ISSN No. 0976-5697

Volume 3, No. 4, July- August 2012

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

A Survey on Clustering Algorithms in MANET using Spanning Tree

Dr. H. B. Walikar[1], Ravikumar H. Roogi*[2] and Shreedevi V.Shindhe[3]
Professor[1], Research Scholar[2] and Research Scholar[3]
Dept. of Computer Science
Karnatak University Dharwad, India
hbwalikar@gmail.com[1], raviroogi@gmail.com[2], svshindhe@gmail.com[3]

Abstract: This paper aims to provide a survey of existing graph models and algorithms that arise in searching, sorting, network structures and MANET's. In particular we discuss the graph problems on spanning tree, minimum spanning tree and clustering algorithms in MANET. An ad hoc network is a multihop wireless communication network supporting mobile users without any existing infrastructure. Clustering provides a method to build and maintain a hierarchical address in ad hoc network. This paper highlights the concept of graph algorithms that address searching, sorting, network structures and MANETs.

Keywords: spanning tree, minimum spanning tree, clustering, mobile ad hoc network (MANET).

I. INTRODUCTION

Mobile Ad-hoc networks have been widely researched for many years. Research on wireless Ad-hoc networks has been ongoing for decades. Mobile Ad-hoc networks do not have any fixed infrastructure and consist of wireless mobile nodes that perform various data communication tasks. The history of wireless ad-hoc networks can be traced back to the Defense Advanced Research Project Agency (DAPRAPA) packet radio networks (PRNet), which evolved into the survivable adaptive radio networks. Ad-hoc networks have play an important role in military applications and related research efforts, for example, the global mobile information systems (GIOMO) program [1] and the near-term digital radio (NTDR) program. MANETS's have potential applications in rescue operations, mobile conferences, battle field communications etc.

Conserving energy is an important issue for MANET's as the nodes are powered by batteries only. Clustering has become an important approach to manage MANET's. Clustering has many advantages in mobile networks. Cluster algorithms have been widely used in MANET to determining subsets of nodes for saving energy, enhancing routing protocols, finding efficient flooding and broadcasting, or to generally low-cost backbones.

When dealing with graphs, one fundamental issue is to searching, sorting in other words to visit each vertex and each edge. To solve this problem, many interesting algorithm exist.

In this survey, we search various graph algorithms for Clustering in MANET using spanning tree. The Tree definition, Spanning tree definition and method of producing spanning tree by using BFS and DFS, the Minimum spanning tree definition and method of producing minimum spanning tree by Kruskal's algorithm and Prim's algorithm are given in section 2, clustering using spanning tree given in section 3 and section 4 concludes the paper.

II. TREE

Trees are the most commonly occurring type of graph in models and computation of all kinds. Trees as graph have many applications, especially in data storage and communication.

Graph T is a tree if [2]:

a. One and only one simple path between every vertex pair (i, j)
b. If $N = |V|$:
   a) Exactly $N-1$ edges (arcs)
   b) Connected, no cycles.

A spanning tree is a spanning sub graph that is a tree.

A sub graph of a graph $G(V, E)$[2] is obtained as follows:

a. $G'(V'E')$, Where V' and E' are subsets of V and E respectively.
b. For every arc (edge) belonging toE', both i and j must belong toV'.

A sub graph $T(V'E')$ of $G(V, E)$ is a spanning tree for G if

a. T is atree.
b. $V' = V$

There is no unique spanning tree in general.

Many graph algorithms require a systematic method of visiting the vertices of graph. In this section, we describe two types of spanning tree that can produce some particularly efficient algorithms.

The breadth first search (BFS) and the depth first search (DFS) are such methods.

a. ***BFS:*** BFS is an uniformed search method that aims to expand and examine all nodes of a graph or combination of sequence by systematically searching through every solution. In other words, it exhaustively searches the entire graph or sequence without considering the goal until it find it. We describe BFS search in pseudo code as Algorithm. In this algorithm, we assume the vertices of the connected graph G are ordered as $v_1, v_2, \dots . v_n$. In the algorithm we uses the term "process" to describe the procedure of adding new vertices, and corresponding edges, to the tree adjacent to the current vertex being processed as long as a simple circuit is not produced [3].

