wjert, 2019, Vol. 5, Issue 3, 500-509. Review Article ISSN 2454-695X

World Journal of Engineering Research and Technology
WJERT
www.wjert.org SJIF Impact Factor: 5.218

DESIGN AND IMPLEMENTATION OF A NETWORK ROUTING FOR PATH SELECTION USING DIJKSTRA ALGORITHM

***[1]Dr. ILO Somotoochukwu, [2]Amadi Christopher, [3]Umesi Cosmos N. and [4]Osondu U. S.**

[1,2,3]Computer Engineering Department, Michael Okpara University of Agriculture, Umudike, Abia State.

[4]Electrical/Electronic Engineering Department, Federal Polytechnics Nekede, P.M.B 1036, Owerri, Imo State.

Article Received on 09/04/2019 Article Revised on 30/04/2019 Article Accepted on 20/05/2019

***Corresponding Author**
Dr. ILO Somotoochukwu
Computer Engineering Department, Michael Okpara University of Agriculture, Umudike, Abia State.

ABSTRACT

The complete network of large number of interconnected autonomous system, each of which consists of a distinct domain, need to communicate to different nodes to forward the information. There are various means of communications between these systems. Routing is one of them. Routing refers to the overall network wide process that determines end to end paths that datagram will take from source to destinations. There are different routing algorithms available such as BELLMAN FORD ALGORITHM which is decentralized routing algorithm and DIJKSTRA ALGORITHM which is a global routing algorithm. In global routing algorithm, each router has a complete view of the network, whereas in a decentralized routing algorithm each router has a local view consisting of its directly attached neighbors. Out of this Dijkstra algorithm is mostly preferred as it is faster as compared to any other algorithm and its implementation is easier. In this algorithm, router based information that has been collected from other routers, builds a graph of the network which shows the locations & links of routers within themselves and network. These links are labeled with a number called weight or cost. If there are two links between a node and destination, the router chooses the link with the lowest weight. One of the advantages of Dijkstra algorithm is that router computes routes independently using the same original status data; they do not depend on the computation of intermediate machines.

Because link status messages propagated unchanged, it is easy to debug problems. Because routers perform the route computation locally, it is guaranteed to converge. Finally, because link status messages only carry information about the direct connections from a single router, the size does not depend on the number of networks in the networks in the internet. Thus, Dijkstra algorithms scale better than distance vector algorithms. We have implemented Dijkstra algorithm for routing purpose.

KEYWORDS: Dijkstra Algorithm, Network Routing, Data Transfer, Simulation.

I. INTRODUCTION

The internet is one of the 20th century's greatest communications developments. It allows people around the world to send E-Mail to one another in a matter of seconds. But none of these parts would ever make it to our computer without a piece of the Internet that we have probably never seen. In fact, most people have never stood face to face with this piece of machine responsible for the technology that allows the internet to exit at all. The equipment is the ROUTER. In order to transfer data from a source host to the destination host, the network must determine the path or the route links between source and destination. This can be achieved by using routers. The routers in a network are responsible for receiving and forwarding packets through the interconnected set of networks. In particular, the router must avoid the portion of the network that has failed or has too much congestion. At the heart of the router, there is an algorithm which takes the dynamic decisions and directs the datagram from source the destination. This is better known as Routing Algorithm. Routing Algorithms are classified depending upon the technique they employ to calculate the shortest path. Dijkstra Routing Algorithm is one of them.

II. LITERATURE SURVEY

This section reviews various works by different writers and researchers on dijkstra algorithm in improving network routing. There are many different proposed method available.

Review of Relevant Literature

A* Search Algorithm

In computer science, A* (pronounced "A Star") is a best-first, graph search algorithm that finds the least-cost path from a given initial node to one goal node (out of one or more possible goals). It uses a distance-plus-cost heuristic function (usually denoted f(x) to determine the order in which the search visits nodes in the tree). The distance-plus-cost heuristic is a sum of two functions:

- The path-cost function (usually denoted g(x), which may or may not be a heuristic) and an admissible "heuristic estimate" of the distance to the goal (usually denoted h(x)).
- The path-cost function g(x) is the cost from the starting node to the current node.

