International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 02 Issue: 03 | June-2015 www.irjet.net p-ISSN: 2395-0072

Subnet Shortest Path Pseudocode based on Dijkstra's Algorithm

Salah Elrofai[1], Abdeen Abdelkareem[2]

[1]*Assistant Professor, School of Electronic, Collage of Engineering, Sudan University of Science and Technology, Sudan.*
[2]*Full Professor, School of Electronic, Collage of Engineering, Sudan University of Science and Technology, Sudan*

Abstract - *Routing algorithms classified as adaptive and Non-adaptive. Non-adaptive algorithm is called static routing. The most static routing widely used is a shortest path routing. This type of algorithms builds a graph of subnet, with nodes for routes and arcs for links. Each arcs labeled with the result of a certain weighting function for computing the shortest path.* ***Dijkstra's algorithm is a single*** *source shortest path algorithm that can find the shortest paths from a given source node to another given one. Accordingly design a subnet used the C-program for* ***Dijkstra's algorithm*** *for computing the shortest path through subnet nodes, this program is translated into a pseudocode in order to be used by any other languages.*

Key Words: Routing algorithms, Non-adaptive algorithm, shortest path, Dijkstra's algorithm, pseudo code.

1. Introduction

Routing algorithms are numerous and can be differentiated on basis of several key characteristics. First the particular goals of the algorithm designer affect the operation of the resulting routing protocol. Secondly various types of routing algorithms exist and each algorithm has a different impact on network and router resources. Routing algorithms compose out of adaptive and Non-adaptive. In non-adaptive algorithm in which the route between routers is computed in advance, this procedure is called static routing. The most static routing widely used is a shortest path routing because it is simple and easy. This type of algorithms builds a graph of subnet, with nodes for routes and arcs for links. Each arcs labeled with the result of computed distance, bandwidth, average traffic, communication traffic, mean queue length and measured delay. Due to the result of a certain weighting function this algorithm computes the shortest path. Dijkstra's algorithm is a single source shortest path algorithm that can find the shortest paths from a given source node x to a given node y [1]. By building a set called (IN) that initially contains only x but grows as algorithm proceeds. At any given time IN contains every nodes whose shortest path from x, using only nodes in IN, has so far been determined. For every node z outside IN keeping track of the shortest distance d[z] from x to that node, using a path that has only non – IN node is z. Track for the nodes adjacent to z on this path, s[z], so it is prefer to represent the adjacency-matrix representation graph [2]. Picking the non-IN node with the smallest distance d*1. Once added that node, call it p, to IN, that d must be recomputed for all the remaining non - IN nodes because there may be a short path *2 then also s[z] must updated so that p is now shown to be the node adjacent to z on the current shortest path. As soon as the destination is moved into selected IN, this IN stops growing. The current value of d[y] is the distance for the shortest path, and its nodes are found by looking at y, s[y],s[s[y], and so on until the path is traced back to x . the input for such algorithm, call it algorithm shortest path is adjacency matrix for a simple connected graph G with positive weights and nodes x and y the algorithm writes out the shortest path between x and y and the distance for that path. The performance of Dijkstra's algorithm depends of how being choose to implement the priority IN [3].

2. Related work

The author uses the most popular Dijkstra's algorithm in computer science, for providing a dynamic programing syllabus and in turn dynamic programming should be (at least) alluded to in a proper exposition/teaching of the algorithm [4].
The author selecting dijkstra's algorithm for finding the path problem and is implemented in quantum search. The implementation results represented the dijkstra's algorithm as a probabilistic quantum – classical algorithm [5].

