodes now becomes;

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Permanent
5	[13.9, 3]	Permanent
4	[14.6,3]	Temporary
6	[16.5, 5]	Permanent
8	[21.1,5]	Temporary
7	[18.5, 6]	Permanent

Table 10

Iteration 6: Node 8 can be reached only from node 7.

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Permanent
5	[13.9, 3]	Permanent
4	[14.6,3]	Temporary
6	[16.5, 5]	Permanent
8	[21.1,5]	Temporary
7	[18.5, 6]	Permanent
8	[18.5+7.2, 7]	Temporary

Table 11

Node 8 temporary label [25.7, 7] obtained from iteration 6, is changed to [21.1, 5] obtained in iteration 5 to indicate a shorter route has been found through node 5. Thus we change the status of node 8 (u_8 = 21.1) permanent. The list of labelled node now becomes;

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Permanent
5	[13.9, 3]	Permanent
4	[14.6,3]	Temporary
6	[16.5, 5]	Permanent
8	[21.1,5]	Permanent
7	[18.5, 6]	Permanent

Table 12

Iteration 7: Again node 7 can be reached from node 4. The list of labelled nodes;

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Permanent
5	[13.9, 3]	Permanent
4	[14.6,3]	Temporary
6	[16.5, 5]	Permanent
8	[21.1,5]	Permanent
7	[18.5, 6]	Temporary
7	[14.6+13.3, 4]	Temporary

Table 13

Node 7 temporary holds label [27.9, 4] obtained from iteration 7, is thus changed to [18.5, 6] obtained in iteration 6 to indicate a shorter route has be found through node 4. Therefore node 7 is updated as (u_7 = 18.5) labelled [18.5, 6], we cannot change the status of node 7 since it is already permanent. The list of labelled node now becomes;

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Permanent
5	[13.9, 3]	Permanent
4	[14.6,3]	Temporary
6	[16.5, 5]	Permanent
8	[21.1,5]	Permanent
7	[18.5, 6]	Permanent

Table 14

Iteration 8:

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Permanent
5	[13.9, 3]	Permanent
4	[14.6,3]	Temporary
6	[16.5, 5]	Permanent
8	[21.1,5]	Permanent
7	[18.5, 6]	Permanent

Table 15

Since node 4 can only be accessed by node 3 therefore we make node 4 (u_4 = 14.6) status permanent.

NODES	LABEL	STATUS
1	[0, ∞]	Permanent
2	[6.2, 1]	Permanent
3	[6.3, 1]	Permanent
5	[13.9, 3]	Permanent
4	[14.6,3]	Permanent
6	[16.5, 5]	Permanent
8	[21.1,5]	Permanent
7	[18.5, 6]	Permanent

Table 16

Now all nodes in the list have attained its status as permanent, we can therefore say we have obtained the maximum iteration for the problem.
Therefore the solution to the shortest route and distance from node 1 to any other node in the network is thus;

NODES	ROUTES	DISTANCE
1	1	0
2	1 – 2	6.2
3	1 – 3	6.3
4	1 – 3 – 4	14.6
5	1 – 3 – 5	13.9
6	1 – 3 – 5 – 6	16.5
7	1 – 3 – 5 – 6 – 7	18.5
8	1 – 3 – 5 – 8	21.1

Table 17

Comparing the results with TORA Operation Research software which gives the same routes and distance (see Appendix A) for the solution shows that the analytical processes and solution is correct.

V. Conclusion

The shortest route between node 1 (production plant) and any other node in the network can be determined by starting at the desired destination and back-tracking through the nodes using the information given by the permanent labels (Table.4.1.16), for example the sequence or route determines the shortest route from node 1 to node 7.
(7) – [18.5, 6] – (6) – [16.5, 5] – (5) – [13.9, 3] – (3) – [6.3, 1] – (1)
Thus the desired route is;
1 – 3 – 5 – 6 – 7

Summary

In this paper,we discussed the problem of finding the shortest route motivated by the need to minimize the distance and time of transporting goods from the company's production plant to seven different dealers in the road network given by the data. The application of the Shortest Path using Dijkstra's Algorithm tackled the problem effectively. The results from the analysis of the data show that the company can implement the use of Dijkstra's Algorithm to obtain the shortest route to transport their products from Aluu to another destination of choice.