Algorithm: BFS [3]:

```
procedure BFS ($G$: connected graph with vertices $v_1$, $v_2$, ….$v_n$)
$T$ := tree consisting only of vertex $v_1$
$L$ := empty list
put $v_1$ in the list $L$ of unprocessed vertices
while $L$ is not empty
begin
   remove the first vertex $v$, from $L$
   for each neighbor $w$ of $v$
     if $w$ is not in $L$ and not in $T$ then
     begin
        add $w$ to the end of the list $L$
        add $w$ and edge $\{v, w\}$ to $T$
      end
end
```

The computational complexity of BFS algorithm uses $O(G)$ or $O(n^2)$ steps, where e and n are no. of edges and vertices in G respectively.

b. ***DFS:*** DFS is a uniformed search that progress by expanding the first child node of the search tree that appears and thus going deeper and deeper until a goal node is found, or until it hits a node that has no children. Then the search back tracks, returning to the most recent node it hasn't finished exploring. In a non-recursive implementation, all freshly expanded nodes are added to stack for exploration. We describe DFS search in pseudo as Algorithm. In this algorithm, we construct the spanning tree of a graph G with vertices v_1, v_2, ….v_n by first selecting the vertex v_1 to be the root. We initially set T to be the tree with just this one vertex. At each step we add a new vertex to the tree T together with an edge from a vertex already in T to this new vertex and we explore from this new vertex [3].

Algorithm DFS *[3]:*

```
procedure DFS ($G$: Connected graph with vertices $v_1$, $v_2$, ….$v_n$)
$T$ := tree consisting only of the vertex $v_1$
visit ($v_1$)
procedure visit (: vertex of $G$)
for each vertex $w$ adjacent to and not yet in $T$
begin
    add vertex $w$ and edge $(v, w)$ to $T$
    visit ($w$)
end.
```

The computational complexity of the DFS constructs a spanning tree using $O(e)$ or $O(n^2)$, steps where e and n are no. of edges and vertices in G respectively.

c. ***Minimum Spanning Tree***: We seek a spanning tree of G whose weight is minimum among all spanning trees of G. Such a spanning tree is called minimum spanning tree [4].

The importance of the Minimum Spanning Tree problem is due to its applications in the design of computer, communications and transportation networks. The history of this problem was researched by Ronald L. Graham and Pavol Hell in 1985. They concluded that the minimum spanning tree problem was initially formulated by Otakar Boruvka in 1926 because of his interest in the most economical layout of a power line network. He also gave the first solution of the problem prior to Boruvka, however, the anthropologist Jan Lzeknowski's work on classification schemes led him to consider ideas closely related to the Minimum Spanning Tree problem [4].

We will present two algorithms for constructing minimum spanning trees. Both proceed by successively adding edges of smallest weight from those edges with a specified property that have not already been used. Kruskal Algorithm and Prims Algorithm

To carry ***Kruskal algorithm***, choose an edge in the graph with minimum weight, successively add edges with minimum weight that do not form a simple circuit with those edges already chosen. Stop after $n-1$ edges have been selected [3].

Algorithm Kruskal's *[3]:*

```
procedure kruskal ($G$: weighted connected undirected graph with $n$ vertices)
$T$ := empty graph
for $i := 1$ to $n-1$
begin
   $e$ := any edge in $G$ with smallest weight that does not form a simple circuit
   when added to $T$
   $T := T$ with $e$ added
end {$T$ is a minimum spanning tree of $G$}.
```

The complexity to find the minimum spanning tree of a graph with e edges and vertices using kruskal can be carried out using $O\ (e \log e)$.

To carry out ***Prim's algorithm***, begin by choosing any edge with smallest weight, putting it into the spanning tree. Successively add to the tree edges of minimum weight that are incident to a vertex already in the tree and not forming a simple circuit with those edges already in the tree stop when $n-1$ edges have been added [3].

Algorithm Prim's *[3]:*

```
procedure Prim ($G$: weighted connected undirected graph with $n$ vertices)
$T$ := a minimum- weight edge
for $i := 1\ to\ n-2$
begin
    $e$ := an edge of minimum weight incident to a vertex in $T$ and not forming a simple circuit in $T$ if added to $T$
    $T := T$ with $e$ added
end {$T$ is a minimum spanning tree of $G$}.
```

The complexity to find the minimum spanning tree of a graph with e edges and vertices using prim's can be carried out using $O\ (e\ log\ v)$.