Since the h(x) part of the f(x) function must be an admissible heuristic, it must not overestimate the distance to the goal. Thus for an application like routing, h(x) might represent the straight-line distance to the goal, since that is physically the smallest possible distance between any two points (or nodes for that matter).

The algorithm was first described in 1968 by Peter Hart, Nils Nilsson and Bertram Raphael. In their paper, it was called algorithm A. Since using this algorithm yields optimal behavior for a given heuristic, it has been called A*.

The bellman-ford algorithm

The Bellman-ford algorithm, sometimes referred to as the Label Correcting Algorithm, computes single-source shortest paths in a weighted digraph (where some of the edge weights may be negative). Bellman-ford is usually used only when there are negative edge weights. Bellman-ford is in its basic structure very similar to Dijkstra's algorithm, but instead of greedily selecting the minimum-weight node not yet processed to relax, it simply relaxes all the edges, and does this !V! – 1 times, where !V! is the number of vertices in the graph. The repetitions allow minimum distances to accurately propagate throughout the graph, since, in the absence of negative cycles; the shortest path can only visit each node at most once.

Floyd-Warshall algorithm

In computer science, the Floyd–Warshall algorithm (sometimes known as the Roy–Floyd algorithm or WFI Algorithm, since Bernard Roy described this algorithm in 1959) is a graph analysis algorithm for finding shortest paths in a weighted, directed graph. A single execution of the algorithm will find the shortest path between all pairs of vertices. The Floyd–Warshall algorithm is an example of dynamic programming.

Theoretical Framework

Routing Technique

There are three various routing techniques most widely used. The techniques differ in the manner in which routes are defined and selected and how route information is shared between the networking nodes. The three routing techniques are source routing, label swapping, and destination routing.

Source routing: This is the familiar source routing technique proposed by IBM to IEEE 802.1 committee in March 1990. Source routing pertains to the concept that each information frame contains the complete route needed for the soured station to transport data to the destination stations. The source station in some manner, either through broadcast searches or static definitions, provides the road map (directions) to the destination station. Replies from a destination station would use the same directions but only in reverse. While this technique may be the most efficient for routing information frames it comes with an overhead cost because each frame contains the complete description of the route. The size of the descriptor field varies depending on the protocol being used.

Label-based routing: This routing technique is often referred to as address swapping and is the dominant technique used in switching network models. Switching networks exercise label swapping at layer 2 and router networks use label swapping at layer 3. The label or address is in actuality an index into a switching table at each intermediate routing node in the network. The intermediate node uses the label as an index into the switching table to determine which port or link the information frame should be sent out on and provides a new label for the next node on the route to use as an index into the table.

Destination *routing:* This routing technique is based on layer 2 information. The information provides the destination and source address but does not include a specific route. The concept of this technique in routing can be summed up in a question: “How can I get there from here and which is the best way?” Each node and intermediate node, after interpreting the layer 3 information, will assess he current topology and congestion of the network and make its decision based on predefined information on routing to the destination. The three most widely used protocols; IP, IPX, and SNA utilize destination routing. Each of these routing techniques requires a routing algorithm to determine the optimal route.

III. MATERIAL AND METHODOLOGY

Material Used

In implementing this system, we need to use the following materials:

- Personal Computer.
- Packet Tracer.
- Java Programming.

Personal Computer

This material served as the development and testing machine during the project implementation. Personal computer is a multi-purpose electronic computer whose size, capabilities and price make it feasible for individual use.

Packet Tracer

This material served as the simulation tool designed by Cisco systems that allow user to create network topologies and imitate modern computer network. The software allow users to simulate the configuration of Cisco routers and switches using a simulated command line interface.

Java Programming

This is a general purpose computer programming language that is concurrent, class based, object oriented and specifically designed to have as few implementation dependencies as possible. This material served as an interface between the packet tracer and the personal computer.

