Journal of Theoretical and Applied Information Technology
20th June 2014. Vol. 64 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A TRIANGULAR MATRIX ROUTING TABLE REPRESENTATION FOR EFFICIENT ROUTING IN MANET

[1]ABD AL-RAZAK TAREQ RAHEM, [2]MAHAMOD ISMAIL, [3]AWS SAAD
[1]Department of Electrical, Electronics and Systems Engineering, Faculty of Engineering and Built Environment, UKM, Malaysia
[2]Prof., Head of Department of Electrical, Electronics and Systems Engineering, Faculty of Engineering and Built Environment, UKM, Malaysia
[3] Computer Department, Faculty Al-Imam Al-Adham University College
E-mail: [1]abdtareq@yahoo.com, [2] mahamod@ eng.ukm.my. [3]awsit85@gmail.com

ABSTRACT

To improve the routing efficiency of a MANET, two issues need to be investigated: routing table size and path selection from source to destination. Given that routing performance depends on routing table size, which is determined by the number of nodes, a detailed table representation is required. In this study, we propose a triangular routing table representation to improve routing performance. By using a low triangular matrix, routing information can be transferred between nodes more efficiently by using a smaller memory size than the memory size used by conventional routing tables. The proposed routing representation is validated by simulating 25 MANET nodes that are generated randomly by shortest path selection. The simulation shows that the proposed approach significantly reduces the memory size in transfer routing information. Besides, it is proved through using 255 nodes that they require only 3.95 KB of memory size, which does not monopolize the throughput of a network.

Keywords: *Routing protocol, Routing table, Topology, Triangular matrix, Routing Triangular Matrix RTM.*

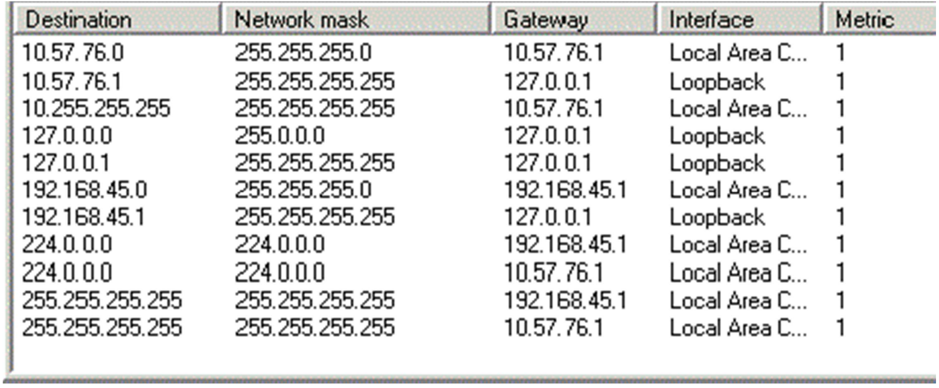
1. INTRODUCTION

MANET or mobile ad hoc network is a number of autonomous mobile nodes travelling inside a particular network. MANET is a type of temporary network that requires no infrastructure (server, router, or access point). MANET networks basically build temporary wireless networks and do not depend on any type of infrastructure for deployment or centralized administration [1-3]. Hence, all paths made by a routing protocol for each node to establish a link between the source node and destination node are based on the node movement inside the network. The network topology in MANET frequently changes during node movement. Therefore, Routing is important in MANET, whose routes data from the source node to the destination node [4]. A number of different approaches had been proposed to implement routing table in the past.

Thus, these protocols gather information about the network topology and save this information in a route repository,[5] which is an ether routing table or route cache, both of which consist of a data table that contains information on network routes.[6-8] To find any route, the routing table must be searched [9]. Therefore, routing information (routing table, route cache) is essential to routing protocols. Each node gathers knowledge on an entire network and shares it periodically (every 30 s) with neighboring nodes.[10, 11] The routing information consists of this message:

(-source node-destination node-next node-total cost-)

where (source node) refers to the sender source node address (destination node) refers to the receiver node address that should be reached. To reach the next node address, a node should use (destination node) to reach the destination via an optimal route (next node) based on the total minimal cost (total cost).The cost includes certain factors such as number of hops, bandwidth speed, delay, and lost packets. The nodes build their routing table whenever they receive such messages from neighboring nodes with one condition, that is, remove any duplicate entries from the table to reduce the routing table size, enhance search performance, and keep the entry with the minimal cost. In this manner, the routing table of each node will converge to the optimal routing table. Figure (1) shows an example of building a routing table.



Destination	Network mask	Gateway	Interface	Metric
10.57.76.0	255.255.255.0	10.57.76.1	Local Area C...	1
10.57.76.1	255.255.255.255	127.0.0.1	Loopback	1
10.255.255.255	255.255.255.255	10.57.76.1	Local Area C...	1
127.0.0.0	255.0.0.0	127.0.0.1	Loopback	1
127.0.0.1	255.255.255.255	127.0.0.1	Loopback	1
192.168.45.0	255.255.255.0	192.168.45.1	Local Area C...	1
192.168.45.1	255.255.255.255	127.0.0.1	Loopback	1
224.0.0.0	224.0.0.0	192.168.45.1	Local Area C...	1
224.0.0.0	224.0.0.0	10.57.76.1	Local Area C...	1
255.255.255.255	255.255.255.255	192.168.45.1	Local Area C...	1
255.255.255.255	255.255.255.255	10.57.76.1	Local Area C...	1

Figure (1). Routing Table.