VI. Recommendation

Based on the conclusion of the project research work, we recommend that the company implements the Dijkstra's Algorithm to find the shortest routes from their current production plant to any delivery store of choice now and in the nearest future to help them;

- Minimize the distance of transporting the goods.
- Save time in transporting the products profit of the company.
- Minimize the cost of running the transportation of goods to maximize profits of the company.

Reference

[1]. Ahmat, K. A. (n.d.). *Graph Theory and Optimization Problems for Very Large Networks.* New York: City University of New York/Information Technology.

[2]. Brendan, H. a. (2005). Generalizing Dijkstra's Algorithm and Gaussian Elimination for solving MDPs. *International Conference on Automated Planning and Scheduling/Artificial Intelligence Planning System*, (pp. 151 - 160).

[3]. Chen, K. M. (2009). A real-time wireless route guidance system for urban traffic management and its performance evaluation. *Papers of the 70th Vehicular Technology Conference Anchorage VTC*, (pp. 1 -5).

[4]. Ebrahimnejad, S. A. (2011). Find the Shortest Path in Dynamic Network using Labelling Algorithm. *Journal of Business and Social Science*.

[5]. Goldberg, A. V. (n.d.). *Point - to - Point Shortest Path Algorithms with Pre-processing.* Silicon Valley:MicrosoftResearch.

[6]. Goyal, S. (n.d.). *A Survey on Travelling Salesman Problem.* North Dakota: Department of Computer, University of North Dakota.

[7]. Lee, D. C. (2006). Proof of a modified Dijkstra's algorithm for computing shortest bundle delay in networks with deterministically time-varying links. *IEEE Communications Letters*, 734 -736.

[8]. Lieberman, F. S. (2001). *Introduction to Operations research seventh edition.* Mc Graw Hill.

[9]. Lili Cao, X. Z. (n.d.). *Approximating Shortest Path in Social Graph.* U.C. Santa Barbara: Computer Science Department.

[10]. Meghanathan, D. N. (n.d.). *Review of Graph Theory Algorithms* . MS: Department of Computer Science , Jackson State University.

[11]. Nar, D. (1997). *Graph theory with applications to engineering and computer science.* Prenctice Hall.

[12]. Sadavare, A. (2012). *International Journal of Computer Science and Information Technologies*, 5296 - 5300.

[13]. Scutella, S. P. (1997). *Technical report on Shortest Path Algorithms in Transportation models, Classical and innovative aspects.*

[14]. Sommer, C. (2010). *Approximate Shortest Path and Distance Queries in Networks.* Tokyo: Department of Computer Science, Graduate School of Information Science and Technology.

[15]. T. Li, Q. a. (2008). An efficient Algorithm for the single source Shortest Path Problem in Graph Theory. *International Conference on Intelligent System and Knowledge Engineering*, (pp. 152 - 157).

