Fault Tolerance in Mobile Ad hoc Networks

Navroop Kaur[1], Neeraj Kumar[2]
Department of computer science & engg.[1,2],Thapar University, Patiala, India[1,2]
Email: navroopgoraya@ymail.com[1] , neeraj.kumar@thapar.edu[2]

Abstract- Mobile Ad hoc Networks provide flexibility and scalability which was not taken into consideration by the existing distributed systems. These networks are distributed networks and do not require any existing infrastructure. But in these types of networks there occurs some problems among which one may be occurring of fault. This may occur due to link failure, failure of nodes or network. We illustrate the solution for maintaining link connectivity even if the links to be used fails. A modified fault tolerance global spanning subgraph algorithm is determined which proves k-connectivity. This algorithm helps to form the minimum spanning tree considering the probability distribution function as edge weights in case the minimum weight edge to be added fails. This paper illustrates the way to toleratefault using modified global spanning subgraph algorithm and tested the performance of the proposed algorithm using NS-2 simulation.

Index Terms-Mobile Ad hoc Networks; Fault tolerance; K-connectivity; minimum spanning tree.

1. INTRODUCTION

Mobile Ad hoc Networks (MANETs) are a group of autonomous nodes or terminals which by forming multi-hop radio network exchanges information between each other and in a decentralized way maintain connectivity over wireless links. In MANETs nodes make a network temporarily without taking into consideration the centralized administration so these are a type of decentralized network. The network is called ad hoc as it is independent of pre-existing network infrastructure. They are not having any fixed infrastructure. Each node in MANETs is free to move in any direction independently, therefore can change its links to the other nodes frequently. As nodes are free to move the topology changes frequently so changes in the topology may not be easy to predict. This network configuration can be either static or dynamic. Its life is very limited and variable. MANETs uses flooding in contrast to classic routing for transmitting data.

The different network devices can be added easily in these types of networks. Here every user has a specific network address that is specified as a part of the network. It is an example of the network type in which a group of two or more devices have the capability of networking and exchanging information with each other. They generally help in anytime and anywhere computing. They are an auto configurable and self- organising type of network. Here nodes in the network have the capability of performing routing. Every host in the network works like a router. As the nodes are free to move they are having dynamic topology.

MANETs are different from the cellular networks in the sense that as in cellular networks there is pre-planned locations for establishing the base stations but

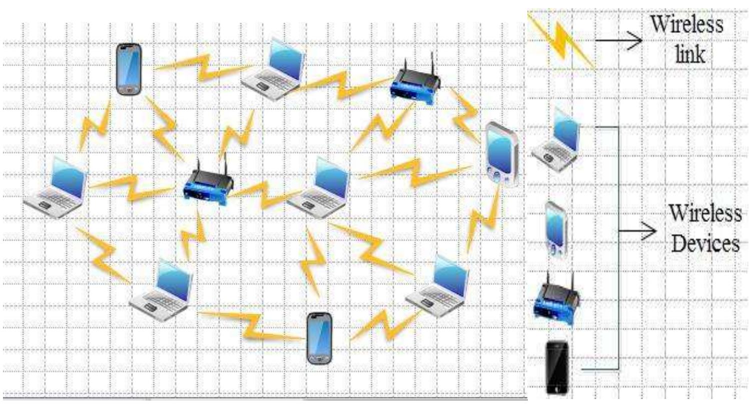

Fig. 1. Generalized framework of MANETs

in these the base stations are not employed and this type of network forms automatically and is alsoliable to change. Moreover in the cellular networks there is present a static backbone topology whereas in Ad hoc networks dynamic topologies are there which changes frequently in response to change of the transmission range of the various nodes present in the network to exchange information with the other nodes.

The path between nodes in the mobile ad hoc networks generally includes multiple hops so it is genuine to call this type of networks multi hop MANETs. In Fig. 1, an example of Mobile Ad hoc and communication topology is shown which consist of various computer devices such as PDA's, laptops and routers. Each node can exchange the information directly with every node that lies in its scope of transmission. To communicate with nodes that lie around its radius, the node relies on the intermediate nodes to transmit the message by jumping.