The table is updated continuously and shared with neighboring nodes, thus affecting the impact of nodes on the throughput of the network and degrading network compatibility status in different parts of the network [6, 9]. Link state routing has recently been proposed as an efficient alternative to distance vector routing for solving the scalability problem in routing tables. Link state routing has two parts: gathering network information and creating the best routing table. However, the routing information is shared periodically with the entire network (but at larger intervals, e.g., 30 minutes). The link state packet refers to the routing information and has the following form:

(-source node-next node-link cost-)

Link cost is a general term that refers to delays, link length, bandwidth, and congestion. The link state uses a flooding mechanism to send packets to all nodes. This sending method is called announcement. For instance, if a new node is introduced, the costs of new nodes are introduced by neighbor nodes for all nodes in the network except itself. Furthermore, the link cost become infinite if a link fails. After a certain period (30 minutes or even hours), the node sends routing information periodically to other nodes, thus reducing traffic in the network [4, 11]. The management of routing table is the process of determining whether current route is still live or not using the parameters like source sequence numbers, destination sequence numbers, route request expiration timer and route caching timeout [10]. The route request expiration timer is used to invalidate all the entries of those nodes that do not lie on the path from the source to destination. The expiration time depends on the size of network. The route caching timeout is the time beyond which a route is no longer considered to be valid. There are no routing table entries for each valid root as maintained by a node, the node that can be precursors to forward packets on this list of routes. These traders of next hop link loss detection will receive notification from the node which contains routing table entry traders. List neighbor nodes generate a route to reply or forward [12].

However, the routing table increases in size again and becomes the largest issue in the routing table with a size of 40,000 entries or approximately more than 1 MB [13].

By analyzing and comparing wireless network protocols in terms of routing tables, particularly in MANET and wireless sensor networks (WSN), we discovered two major issues that have not been solved in previous studies. The first issue is that the routing table is extremely large. The second issue is finding the perfect path from the source node to the destination node. Thus, we propose a new approach called routing triangular matrix (RTM), which uses a triangular matrix instead of a table to transfer routing information. We implement this approach in a simulator by using MATLAB and obtained successful results.

This paper is organized as follows. Section 1 introduces the MANET routing protocol and routing table, with a problem statement concerning the routing table size. Section 2 presents the problem formulation and explains the network topology and triangular matrix mathematically. Section 3 discusses the representation of network topology inside the lower matrix with examples. This section also discusses the method of finding a route inside the lower matrix. Section 4 concludes and shows the simulation and experimental results obtained by using MATLAB.

2. PROBLEM FORMULATION

The essential thought in this section is how can network topology be represented mathematically and graphically.

2.1. Network topology

Network topology is the topological structure of a network and consists of various elements such as the nodes and links of computers, mobile phones, switches, and routers. Network topology is divided into two types: physical and logical topology. Physical topology refers to the placement of devices in the network, including various components, device location, and cable installation. Logical topology refers to how data flows through a network without physical design.

Topology is the mathematical study of shapes and spaces. For instance, let G(N, E) be the topology graph, N be the nodes, E be the links, and (i, j) be the link from Node I to Node J. Let the information be transmitted by f. The variable vector X_{ij}^{f} is then defined as follows:

$$X_{ij}^{f} = \begin{cases} 1, \text{if link } (i, j) \text{is used for flow f} \\ 0, \text{if linke } (i, j) \text{is used for flow f}' \end{cases}$$

The above equation indicates that the variable vector X_{ij}^{f} is one if link (i, j) is used to transmit flow f; otherwise, the vector is zero if the vector cannot be used for transmission. To reach Node 5, only two possible routes exist from Node 1: through the first path (i.e., (1, 2) and (2, 5)) and through the second path (i.e., (1, 3), (3, 4), and (4, 5)) (Figure 2). Suppose that the first path ((1, 2) and (2, 5)) is selected as the route because of transmission flows f1, X_{12}^{1} =1, and X_{25}^{1} =1 and that the values of the vector of links (1, 3), (3, 4), (4, 5) are equal to zero.

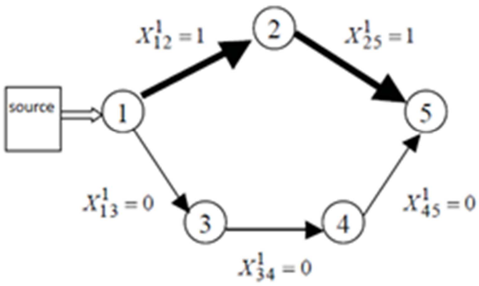


Figure (2). Graph Of Routes From Nodes 1 To 5.

Assume that N is the set of nodes and E is the set of links, where n denotes the number of network nodes, that is , n = |N|. Inside the network is a source node s □ N and a destination node t □ N , and (i, j) □ E is the link from Node i to Node j [3] .

A group of mobile nodes in a specific domain forms a network that can be represented in a graph where the nodes represent the network devices, such as routers, switches, modems, and satellites. The edges represent the physical connection link among devices and allow physical transmission between nodes. A graph is a data type structure that consists of two components: vertices or nodes and the edges or links that connect them. Graphs can either be undirected or directed. Undirected graphs consist of a group of nodes and a group of links with no direction between a pair of nodes.

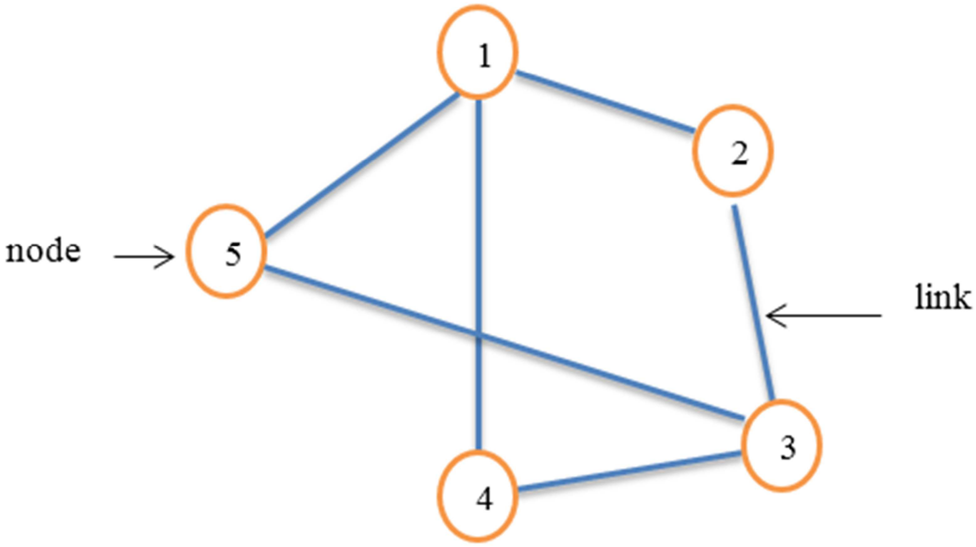


Figure (3). Undirected Graph.