Methods

This is a manner or mode of procedure, especially an orderly, inquiry, experiment, presentation, logical and systematic way of doing something accordance with a definite plan. The are two methods used for the implementation of the system which are:

- **Connection Oriented Network Routing**
- **Connectionless Networks Routing**

Connection Oriented Network Routing

In connection-oriented network, the path used for the transmission of information between the source and destination is predetermined prior to the sending of the data. Once a path is selected the information utilizes the established path for the duration of the transmission. The most well-known and used connection-oriented network is the telephone system.

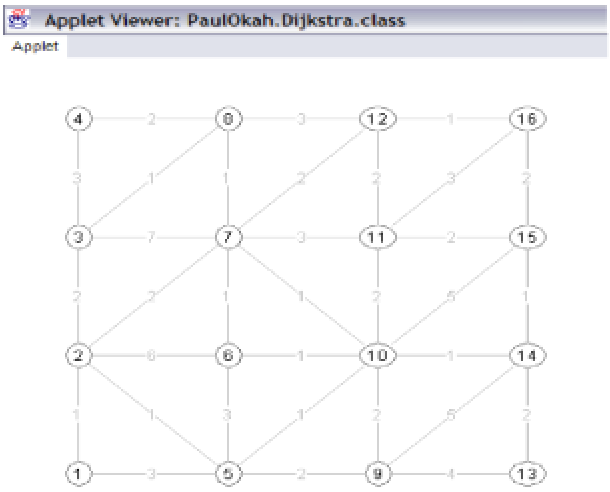
Connectionless Networks Routing

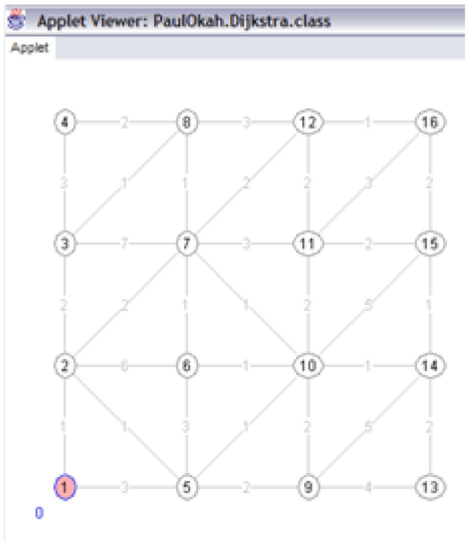
The main difference between connection-oriented and connectionless networks is in the responsibility for error correction and message unit sequencing. Connection-oriented networks place error detection, recovery, and sequencing clearly in the network. The end nodes (i.e., users, applications) are not to be concerned with these networking functions when using connection-oriented conversations. Connectionless networks, however, view these functions in the end nodes themselves. They are responsible for delivering the data in any way possible. This is known as a best-effort delivery system.

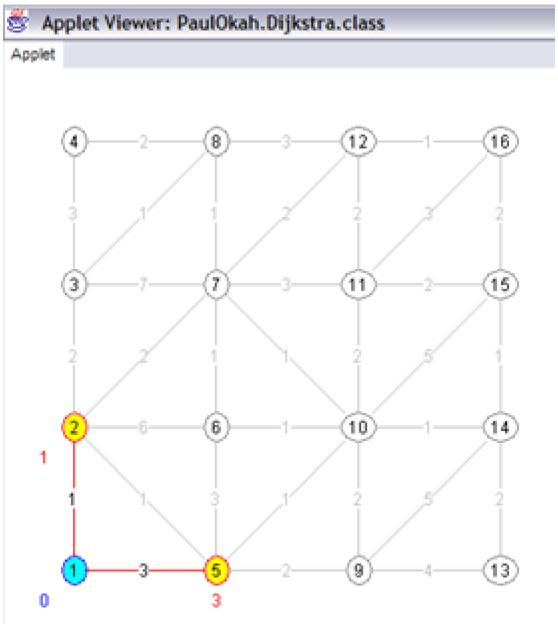
IV. Software Documentation and Implementation