The advancement in the field of the wireless technology has posed many challenges under which comes the problem of fault tolerance that can occur in the network. The nodes in MANETs consume power for communicating with the other nodes present in the network. But if the nodes collectively determine such a topology in which the number of links should be less. Then the power consumed for transmitting the message from one node toother consequentlyreduces, as in turn the number of paths for routing the message in the network decreases.

In forming such a network topology like minimum spanning tree of the graph modelled as a network, k-connectivity is considered. The network is said to be k-vertex connected if on removing at most k-1 vertices or nodes even, the network remains connected. The links to form k-connected topology are chosen based on minimum weight but in case the link having minimum weight cannot be added because of failure of that link, then from that node choose the link having minimum probability distribution function (pdf).

The paper is organized further as follows: the related work is discussed in section II, section III summarizes the problem formulation, then the section IV provides insights into the performance analysis and finally in section V the report is concluded.

2. RELATED WORK

Penrose et al. [1] studied in random graph of n nodes the property of k-connectivity by inserting an edge at most r distance apart between each pair of nodes. He proved that the minimum value of r at which the graph is k-connected is equal to the minimum value of r at which the graph has minimum degree of k, with the probability 1 as n goes to infinity. The importance of this outcome is that it connects a global property of the graph i.e. k-connectivity to the node degree, a local parameter. So it's possible to construct localized algorithms which can preserve asymptotic k-connectivity. But in this work the minimum value of r to be taken is not given. Bettstetter et al. [2] also found the connection between k-connectivity and minimum node degree for random graphs. For the almost surely k-connected network analytical expression of the important range r_0 is calculated and afterwards verified by simulations.

Li et al. [3] further summarizes Penrose's work by giving the lower and upper bound on the minimum value of r at which value the graph is k-connected having high probability. They also proposed a localized topology control algorithm under the homogeneous network assumption that preserves k-connectivity. The proposed design is based on the Yao structure.

Hajiaghayiet al.[4] for finding the minimum power k-connected sub graph have given three approximation algorithms. Two global algorithms are based on predefined approaches. The first gives an *O* (kα) - approximation, here αis the best approximation factor for the k-UPVCS problem. The other advances the approximation factor to *O* (k) for general graphs. The third algorithm, Distributed k-UPVCS, is an algorithm which presents a $k^{O(c)}$ approximation, in which cis the exponent in the propagation model. For maintaining 2-connectivity, employingdistributed algorithm it firstly makes MST (minimum spanning tree) of the input graph and then afterwards arbitrary paths are inserted between each node and its neighbours in the resultant tree. So it can be concluded that it is not a fully localized algorithm as this distributedalgorithm is based on the distributed MST algorithm. It means that it depends upon information which is multiple hops away to make the minimum spanning tree. So this indicates involvement of large overhead for maintenance and delay when in response to node failure or mobilitythe topology needs to be changed. Moreover, it has been determined in inspecting the distributed algorithm that generally the node on the minimum spanning tree cannot be able to exchange information with the neighbouring nodes because of the limited transmission power. Due to this very basic reason, the arbitrary path that connects the neighbouring nodes in the algorithm may not exist in a network of low density.

Ramanathan et al. [5] has given two centralized algorithms, CONNECT and BICONN-AUGMENT, which decreases the maximum power used by each node in the network while preserving the bi-connectivity of the network. These two algorithms are the simple greedy algorithms that combine the various components in iterations until one of them remains. Calinescu et al. [6] also referred through the Min-Power Symmetric Biconnectivity problem. They proved that the Min-Power Symmetric 2-node problem is NP hard problem. But after that they presented a new algorithm which has a constant approximation ratio, MST-Augmentation, for 2-connectivity.

As the proposed modified fault tolerant global spanning subgraph is similar to the CONNECT and BICONN-AUGMENT in context to the process in which the topology is derived, like the way of combining the various components iteratively but in various concerns it is different from these two algorithms like the Modified fault tolerant global spanning subgraph preserves k-connectivity while the BICONN-AUGMENT only preserves 2-connectivity, so we can say that it is more general. Secondly there is no proof given that satisfy the correctness of the BICONN-AUGMENT but just mentioned in [5] while the whole considerations for proving the correctness of the MFGSS are detailed. Moreover the CONNECT and BICONN-AUGMENT works under the supposition of only homogeneous networks, whereas modified fault tolerant global spanning subgraph can be networks where maximal transmission power of every node present in the network is different.