3. Shortest Path Pseudocode

Shortest path (n*n matrix A:x,y)
// Dijkstra's algorithm. A is a modified adjacency matrix for a simple, connected graph with positive weights: x and y are nodes in the graph: writes out nodes in the shortest path form x to y. and the distance for that path.
Local variables:


```
Set of nodes IN                 //set of nodes whose shortest path form x is known
Nodes p                         // temporary nodes
Array of integers d             //for each node, the pervious form x using nodes in IN
Array of node s                 // for each node the pervious node in the shortest path
Integer Old Distance            // distance to compare against
initialize set IN and           arrays d and s.
IN=(x)
d[x]=0
for all nodes z not in IN do
d[e] = A[x,y]
s[z] = x
end for
//process nodes into IN
While y not in IN do
//add minimum distance node not in IN
P = node z not in with minimum d[z]
IN = IN υ (P)
// recomputed d for non IN nodes, adjusts if necessary N-link for z not in
 IN do
Old Distance = d[z]
d[z] = min (d[z], d[p] +A[p,z])
If d[z] = Old Distance then
S[z] = p
end if
   end for
end while
// write out path nodes
Write ("IN reverse order, the path")
Write (y)
Z = y
Repeat
Write (s[z])
Z = s[z]
Until z = x
// write out path distance
Write ("the path distance is", d[z])
End shortest path (7)
```

4. Subnet design

Consider the graph in figure 1 and corresponding modified adjacency matrix shown in table 1.

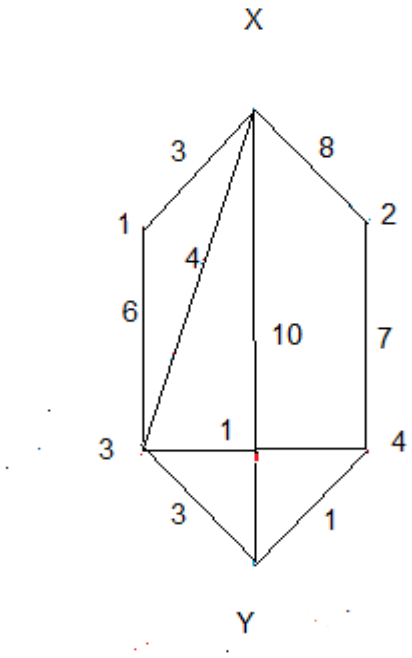


Figure 1 subnet graph

Table 1 Adjacency matrix

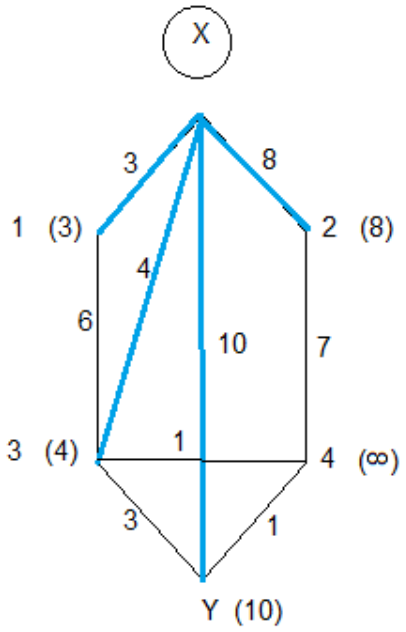
	X	1	2	3	4	Y
X	∞	3	8	4	∞	10
1	3	∞	∞	6	∞	∞
2	8	∞	∞	∞	7	∞
3	4	6	∞	∞	1	3
4	∞	∞	7	1	∞	1
Y	10	∞	∞	3	1	

Trace an algorithm of the shortest path on this graph. At the end of this process an initialization phase IN contains x and d, contains all the direct distances from x to other nodes:
IN = [x]

Table 2 shows the direct distance from x to other nodes.

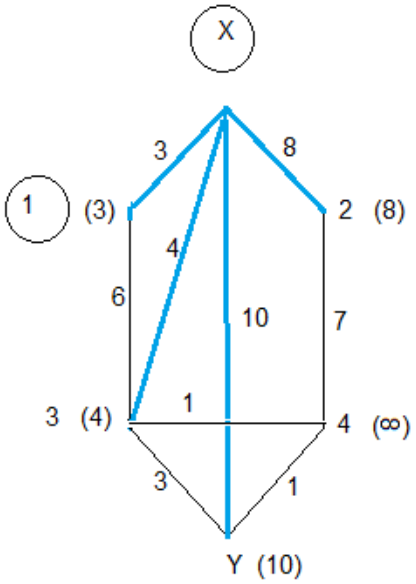
	X	1	2	3	4	Y
d	0	3	8	4	∞	10
s	-	x	x	x	x	x

In the figure (2-a), circled nodes are those in set IN, blue lines show the current shortest paths and the d- value for each node is written along with the node label. Figure is the picture after initialization shows shortest path.