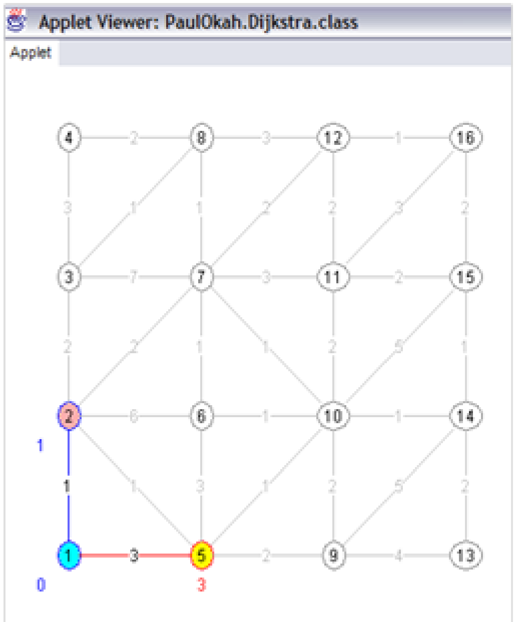
Running the code for the Dijkstra routing algorithm, this is what you see. It is the simulated network when the program is running. The circles with numbers in them are the nodes or stations or routers. The lines joining the nodes are the links/ linkages which are considered as distances between nodes (cables). The numbers on the linkages are the weights of linkages. The total cost of going from a starting node to a target node is written just around the targeted node. But before the movement starts, it must first listen for the mouse click. At the click of the mouse it goes from the start node assumed to be node 0 to node 1. It first paints the targeted node red and waits to hear the next click before changing the color to blue. The blue color indicates that the node has been visited. So it's permanent. We also have the pleasure of changing the circle nodes to rectangles. It is also pertinent to understand that Dijkstra routing algorithm is a global routing algorithm. This means that each node has the complete router intelligence of the entire network. The algorithm surveys the entire network before it can take decision on which route to take. All the network attributes are considered before routes are chosen.

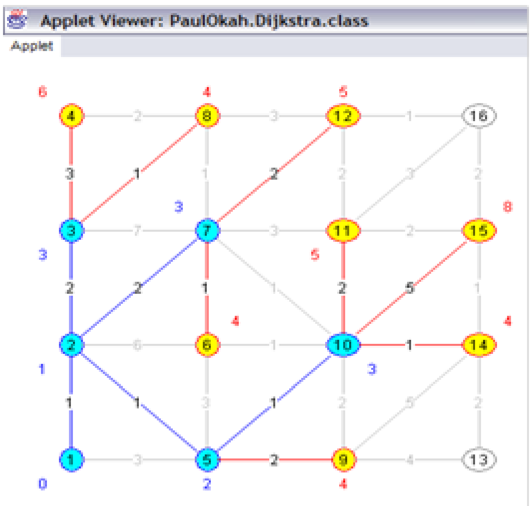
Also notice that a blue line is drawn to trace the shortest path. Below is written the entire analysis of the result.