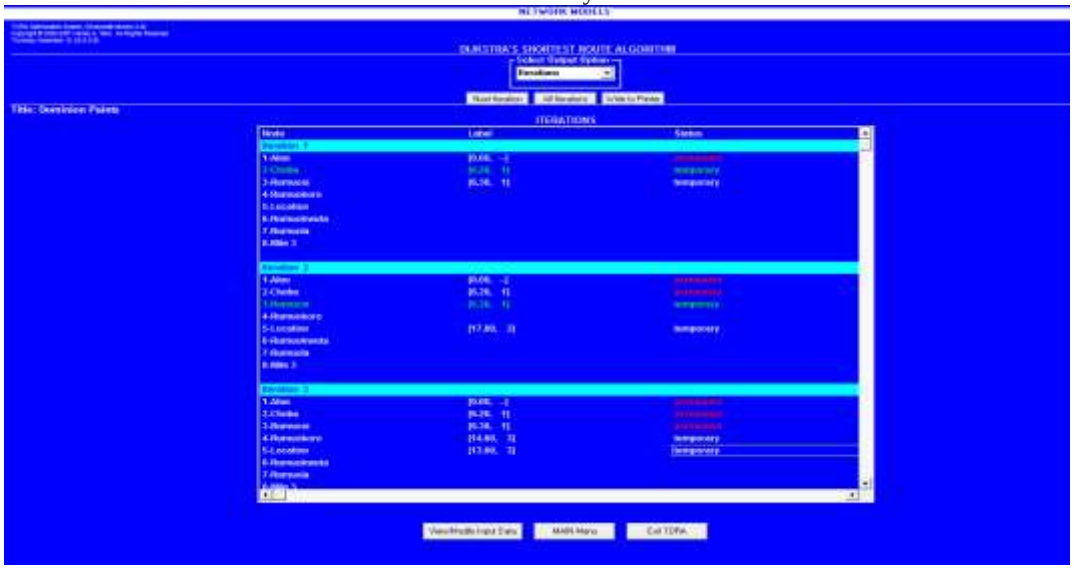
[16]. Taha, H. (2011). *Operations research an introduction, ninth edition*. Pearson Publisher. Wadhwa, S. (n.d.). *Analysis of a network design problem.* 2000: Lehigh University

APPENDIX A

TORA Operation Research Software.

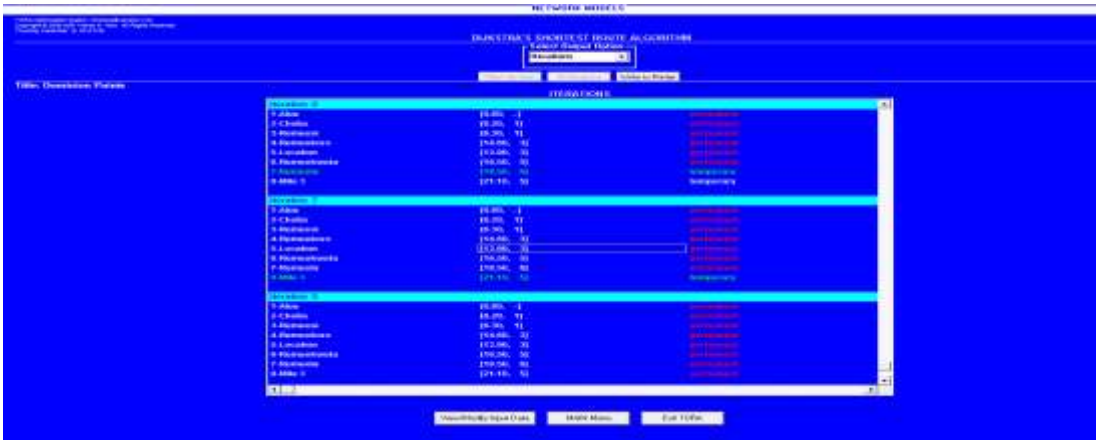
Data Input for Dijkstra's Algorithm.

The Shortest route from node 1 to any other node in the network.



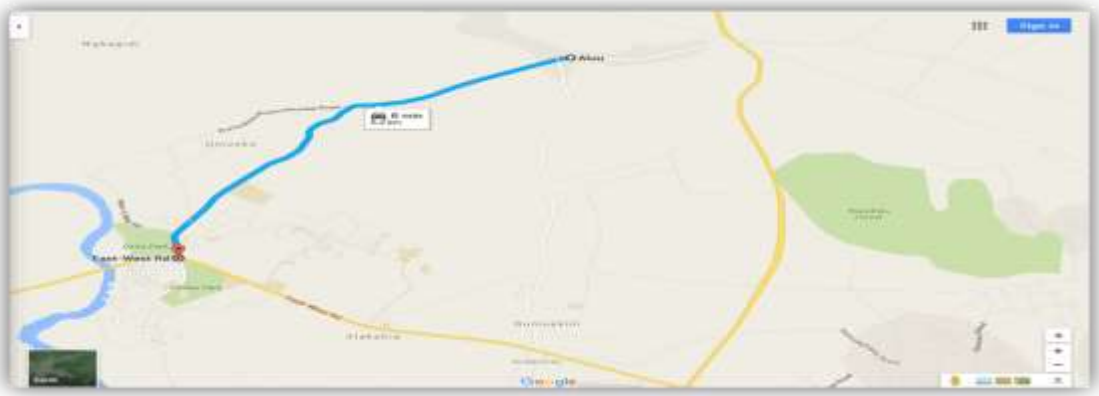
Iteration 1 – 3.