III. CLUSTERING USING SPANNING TREE

An undirected graph is defined as $G = (V, E)$, where V is a finite nonempty set and $E \subseteq V \times V$. The V is a set of nodes v and the E is a set of edges e. A graph $G_s = (V_s, E_s)$ is a spanning subgraph of $G = (V, E)$ if $V_s = V$. A spanning tree of a graph is an undirected connected acyclic spanning subgraph. Intuitively, a minimum spanning tree (MST) for a graph is a sub graph that has a minimum number of edges for maintain connectivity [5]. A sample MANET and a minimum spanning tree constructed can be seen in Fig. 1 where any node other than the leaf nodes which are shown by black color depict a connected set of nodes.

Gallagher et. Al [6] proposed a distributed algorithm for minimum- weight spanning tree for an undirected graph that has distinct finite weights for every edge. Aim of this algorithm is to combine given fragment of an MST, let e be a minimum – weight outgoing edge of the fragment. Then joining e and its adjacent non- fragment node to the fragment yields another fragment of an MST. If all the edges of a connected graph have different weights, then the MST is unique. In this algorithm start with fragments of one node, enlarge fragments in any order and combine fragments with a common node.

a. ***Fragments:*** Every node starts as a single fragment, each fragment finds its minimum outgoing edge then tries to combine with the adjacent fragment.
b. ***Levels:*** Every fragment has an associated level that has impact on combining fragments. A fragment with a single node is defined to be at level 0. The combination of two fragments depends of the level of fragments.
   a) If a fragment F wishes to connect to a fragment F^1 and $L < L^1$ then: F is absorbed in F^1 and the resulting fragment is at level L^1.
   b) If fragments F and F^1 have the same minimum outgoing edge and $L = L^1$ then the fragments combine into a new fragment F^{11} at level $L^{11} = L + 1$.
   c) If a fragments F and F^1 with same level were combined, the combining edge is called to core of the new segment.

The author defined three possible nodes state:
   a) Sleeping- initial state.
   b) Find – during fragments search for a minimal outgoing edge.
   c) Found – otherwise (when a minimal outgoing edge was found.)

The upper bound of the algorithm is $5N \log_2 N + 2E$ where N is the no. of nodes and E is the no. edges in the graph.

Awerbuch [7] proposed an algorithm in which each tree will hook itself on edge leading to the neighboring tree of maximum level instead of hooking itself on its minimum weight edge. The algorithm consists of two stages.
   a) The first stage, referred to as counting stage, finds some spanning tree and computes the no. of nodes in network.
   b) The second stage, referred to as MST stage, receives as an input the no. of nodes in the network and using this information finds the minimum weight spanning tree.

Both stages require $O(E + V\,logV)$ messages and $O(V)$ time, i.e are optimal both in communication and time. Where E is no. of edges. This has a lot similarity with Gallagher et.al's algorithm [6]. The only difference is that level increases are originated by many nodes, not only the root node. The first stage runs an algorithm indicate to Gallagher et.al's algorithm [6]. The new algorithmic ideas are introduce in the second stage.

Garay et.al [8] provides a modified, controlled version of the Gallagher et.al's algorithm [6].
Garay et.al [8] algorithm consists of three parts.

a. ***Controlled GHS***: is a modified variant of original algorithm of [GHS]. The purpose of the modification is to produce a balanced outcome in terms of number and diameter of the resulting fragments. This is achieved by computing in each phase a small dominating set on the fragment forest and merging fragments accordingly. This, in turn, is achieved by invoking the distributed minimal independent set (MIS) algorithm of [PS]. At the end of this phase, we left with "small" no. of fragments which have "small" diameter.
b. ***Cycle elimination:*** uses pipelining techniques to gather information in each fragment center, and uses this information to perform the elimination of some of possible multiple inter – fragment edges in the fragment graph. This leaves us with sparse set of inter-fragment edges from which the final tree is to be selected.

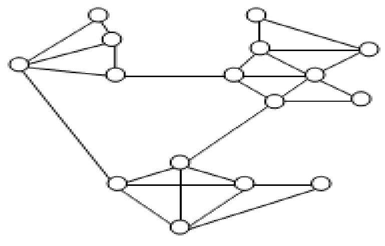
Figure 1.a A MANET

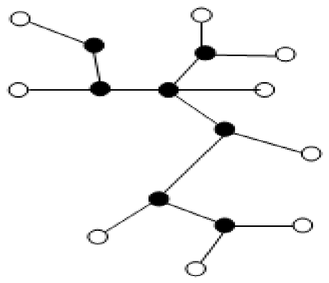
Figure 1.b ItsMinimum Spanning Tree

c. ***Global edge elimination***: performs the global elimination of most of the remaining edges, leaving only a tree connecting the fragments, and thus yielding the final MST. This elimination process carried out on a super imposed breadth first search structure, this yields the final MST tree.