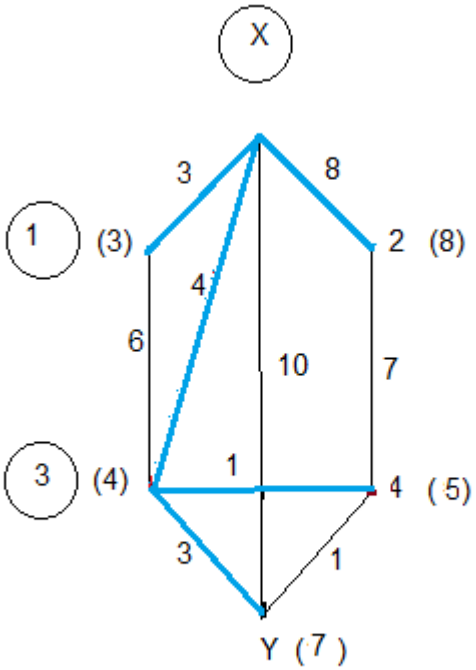
(a)

Now entering while loop and search through the d-values for the node of Minimum distance that is not in IN this turns out to be node 1, d [1] = 3. In the for loop recomputed all the d-values for remaining nodes, 2, 3, 4 and y.
P = 1
IN = [x, 1]
D [2] = min (8, 3 +A [1, 2]) = min (8, ∞) = 8
D [3] = min (4, 3 +A [1, 3]) = min (4, 9) = 8
D [4] = min (∞, 3 +A [1, 4]) = min (∞, ∞) = ∞
D[y] = min (10, 3 +A [1, y]) = min (10, ∞) = 10
There were no change in d-values, so there were no change in these values were no shorter path form x by going through node 1 than going directly shown in figure (3-b).



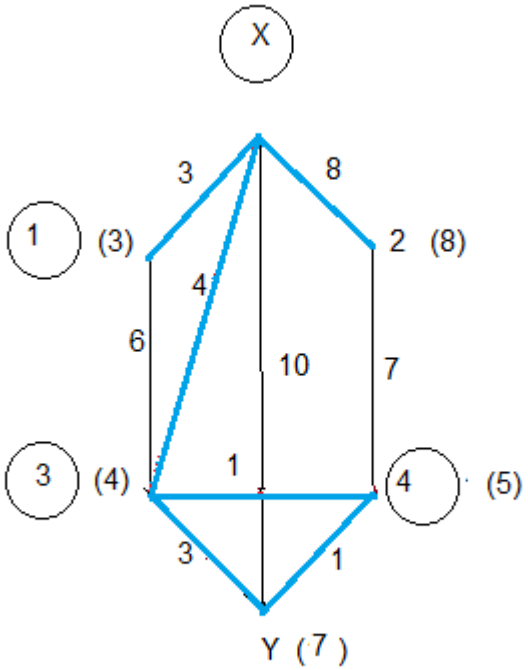
(b)

The second path through the while loop procedures the following:
P = 3 (3 has the smallest d- value namely 4, of 2, 3, 4, or y)
IN = [x, 1, 3] shown in figure (4-C).