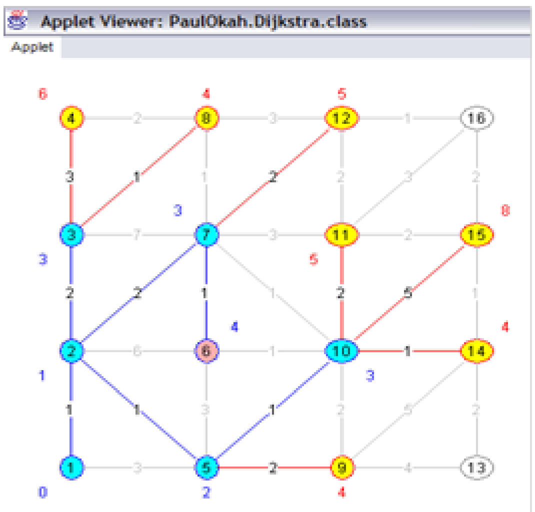

Applet Viewer: PaulOkah.Dijkstra.class
Applet

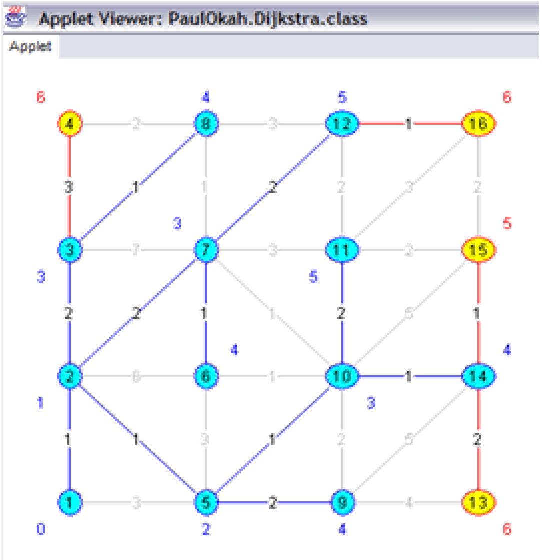

Applet Viewer: PaulOkah.Dijkstra.class
Applet

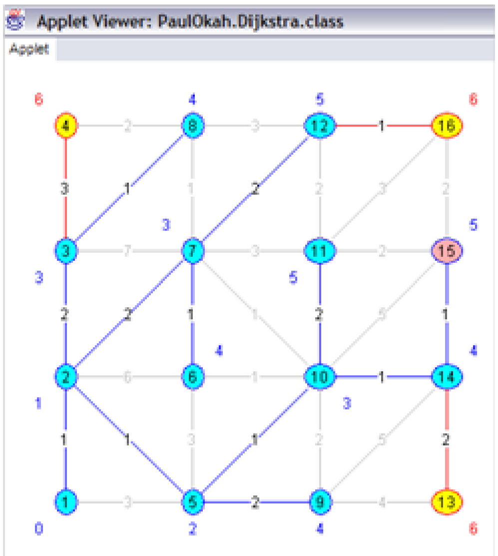

Applet Viewer: PaulOkah.Dijkstra.class
Applet

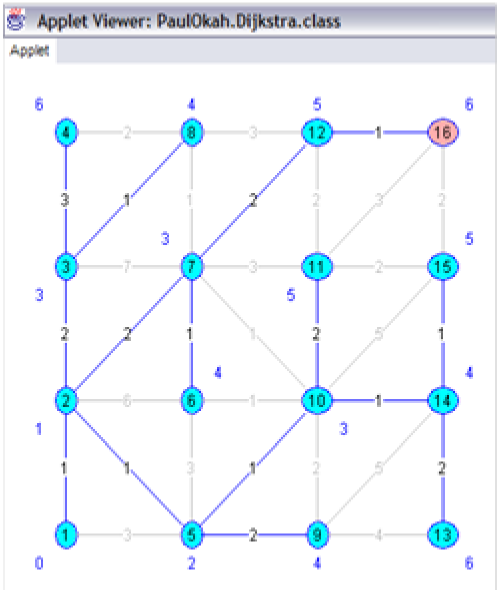

Applet Viewer: PaulOkah.Dijkstra.class
Applet

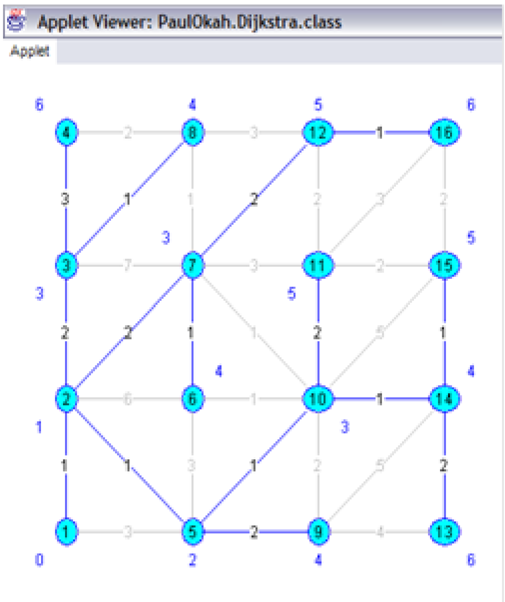

Applet Viewer: PaulOkah.Dijkstra.class
Applet

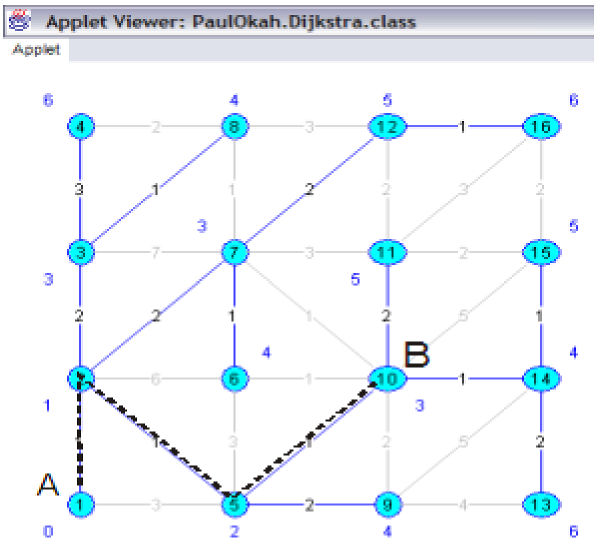

Applet Viewer: PaulOkah.Dijkstra.class
Applet


Applet Viewer: PaulOkah.Dijkstra.class
Applet


Applet Viewer: PaulOkah.Dijkstra.class
Applet


Applet Viewer: PaulOkah.Dijkstra.class
Applet


Applet Viewer: PaulOkah.Dijkstra.class
Applet


Applet Viewer: PaulOkah.Dijkstra.class
Applet
B
A

CONCLUSION

In today's world, networking plays an important role in communication between autonomous computers. For this many hardware devices and software algorithms have been designed. So far, the traditional system used for the communication were the hub networking system and many other hardware applications present in the market, but as they operate on electricity, it may lead to the failure of device due to some malfunctioning in the hardware circuitry. Dijkstra Algorithm has been incorporated in 'java', which provides easy understandability and hence its chances of failure is negligible. Generally, under normal circumstances after we have determined the optimum path for a datagram in a network using Dijkstra Algorithm, if a link failure occurs then the complete network topology changes including the path that the datagram was following. Hence in case of a link failure we have to run the algorithm all over again to provide the shortest path to