The time complexity of the algorithm is $O(Diam(G) + n^{0.614})$ where n is the no. of nodes.

The algorithm proposed by Gallagher et.al [6] and Awerbuch [7] uses Tree- join- tree approach. Yao- Nan Lien [9] proposed a distributed minimum spanning tree algorithm that uses Node- join- tree approach. The proposed algorithm is initialized from single node such that there is no need to wake up all nodes at the beginning. The algorithm is executed in 5 steps. Step 1 and step 5 are only executed at the root node of each iteration while step 2 to step 4 are executed at the other nodes.

For convenient, the colors Red, Green, Blue, Yellow and White are used to mark the edges and nodes during the tree growing process.

i. ***Step 1:*** initially, all edges and nodes are marked white; all nodes are in the state 'Asleep'. An arbitrary node wake up as the first root node and mark itself green. The node then executes step 1 to initiate the process.
ii. ***Step 2:*** The new root node sends a 'Follow- me' message to each of its neighboring nodes except its preceding node and enters the 'Wait for terminate' state.
iii. ***Step 3:*** receiving a 'Follow- me' message from particular edge, a node is waken up if it is in 'A

sleep' state and enter the 'Wait- for- draft' sleep. Once it is waken it executes following steps.

a) If it is in M and 'Wait- for- reply' state, it marked the edge red.
b) If it is in M and 'Wait- for- draft' state, it replies 'Red' message and marked the edge red.
c) If it is in 'Wait- for- draft' and its minimum edge is not emanating to drafting node, it replies 'Yellow' message and marked the edge and node yellow.

Yellow nodes are neighboring nodes of M. To reduce overhead, a yellow node can reply with a 'Red' messages and mark the edge red if the weight of the edge is not smaller that of yellow edge it has marked. They are wairing for a chance to be a new root.

a) Otherwise, its minimum edge is emanating to the drafting node. The node decides to hook itself to M as a new terminal node of M. All yellow edges adjacent to the node are changed to red. Then, it initiates a new drafting process by sending a 'Follow- me' message to all neighboring white edges. Finally it enters 'Wait- for- reply' state.

iv. ***Step 4:*** Upon receiving a reply from an edge, a node in the state 'Wait- for- reply' will mark the edge according to the color of the replying message and execute step 5 if all reply have been received.

v. ***Step 5:*** Receiving all replies, a node marks itself and the edge emanating to the drafting node as green reports a 'Green' message to the drafting node, then enters the state 'Idle'. The edge emanating to the drafting node is its up- hook with respect to the current root node. The content of a number in the 'Follow- me' and 'Red' message such that each node in M knows at which each iterations its neighbors and itself are induced. The yellow nodes in the 'Root- migration' messages are also associated with iteration numbers. In this way, the new terminal nodes continue the drafting process iteratively until the end of the iteration where there is no node that wants to hook M.

A complete MST is formed if all nodes are included in M. The algorithm needs at most $(2e + n(n-1)/4)$ messages in $O(n^2)$ time where e is the no. of edges, n is the no. of nodes. In best case, it needs $2e$ messages in $O(n\ logn)$ time.

Banerjee and khuller [10] proposed a protocol based on a spanning tree hierarchical routing in wireless network. In their scheme, they present a clustering scheme to create hierarchies for wireless network. A cluster is defined as a subset of vertices, whose induced graph is connected. These subsets are chosen by considering a cluster size and the maximum number of the clusters to which a node can belong. [10] defined their clustering problem in a graph theoretic frame work and present an efficient distributed solution that meets all their desirable properties. For generic graph topologies, sometimes no solution may exit that can satisfy all the requirements of a desirable solution. The algorithm proceeds by finding a rooted spanning tree of a graph. The algorithm creates a BSF tree and then visits each vertex in the tree in post- order. The time complexity of the algorithm is $O(|E|)$ where E is the no. of edges.