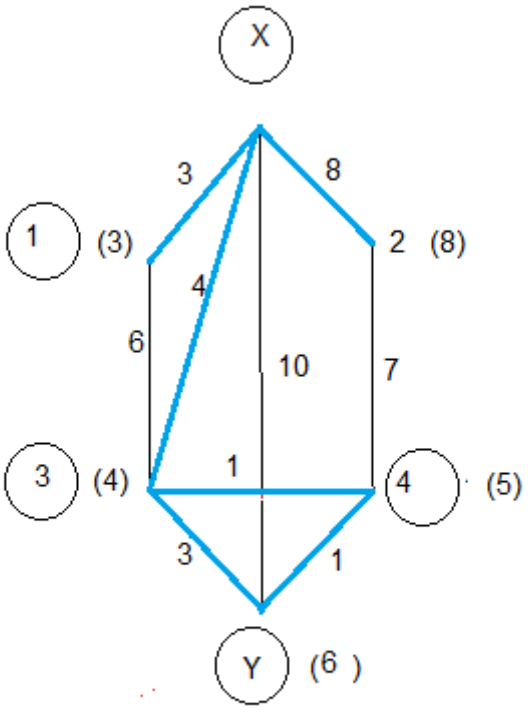
(C)

d [2] = min(8,4 +A[3,2]) = min (8,4 +∞) = 8
d[4] = min(∞,4 +A[3,4]) = min (∞,4 +1) = 5 (update s[4] to 3)
d [y] = min(10,4 +A[3,y]) = min (10,4 +3) = 7(update s [y] to 3), shown in figure (5-D) which shows the direct distance from x.



(D)

Shortest paths from x to the two nodes 4 and y can be found by going through IN number 3.
P = 4(d-value = 5)
IN = [x, 1, 3, 4]
D [2] = min (8,5 +7) = 8
D[y] = min (7, 5 +1) = 6 (update s [y])
Fig 3-12 shows the direct distance from x.
Processing the while loop again
p = [x, 1, 3, 4, y]
D [2] = min (8,6 +∞) = 8
Figure (6-D) shows the direct distance from x.
Now y is part of the while loop. The path goes through y, s[y] = 4, s [4] = 3 and s[3] = x. Thus the path uses nodes x, 3, 4, and y. The distance for the path is d[y] = 6 which is the shortest path from x to y, shown in figure (6-E).



(E)

6. CONCLUSIONS

Dijkstra's algorithm is used strongly to improve the networks congestion by selecting the shortest path for packets to move through the subnet nodes from a source node to destination node. This paper is highlighted the process of designing a simple subnet used the C-program for Dijkstra's algorithm to compute the shortest path through subnet nodes, this program is translated into a pseudocode in order to be used by any other language.

REFERENCES

[1] A. Felner, "Dijkstra's Algorithm versus Uniform Cost," SoCS fourth international proceeding, 2011.
[2] M. Reza, Z. Ahmed, A. Mamat, H. Zainuddin, "A hybrid algorithm for finding shortest path in network routing", JATIT 2009.
[3] K. H. Thomas, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction of Algorithms, MIT press, 2001.
[4] M. Sniedovic, " Dijkstra's algorithm revisited: the dynamic programming connexion", Control and Cybernetics, vol. 35, 2006.
[5] J. Sumitha,i, "Routing Algorithms in Networks", Res. J. Recent. Sci. Vol. 3 (ISC-2013), 1-3 (2014).

BIOGRAPHIES

Salah Elfaki Elrofai, Ph.D., Sudan University of Science and Technology, College of Engineering, School of Electronics. Eastern Diems, Khartoum, Sudan, P.O. Box 72. Ph.D. in electronic Engineering, Communication, SUST and Malaysia (UTM) 2007. M.Sc. in Computer Engineering & Networking (2002), Gezira University. BSc. (Honors), at Sudan University of Science and Technology in Electronics Engineering 1997. Area of Specialization: Communication Engineering (optical Communication Systems & devices). Assistant Professor Department of electronic Engineering Sudan University of science and technology (SUST). Chair of Electronic Department since May 2009 up to 2012 (Involved in evaluation team self- evaluation of undergraduate). Head of Scheduling and Exams for
Electronic Department. Evaluate and translate for computer Engineering Program and Telecommunication program
for Academy of Engineering Science, (AES). Assistant Professor Department of Electrical Engineering at AL-Baha University since 2012.