Figure (3) shows five nodes and six links represented indistinctly. For example, links (1,2) and (2,1) are the same. By comparison, a directed graph is also group of nodes and a group of edges but the elements are ordered pairs of different nodes. Figure (4) shows a directed graph (with five nodes and six links) with the following links: {(1, 2), (1,4), (3,2), (3,4), (5, 1), (5,3)}. In this case, link (1, 2) exists but link (2, 1) does not because no direction from Node 2 to Node 1 is present; therefore, link (2, 1) does not exist.

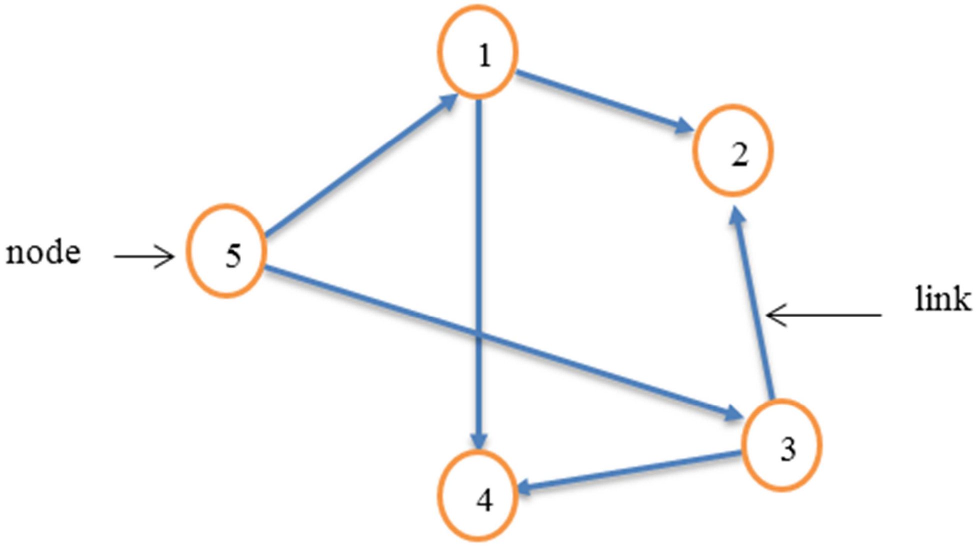


Figure (4). Directed Graph.

After representing a computer network mathematically and graphically, this study represents network topology using a triangular matrix to obtain full topology information using the least memory, as explained in the next section.

2.2. Triangular matrix

A triangular matrix is a specific type of square matrix. An upper triangular matrix and a lower triangular matrix exist inside each square matrix only if all opposite triangular matrix entries above the main diagonal are zero. A triangular matrix is either a lower or upper triangular matrix. Triangular matrixes are significant in numerical analysis because matrix equations with triangular matrixes are easy to solve. Variable U or R is commonly used for the upper triangular matrix, whereas variable L is commonly used to represent a lower triangular matrix. UM denotes an upper matrix, and LM denotes a lower matrix. Figure (5) shows both upper and lower triangular matrixes.

$$U = \begin{bmatrix} 1 & u_{12} & u_{13} & \cdots & u_{1n} \\ 0 & 1 & u_{23} & & \vdots \\ 0 & 0 & 1 & & \vdots \\ \vdots & & & \ddots & \vdots \\ 0 & \cdots & & 0 & 1 \end{bmatrix} \qquad L = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ l_{21} & 1 & 0 & & \vdots \\ l_{31} & l_{32} & 1 & & \vdots \\ \vdots & & & \ddots & 0 \\ l_{n1} & \cdots & & \cdots & 1 \end{bmatrix}$$

Figure (5). Upper Matrix And Lower Matrix.

The following are real examples about upper and lower triangular matrixes. The first example is a lower triangular matrix or left triangular matrix:

$$\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 3 & 5 & 1 \end{bmatrix}.$$

The second example is either an upper triangular matrix or a right triangular matrix:

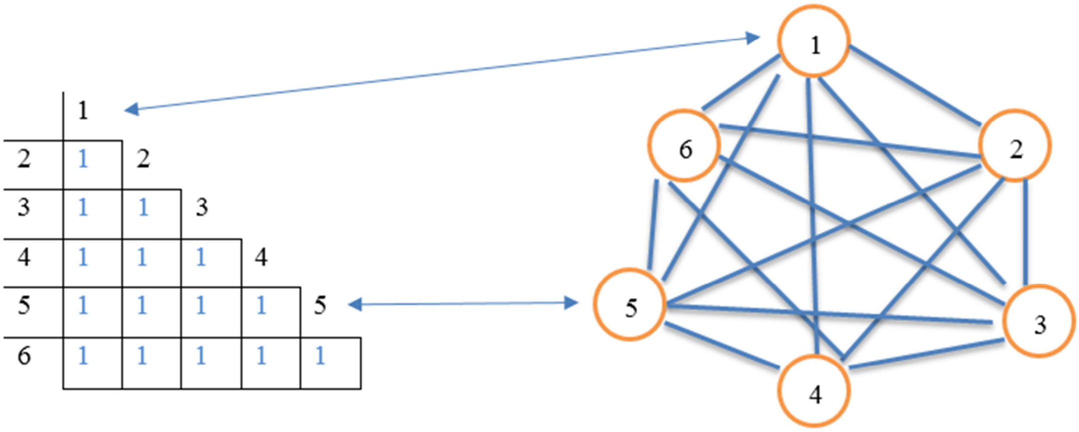
$$\mathbf{U} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & -1 & -5 \\ 0 & 0 & 18 \end{bmatrix}$$

Several operations can be applied to a triangular matrix: the sum of two upper triangular matrixes is an upper triangular matrix, the product of two upper triangular matrixes is an upper triangular matrix, and so on. This simple explanation about triangular matrixes will be used in the next section (implementation and design). We use the lower triangular matrix to represent routing information. By using a lower triangular matrix instead of a routing table to transfer routing information among nodes by using a new routing protocol algorithm, we achieve the optimal memory size used to transfer routing information [14].