Srivastava and Ghosh [11] proposed a distributed algorithm for constructing a rooted spanning tree of a dynamic graph such that root of the tree is located near the center of the graph. They described the $\propto$ cone as the origin concerned node and bounded by two rays with an angle $\propto$ between them. The attribute color is given for each node to define their states.

a. White: parent pointer $=\emptyset$, child list $=\emptyset$
b. Gray: parent pointer $=\emptyset$, child list $\neq \emptyset$
c. Black: parent pointer $\neq \emptyset$.

The algorithm proposed works in two phases.

a) Phase 1: in this phase finds spanning forest, every node executes the procedure find- the parent (). Nodes that are black do not accept any children since they may result in the formation of cycles.
b) Phase 2: the trees of spanning forest are connected together to produce a tree with a single root.

The authors proposed a priority- based algorithm for the second phase. A clustering based analysis are not provided by the authors.

Arun B. R., L. C [12] generate a minimum cost spanning tree for a given network of N- nodes using an efficient algorithm and then they study the problem of constructing a K- node Multicast Minimum Spanning Tree (KMMST) for any given multicasting group with K- nodes, where K is less than n.

Algorithm to compute K- nodes Multicast Spanning Tree Input: Link Cost Adjacency (LCA) matrix of a given network

a. Compute the shortest path from the given multicast source to each of the multicast destinations using Dijikstra's shortest path algorithm.
b. Construct the new link cost adjacency matrix based on the source and multicast destinations with costs $C(i, j) =$ cost of shortest path between i and j.
c. Compute K nodes multicast spanning tree applying prim's algorithm on new LCA matrix.

Detailed analysis of the algorithm and simulation results is provided by authors.

A k- clustering is a frame work in which the wireless network is divided into non- overlapping sub networks, also referred to as clusters, and where every two wireless hosts in sub network are at most k hops from each other. The algorithmic complexity of k- clustering is known to be NP-complete for simple undirected graphs. Fernandess and Malkhi [13] proposed a k- clustering frame work to divide the network into non- overlapping clusters. They modeled MANET as unit disk graphs and introduced two phase distributed polynomial time and message complexity approximation for k- clustering where $k > 1$, that has a competitive worst case ratio of $O(k)$.

a. ***First phase:*** constructs a spanning tree of the network. They showed tree can be built as a BFS, DFS or MCDS (Minimum Connected Dominating Set) structure and compared these structures.
b. ***Second phase:*** partitions the spanning tree into sub- tree with bounded diameter. Given a tree $T = (V, E)$ the algorithm finds a sub- tree whose diameter exceeds k, it then detaches the highest child of the sub- tree and repeats over he reduced tree.

Detailed analysis of the algorithm is provided but simulation results are not given by authors.

P. Victer., T.V., [14] proposed a distributed algorithm to formulate of Distributed Spanning Tree (DST) in MANET with respective constraints. The DST Algorithm uses five procedures:

- ***a.*** ***DST_initialze ():*** a procedure which initializes DST by creating Head Node (HN) in MANET based on some test criteria. Each HN is provided with unique Priority Number (PN) to provide write priority among the HNs.
- ***b.*** ***DST_Probe ():*** called by every HN creates probe message and set 'id' field of message as its own id and flood the message to all nodes if it connected. Also creates Leaf Node (LN).
- ***c.*** ***DST_recieve ():*** where receiving a message every node executes this procedure where 'message' is received.
    - b) If the message is received by a HN, then it is just discarded.
    - c) If the message is received by a LN, which is not under any HN, then LN stores the Head variable as id which is read from pmsg. Then call the procedures DST_reply (N(id)) and DST_forward (pmsg).
- ***a.*** ***DST_reply (N(id)):*** called by LN to reply to its HN. After the reply message sent to HN, the LN calls.
- ***b.*** ***DST_forward (pmsg):*** To flood the probe message to all the nodes except the node from where it was received .

After the completion of these five procedures the MANET will be in required DST structure. The time taken for formulation of DST in MANET using the proposed algorithm is very small and effective in routing and cost reduced communication.

Hichem Megharbi, H. K., [15] proposed a virtual spanning tree in ad hoc networks as a analogue of the fixed communication infrastructure or distributed data structure in wired networks and propose simple distributed algorithms where each node makes local decision for construction and maintenance of a such tree in a mobile ad hoc network. The algorithm for construction a spanning tree is described in following steps.