the datagram. However, we have implemented the Dijkstra Algorithm in such a way that under the context of link failure it will provide the next shortest path from the available alternate paths that has been calculated along with the Optimum Path itself. As a result, there is no need to execute the algorithm again.

REFERENCES

1. B. Baran and R. Sosa. A new approach for Ant Net routing. In Proceedings of ICCCN, Las Vegas, NV, USA, IEEE Press, 2000; 303–308.
2. D. Bertsekas, R. Gallager, Data Networks, Prentice hall, New Jersey, 1987.
3. D. Bertsekas. Dynamic Programming and Optimal Control. Athena Scientific, USA, 1995.
4. D. Bertsekas and R. Gallager. Data Networks. Prentice–Hall, Englewood Cliffs, NJ, USA, 1992.
5. E. Bonabeau, M. Dorigo, and G. Theraulaz. Swarm intelligence: from natural to artificial systems. Oxford University Press, Inc., New York, NY, USA, 1999.
6. E. Bonabeau, F. Henaux, S. Gu´erin, D. Snyers, P. Kuntz, and G. Theraulaz. Routing in telecommunications networks with ant-like agents. In Proceedings of IATA, pages 60–71, London, UK, Springer-Verlag, 1998.
7. J. S. Baras and H. Mehta. A probabilistic emergent routing algorithm (PERA) for mobile ad hoc networks. In Proceedings of WiOpt, 2003.
8. M. steenstrup, Routing in Communication networks, Prentice hall, New Jersey, 1995.
9. O. Babaoglu, G. Canright, A. Deutsch, G. A. Di Caro, F. Ducatelle, L. M. Gambardella, N. Ganguly, M. Jelasity, R. Montemanni, A. Montresor, and

T. Urnes. Design patterns from biology for distributed computing. ACM Transactions on Autonomous and Adaptive Systems, 2006; 1(1): 26–66.

10. S. Appleby and S. Steward. Mobile software agents for control in telecommunications networks. BT Technology Journal, 2000; 18(1): 68–70.