3. IMPLEMENTATION AND DESIGN

3.1. Lower matrix

How can a lower matrix be extracted from a graph? How is a lower matrix represented from network topology? First, the dimensions of a triangular matrix is equal to a number of nodes and each node inside a network has a number that represents the diagonal of the lower matrix. At the beginning the content of the lower matrix must be empty (zero) (Figure (6)). Each link between two nodes represents one bit inside the lower matrix; otherwise, if no link exists between the two nodes, each link will be represented by a zero bit inside the lower matrix depending on the cross of the row node with the column node for nodes with links. For example, to represent a completely connected network (CCN) (Figure 6), the lower matrix has to be all one bit (ones):



	1	2	3	4	5
2	1				
3	1	1			
4	1	1	1		
5	1	1	1	1	
6	1	1	1	1	1

Figure (6). CCN Topology Representing A Lower Matrix.

The CCN can be represented mathematically by using digit pairs.

(1,2),(1,3),(1,4),(1,5),(1,6)

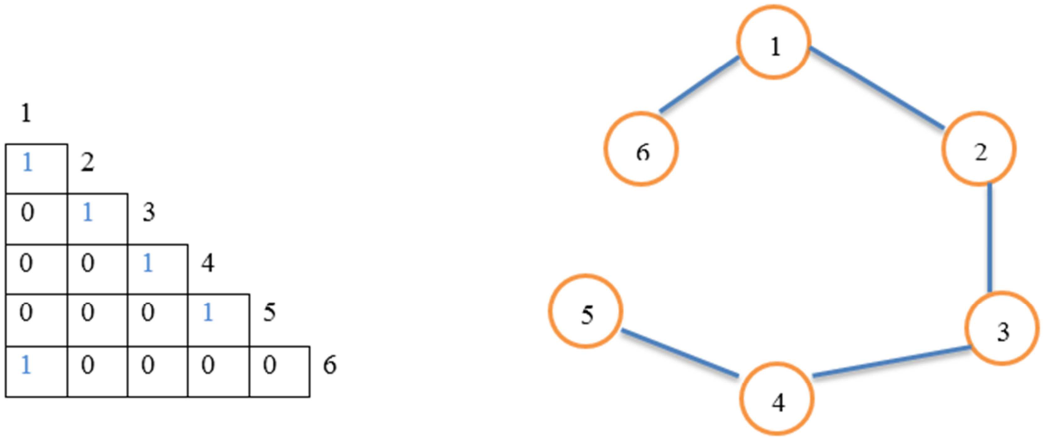
(2,3),(2,4),(2,5),(2,6)

(3,4),(3,5),(3,6)

(4,5),(4,6)

(5,6)

This is a second example, except that the network topology is not a CCN but rather a partially connected network (PCN) or a mesh network. Node 1 is connected with Node 2 as represented by the one bit cross of the first column with the second row, whereas no connection exists between Node 1 and Node 3 as represented by the zero bit in the cross of Column (1) with Row (3) and so on (Figure 7).



	1	2	3	4	5
2	1				
3	0	1			
4	0	0	1		
5	0	0	0	1	
6	1	0	0	0	0

Figure (7) PCN.

(1,2), (1,3),(1,4),(1,5), (1,6)

(2,3), (2,4),(2,5),(2,6)

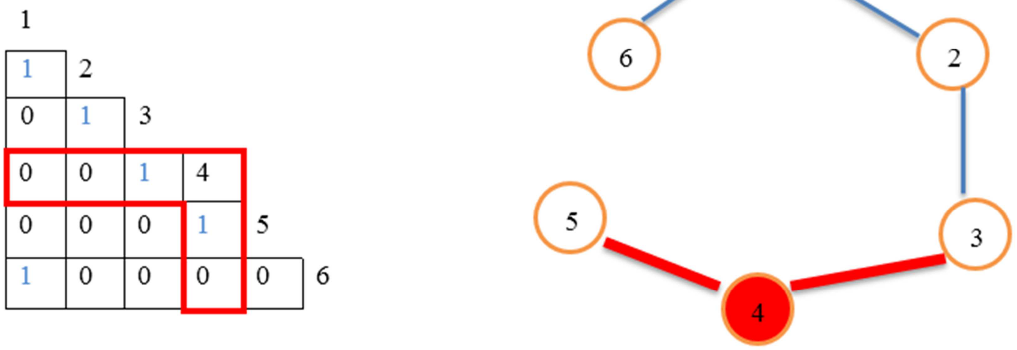
(3,4), (3,5),(3,6)

(4,5), (4,6)

(5,6)