- a. Initialization step: each node is in passive state.
- b. Construction step: any passive node becomes active selected by one of its active neighbor v'. Any active belongs to a sub tree which may be reduced to it. All nodes of a sub tree have the same color. Two distinct sub trees have different colors. Each active node sweeps its neighbor node and try to add new nodes to its sub tree, so it can meet either a passive node or an active node:
    - a) Invitation of passive neighbors to become active: any active nodes detect these passive neighbors and invite them to become active and take its color. The corresponding links are also active. In this case any active node must invite at most limited degree passive nodes to become neighbors in a spanning tree.
    - b) Sub tree fusion: any active node v of a sub tree T_v detects an active neighbor v' of another sub tree$T_{v'}$. After a given time, the active node of T_v which has the smallest detection time actives the link with the corresponding sub tree. So we obtain by composition only one sub tree. The color of new sub tree is given by the composition of the colors of two sub trees. In order to avoid the construction of cycles during this operation, we active only one link each time. So the construction gives a Spanning Tree in ad hoc network.

The time taken is in polynomial time.

Distributed algorithm for maintain a Spanning Tree: [15] proposed a distributed algorithm for maintain a Spanning Tree when some mobiles move, arrive or leave the network. The algorithm takes in account adding/ deleting node/ link.

IV. CONCLUSIONS

Mobile Ad- hoc networks have been widely researched for many years. Wireless ad- hoc networks have attracted significant attention over the past few years. In this paper, we surveyed the state of the research that classified an approach to guarantee network connectivity, efficient routing and maintain network performance in MANET using spanning tree modes. This spanning tree may be considered as infrastructure for the communication in ad-hoc network.

V. REFERENCE

[1] B. Leiner, R. Ruth and A.R. Sastry, "Goals and challenges of the DARPA GIOMIO program". IEEE Personal Communications, vol. 3, no. 6, pp.34-43, December 1996.

[2] Luca Becchetti- Computer networks 11, A.A. 2009/206.

[3] Discrete Mathematics and its Applications with Combinatorics and Graph Theory, Kenneth H. Rosen.

[4] Introduction to graph theory, Gary Chartrand and Ping Zhang.

[5] Girmaldi, R. P. "Discrete and Combinatorial Mathematics, An Applied Introduction", Addison Wesley Longmen, Inc, (1999).

[6] Gallagher, R. G, Humblet, P. A., Sipra, P. M. A, Distributed Algorithm for Minimum- Weight Spanning Trees, ACM Transactions on Programming languages and systems (1983), pp.66-77.

[7] Awerbuch, B., Optimal Distributed Algorithms for Minimum- Weight Spanning tree, Counting, Leader Election and related problems, Proceedings of the 9th Annual ACM Symposium on Theory of Computing, (1987), pp. 230-240.

[8] Garay, J. A., Kutten S., Peleg D., Zhu Y. A Sub- linear time distributed algorithm for minimum weight spanning trees, Proceedings 34th Annual Symposium on Foundations of Computer Science, (1993), pp. 659-668.

[9] Lien., Y. N., "A New Node- Join- Tree Distributed Algorithm for Minimum Weight Spanning Trees", Proceedings of the 8th International Conference on Distributed Computing System, (1988), pp. 334-340.

[10] Banerjee, S., Khuller, S. , A Clustering scheme for hierarchical routing in wireless networks, Tech. Report CS-TR- 4103, University of Maryland, College park, (2000).

[11] Srivastava, R K.Gosh., Distributed algorithms for finding and maintain a k- tree core in dynamic network.

[12] Arun B. R, L.C. Reddy,… P. H., R. S., K-Nodes Multicasting Minimum Cost Spanning Trees in Wireless Mobile Ad Hoc Network (MANET), International Journal of Computer Applications (0975-8887), vol. 1.No.4.

[13] Yaacov . F, Dahlia. M., "K- Clustering in wireless ad hoc networks".

[14] P.Victer Paul, T. V, P.D, R. B., Modeling of Mobile ad hoc Networks using Distributed Spanning Tree Approach, P. Victer Paul et. al./ International Journal of Engineering Science and Technology vol. 2(6), 2010, 2241-2247.

[15] Hichem Megharbi, Hamamache. K., Distributed algorithms for constructing and maintain a Spanning Tree in a Mobile Ad hoc Network.