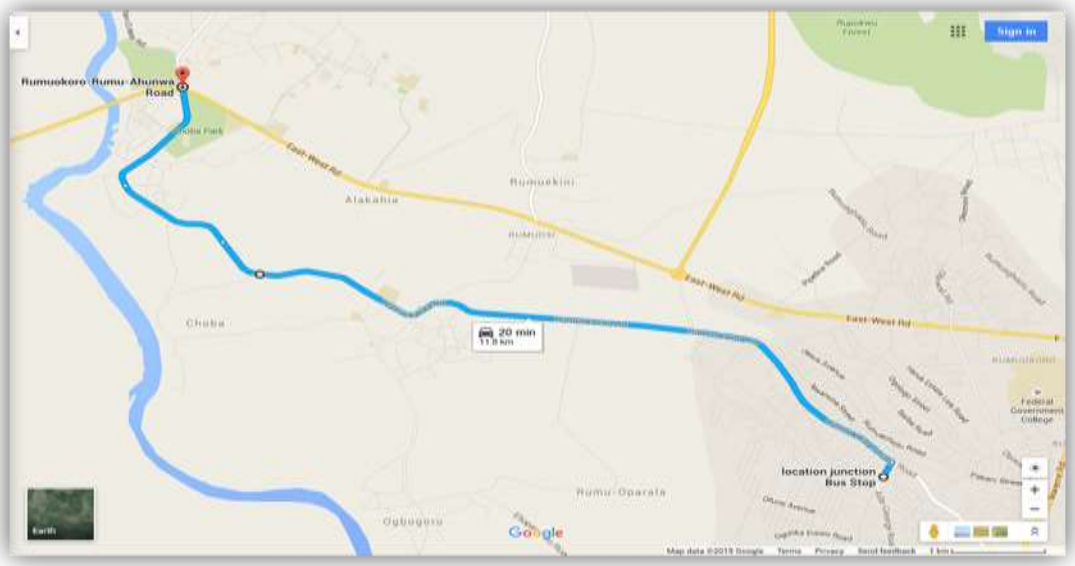
Iteration 4 – 6.