3.2. Finding the route by using the lower matrix

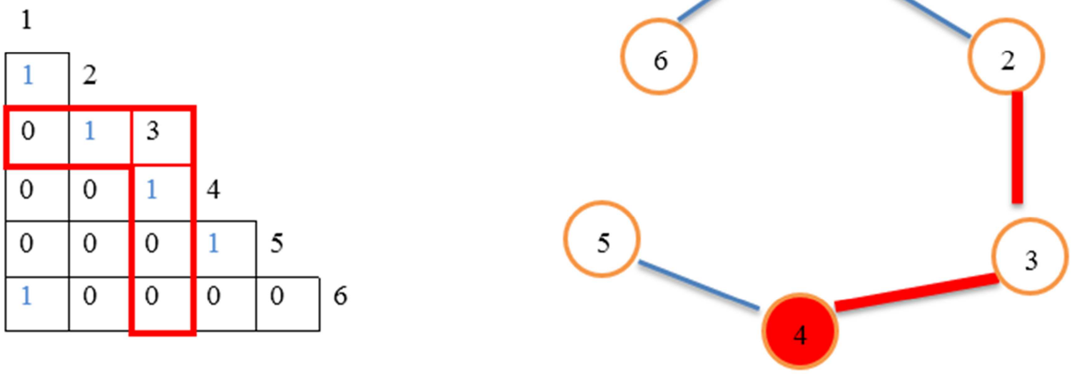
After obtaining the lower matrix from network topology, the issue is how to find a path from the source node to the destination node. What mechanism or algorithm is used to find routes? Figure (8) shows an example to explain how to find routes from the source to the destination node inside a triangular matrix. Suppose that the source node is Node (4) and the destination node is Node (6), we should search only inside the column and row of Node (4). If we find Bit (1) inside the lower matrix belonging to only Row and Column (4), we take the number of the node of the column crossed with Bit (1). At the same time, we take the node number of the rows crossed with Bit (1). In this example, Node (4) is crossed with Node (3) and Node (5) (Figure 8). Thereafter, we must search again for Nodes (3) and (5). Figure (9) shows that Node (5) has no (1) bit in neither row nor column except in Node (4). Therefore, we take Node (3) and apply a new search again inside the column and row of Node (3). We then extract the number of nodes crossed with (1) bit; thus, we reach Node (2) and so on until the destination node. Node (6) is reached by applying the same search mechanism. See Figures (10) and (11).



1					
1	2				
0	1	3			
0	0	1	4		
0	0	0	1	5	
1	0	0	0	0	6

Figure (8). Path From Nodes 4 To 5 And Nodes 4 To 3.

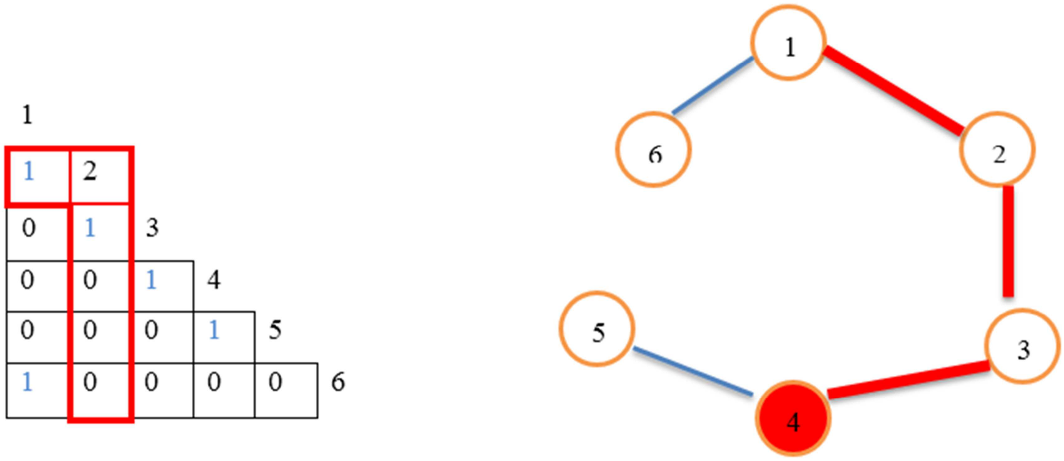
(1,2), (1,3),(1,4),(1,5), (1,6)
(2,3), (2,4),(2,5),(2,6)
(3,4), (3,5),(3,6)
(4,5), (4,6)
(5,6)



1					
1	2				
0	1	3			
0	0	1	4		
0	0	0	1	5	
1	0	0	0	0	6

Figure (9) finds a path from node 3 to node 2

(1,2), (1,3),(1,4),(1,5), (1,6)
(2,3), (2,4),(2,5),(2,6)
(3,4), (3,5),(3,6)
(4,5), (4,6)
(5,6)



1					
1	2				
0	1	3			
0	0	1	4		
0	0	0	1	5	
1	0	0	0	0	6

Figure (10) Finds A Path From Node 2 To Node 1

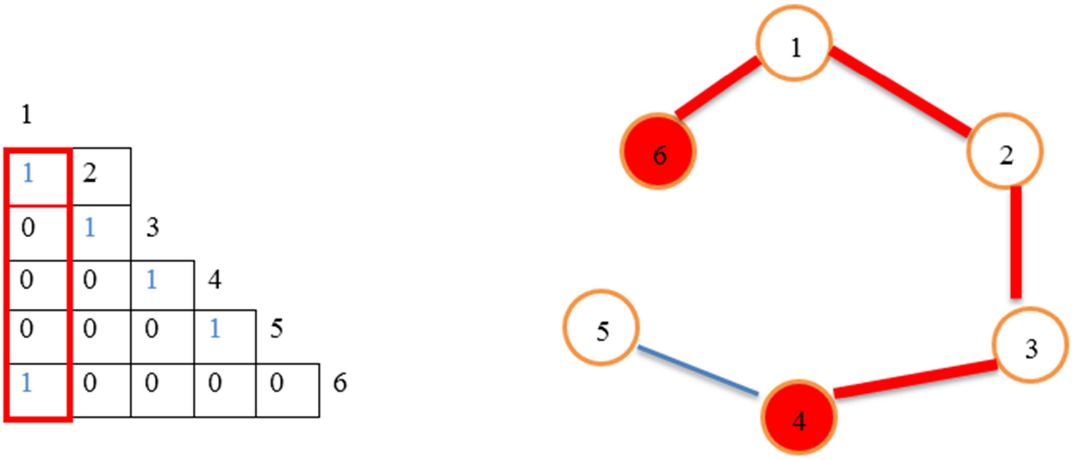
(1,2), (1,3),(1,4),(1,5), (1,6)

(2,3), (2,4),(2,5),(2,6)

(3,4), (3,5),(3,6)

(4,5), (4,6)

(5,6)



1					
1	2				
0	1	3			
0	0	1	4		
0	0	0	1	5	
1	0	0	0	0	6

Figure (11) Finds A Path From Node 1 To Node 6

(1,2), (1,3),(1,4),(1,5), (1,6)

(2,3), (2,4),(2,5),(2,6)

(3,4), (3,5),(3,6)

(4,5), (4,6)

(5,6)

4. SIMULATION AND EXPERIMENTAL RESULTS

According to our objectives, that is, reduce memory size and establish a new efficient routing table, we need to first select the appropriate simulation tool that satisfies all related objectives. Several network simulators are available in the market but the most frequently used are OPNET, Qualnet, and NS2. However, we select MATLAB because of its efficiency at handling matrixes . MATLAB needs to be implemented in RTM. We use the following network scenario: total number of nodes: 25, lower matrix: 300 bits, network topology (Figure 12), routing information: RTM (Figure 13).

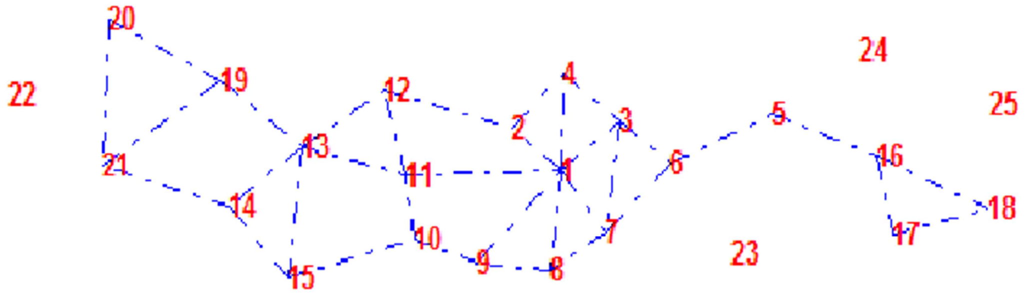
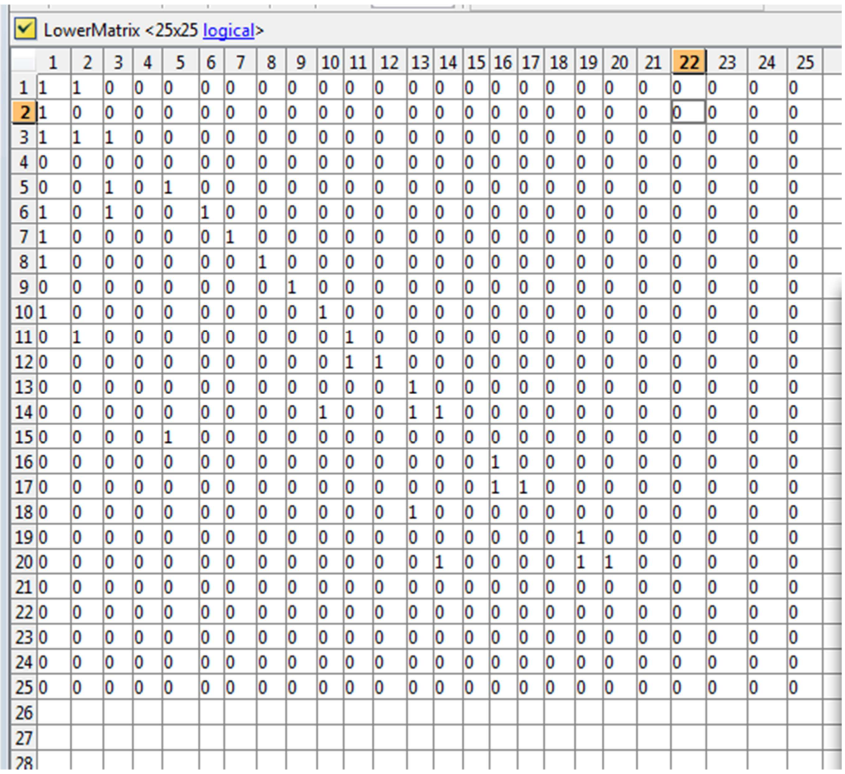


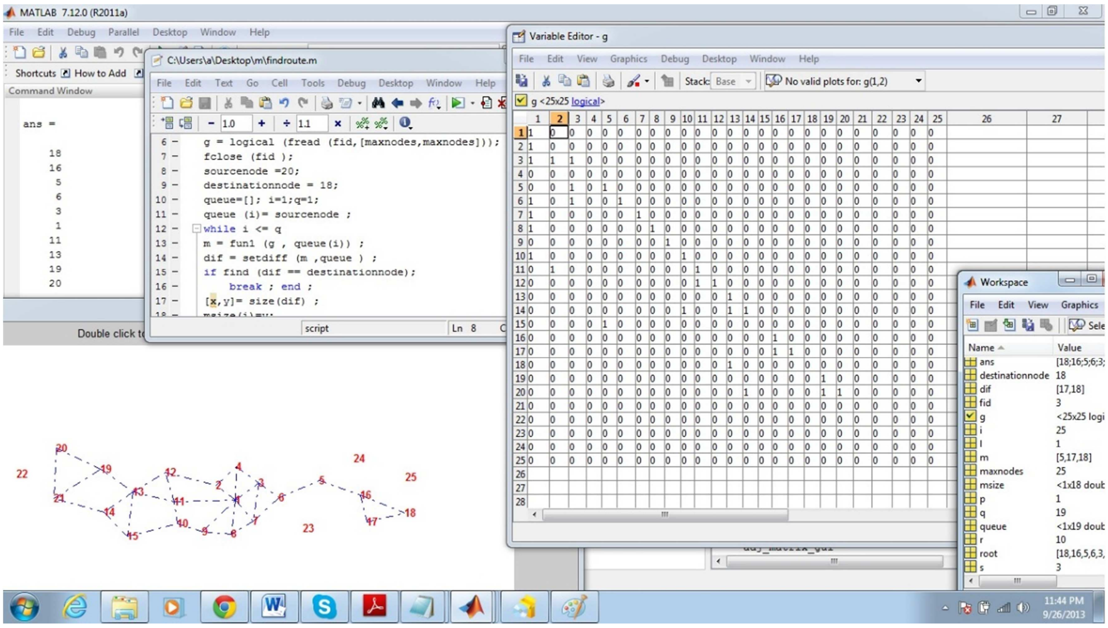
Figure (12). Network Topology For 25 Nodes.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26																									
27																									
28																									