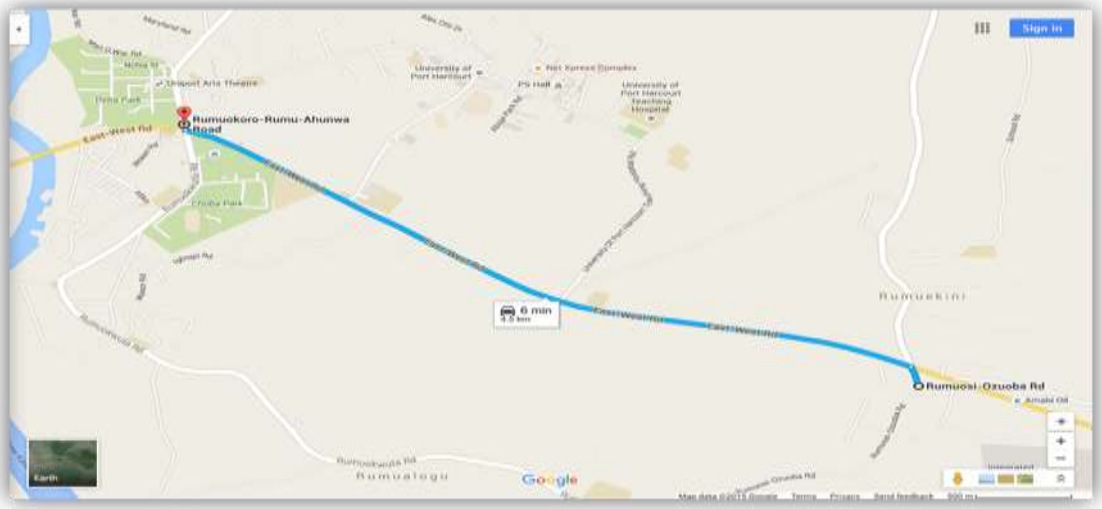


APPENDIX B

Map showing the different route and their distances (km) in the area of our research.



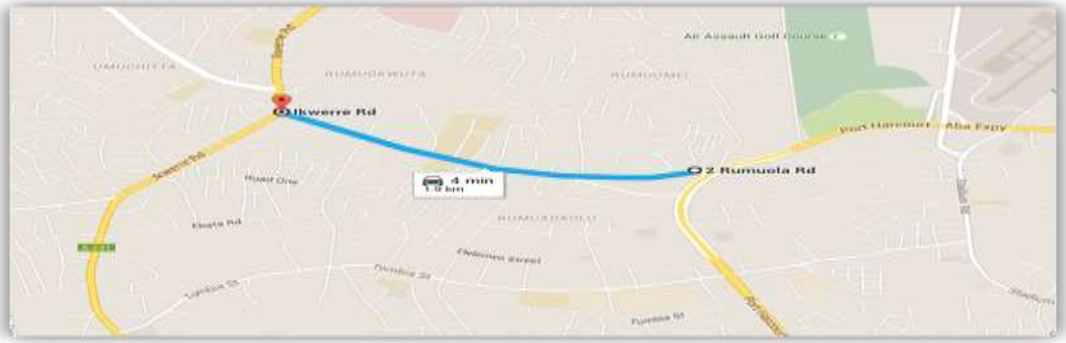

Rumuokoro-Rumu-Ahunwa Road
East-West Rd
Alakahia
Rumuekini
Chuba
20 min
location junction Bus Stop
Rumu-Oparata
Ogbogoro
Google

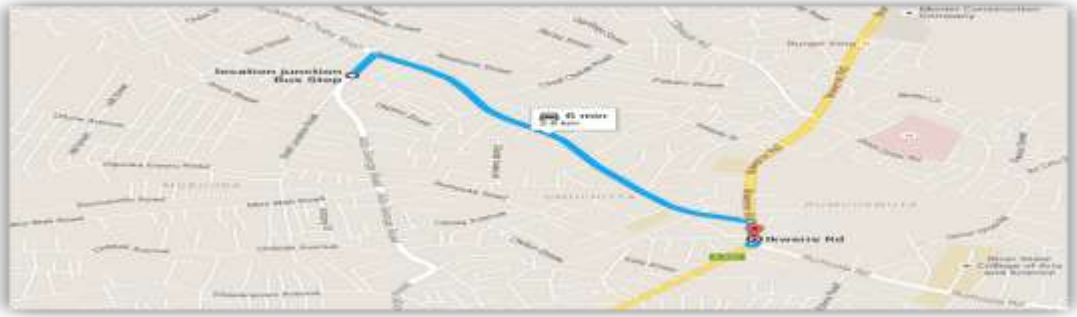

Rumuokoro-Rumu-Ahunwa Road
6 min
Rumuosi-Ozuoba Rd
Rumuekini
Rumualogu
Google


Rumuosi
Rumuokwuta-Choba Road
Rumuologbo
Ogbogoro


Rumuokoro
10 min
Ikwerre Rd


Mile Three Market


Ikwerre Rd
4 min
2 Rumuola Rd


Ikwerre Rd