Figure (13). RTM And Lower Matrix With Network Topology Contents.

From the simulation results obtained by using MATLAB, the routing information for each node save the triangular matrix calls the RTM. When no connection exists among nodes, the RTM is empty with zero memory size. If nodes want to communicate with each other, then RTM is generated depending on the numbers of nodes (at least two nodes). If we want to find a path from the source node to the destination node by using the RTM algorithm, for example, from source Node 20 to destination Node 18 (Figure 14), the path result obtained is {20,19,13,11,1,3,6,5,16,18}. Figure (14) shows the results on the top left window with the beginning word "ANS."



```
ans =

    18
    16
     5
     6
     3
     1
    11
    13
    19
    20
```

```
g = logical (fread (fid,[maxnodes,maxnodes]));
fclose (fid );
sourcenode =20;
destinationnode = 18;
queue=[]; i=1;q=1;
queue (i)= sourcenode ;
while i <= q
m = fun1 (g , queue(i)) ;
dif = setdiff (m ,queue ) ;
if find (dif == destinationnode);
    break ; end ;
[x,y]= size(dif) ;
```

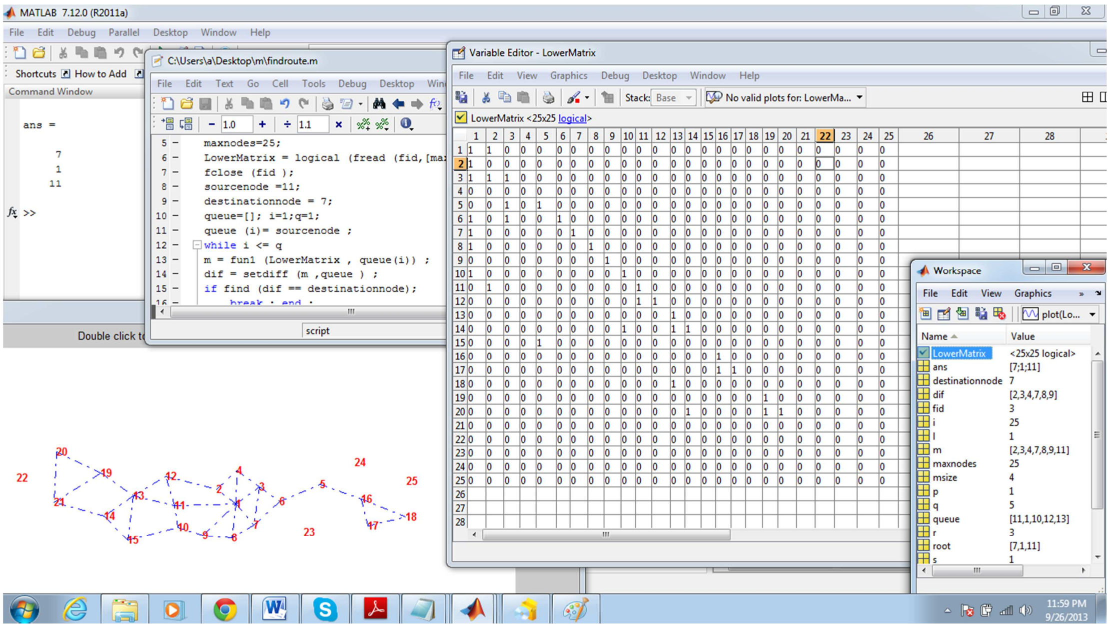
g <25x25 logical>

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Name	Value
ans	[18;16;5;6;3;
destinationnode	18
dif	[17,18]
fid	3
g	<25x25 logi
i	25
l	1
m	[5,17,18]
maxnodes	25
msize	<1x18 doub
p	1
q	19
queue	<1x19 doub
r	10
root	[18,16,5,6,3,
s	3

Figure (14). Route, Network Topology, Lower Matrix, And Part Of Program Commands.

Figure (15) shows how to find a path from source Node 4 to destination Node 15; the route is {4,1,9,10,15}.



```
ans =

     7
     1
    11

fx >>
```

```
maxnodes=25;
LowerMatrix = logical (fread (fid,[ma
fclose (fid );
sourcenode =11;
destinationnode = 7;
queue=[]; i=1;q=1;
queue (i)= sourcenode ;
while i <= q
m = fun1 (LowerMatrix , queue(i)) ;
dif = setdiff (m ,queue ) ;
if find (dif == destinationnode);
```

LowerMatrix <25x25 logical>

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Name	Value
LowerMatrix	<25x25 logical>
ans	[7;1;11]
destinationnode	7
dif	[2,3,4,7,8,9]
fid	3
i	25
l	1
m	[2,3,4,7,8,9,11]
maxnodes	25
msize	4
p	1
q	5
queue	[11,1,10,12,13]
r	3
root	[7,1,11]
s	1

Figure (15). Route, Network Topology, Lower Matrix, And Part Of Program Commands.

5. DISCUSSION

In the field of engineering, solving a problem is not enough, The solution found must be the best solution possible. However, calculating the size of the RTM requires knowing how many nodes are inside the network. Thus, RTM size = N (N − 1)/2 , where N refers to the number of nodes inside the network. For example, if the network has 25 nodes, the RTM size is equal to 300 bits; for 255 nodes, the RTM size is equal to 3.95 KB, which is highly inferior to the routing table in the best previous study at 0.59 MB [13]. Furthermore, the proposed method does not monopolize the throughput of a network, unlike proactive protocols or entire MANET protocols. Furthermore, the proposed algorithm finds routes from the source to the destination node more easily than hybrid protocols. In our results, we apply the shortest path approach (hop count) to find a path. However, this approach is not the only approach available for dealing with end-to-end delay, packet loss rate, and bandwidth consumption. Each node inside a particular network needs to build a routing table by itself and update the table periodically; thus, each node has a routing table that is different from each of the other node. In the end, this method reduces network bandwidth and node performance. However, in the RTM approach, all nodes have the same RTM that is updated only if a new node joint network is established or a node disconnects from a specific node, thereby reducing the consumption of network bandwidth and increasing node performance. Applying a limited search inside the lower matrix leads to high performance in reaching the target. This approach uses less time than a routing table because the system does not search inside the whole lower matrix or inside the whole table to reach targets. Obtaining a node number requires only a search for (1) bit inside the column and row belonging to that node itself.

“Fault tolerance,” which means finding multiple paths to reach the destination node, must be considered a main objective at the design level of routing protocols. Fault tolerance can be easily applied to RTM.

The RTM is a sufficient, optimal solution in the field of MANET routing protocols, VANET, and WSN because the RTM provides full network topology information even though the network topology for the MANET networks is not fixed because of frequent node movement. Node movement results in frequent topology changes. RTM provides network topology information to each node despite node movements; this feature is the essential point of using RTM. The new routing information approach is most significant in decentralized networks.

We have to keep RTM secure because it provides full routing information about the network, thus making RTM a hacking target.

6. CONCLUSION

In this paper, we proposed a new type of routing information storage called RTM as an alternative to the routing table and we simulated RTM by using MATLAB. We also explained how to represent nodes inside the lower matrix from network topology. We then used MATLAB to find the shortest path from the source node to the destination node inside RTM. Finally, we proved that the size of RTM is small (3.95 KB for 255 nodes) with CCN. However, given that we only implemented RTM, we still need to apply RTM to one of the MANET routing protocols or WSN routing protocols in a future study.

REFERENCES:

[1] A. Boukerche, B. Turgut, N. Aydin, M. Z. Ahmad, L. Bölöni, and D. Turgut, "Routing protocols in ad hoc networks: A survey," Computer Networks, vol. 55, pp. 3032-3080, 9/15/ 2011.

[2] S. Buruhanudeen, M. Othman, M. Othman, and B. M. Ali, "Existing MANET routing protocols and metrics used towards the efficiency and reliability- an overview," in Telecommunications and Malaysia International Conference on Communications, 2007. ICT-MICC 2007. IEEE International Conference on, 2007, pp. 231-236.

[3] R. F. Yezid Donoso, Multi-Objective Optimization in Computer Networks Using Metaheuristics. Boca Raton New York: AUERBACH PUBLICATIONS, Dec 12, 2010.

[4] Z. Liu, J. Kim, B. Lee, and C. Kim, "A routing protocol based on adjacency matrix in ad hoc mobile networks," in Advanced Language Processing and Web Information Technology, 2008. ALPIT'08. International Conference on, 2008, pp. 430-436.

[5] A. Hiromori, H. Yamaguchi, K. Yasumoto, T. Higashinoz, and K. Taniguchi, "Reducing the size of routing tables for large-scale network simulation," in Parallel and Distributed Simulation, 2003. (PADS 2003). Proceedings. Seventeenth Workshop on, 2003, pp. 115-122.

[6] P. Fong and T. Nian-Feng, "Concise Lookup Tables for IPv4 and IPv6 Longest Prefix Matching in Scalable Routers," Networking, IEEE/ACM Transactions on, vol. 20, pp. 729-741, 2012.

[7] L. Hoang and V. K. Prasanna, "Scalable Tree-Based Architectures for IPv4/v6 Lookup Using Prefix Partitioning," Computers, IEEE Transactions on, vol. 61, pp. 1026-1039, 2012.

[8] R. Sangireddy, N. Futamura, S. Aluru, and A. K. Somani, "Scalable, Memory Efficient, High-Speed IP Lookup Algorithms," Networking, IEEE/ACM Transactions on, vol. 13, pp. 802-812, 2005.

[9] M. Bando and H. J. Chao, "FlashTrie: Hash-based Prefix-Compressed Trie for IP Route Lookup Beyond 100Gbps," in INFOCOM, 2010 Proceedings IEEE, 2010, pp. 1-9.

[10] B. T. Sharef, R. A. Alsaqour, and M. Ismail, "Comparative Study of Variant Position-based VANET Routing Protocols," Procedia Technology, vol. 11, pp. 532-539, 2013.

[11] C. Zhongyu and L. Gang, "S-AODV: Sink Routing Table over AODV Routing Protocol for 6LoWPAN," in Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010 Second International Conference on, 2010, pp. 340-343.

[12] S. Lee, R. Muhammad, and C. Kim, "A Leader Election Algorithm Within Candidates on Ad Hoc Mobile Networks," in Embedded Software and Systems. vol. 4523, Y.-H. Lee, H.-N. Kim, J. Kim, Y. Park, L. Yang, and S. Kim, Eds., ed: Springer Berlin Heidelberg, 2007, pp. 728-738.

[13] R. C. Chang and B. H. Lim, "Efficient IP routing table VLSI design for multigigabit routers," Circuits and Systems I: Regular Papers, IEEE Transactions on, vol. 51, pp. 700-708, 2004.

[14] S. Axler, Linear Algebra Done Right: Springer-Verlag, 1996.