Link and network failure in mobile ad hoc networks occurs because of the failed components present in the network as a result of the natural disaster or due to malfunction. Even when the node moves from the cluster node failure occurs.

Khazaei et al. [7] improves the data transfer and afault tolerance capability by forming backup path during route reply, route maintenance and while local recovery between source and destination. Ahmed et al. [8] proposed a fault tolerant routing protocol that examines two routes from source to destination. When link failure occurs on one path at the same time it swaps to the other path and fault gets tolerated.

Shaji et al. [9] [10] proposed a Self-eliminating Fault-tolerant based Un-interrupted reliable Service switching mobile Protocol. In this protocol generally proper nodes present in the network are found and reliable connections are formed between them in the heterogeneous surroundings.

Adeluyi et al. [11] present spiral millipede inspired routing protocol as a bio inspired routing approach. This protocol helps reducing the overhead faced during routing and also increases the fault tolerance capability in case of the occurrence of link failure.

3. PROBLEM FORMULATION

In this section the problem related to tolerating fault in the network is described. The fault in the network can occur due to failure of any link in the network. The approach defined is to create a minimum spanning tree of the main connected graph by choosing edges from it on basis of some other parameter rather than minimum weight in case the link having minimum weight fails. The minimum spanning tree to be formed must also be k- connected, i.e., the type of tree in which even on removal of k-1 vertices the graph remains connected. Here we have to choose the edge or link in the graph modeled as network on basis of the pdf rather than on basis of only minimum weight.

The criteria will be to obtain a graph which shows that as the value of one parameter increases correspondingly the other one decreases or vice-versa. The graph obtained describes the relation between pdf and the minimum weight value being specified for the various nodes. The concept is implemented further which describes that even if the edges of the minimum weight fails then the edges having weight more than them can be used based on the pdf of them which will be less to maintain network connectivity.

4. PERFORMANCE ANALYSIS

Dijkstra's and Kruskal's algorithm are two among important algorithms for graph [12]. These algorithms are used for finding path and forming minimum spanning trees from the connected graphs. There can be modified versions of these algorithms which can be used to find different variations of paths throughout a graph depending on what weight (value of an edge, or connectionbetween graph nodes) one is attempting to perform path optimization for. Since Kruskal's algorithm (and a variation of Dijkstra's algorithm) isthe backbone of the simulationcodewhich was used to form the stable broadcast topology, so firstly we will go into detail on Dijkstra's algorithm and Kruskal's algorithm and see how they work to better understand their role.

Dijkstra's algorithm is used for finding path in graphs. This algorithm is used to find the minimum weight path or the shortest path from the source vertex to every other vertex in the mobile graph [12]. The algorithm initiates from the source vertex and continuing in iterations gradually results the minimum-weight path to each and every vertex in the weighted graph. The algorithm assumes the weight of every vertex from the source vertex to be infinity and 0 of itself to calculate the shortest path. In each iteration, the algorithm examines if there exist a way to all the neighbour nodes with less weight than the estimated weight of the current path available for that specific vertex. And if there exist lesser weight, the weighted value of that specific vertex is updated.

Then from there the neighbour vertices are discovered to determine that whether it is possible to reach the other neighbouring nodes on the path with less weight than previously determined by another path. This whole process is basically determined over all the iterations.

Kruskal's Algorithm is a type of the greedy algorithm that helps to find the minimum spanning tree for the connected graph. This algorithm is having awareness about all the nodes present in the graph. Here every node is given a specific key rather than the weight. Kruskal's algorithm uses the knowledge of the edge weights to form the minimum spanning tree. In this, in each iteration the edge having the lesser weight is chosen and the nodes being connected by that particular edge form an edge of the minimum spanning tree. This process is gradually repeated during all iterations unless all nodes are connected and the minimum spanning tree is created. The edges having less weight are added gradually and it is being determined that adding an edge should not create the cycle. In this way the minimum spanning tree of the given connected graph is formed.

4.1. *Modified fault tolerant global spanning subgraph (MFGSS)*

A centralized MFGSS algorithm is considered which forms k-connected minimum spanning subgraph using minimum weight. The minimum spanning subgraph is created from the connected graph considering the calculated pdf between two nodes as the edge weights in case the link having minimum weight in the network fails due to some reason. Depending upon the pdf links or edges are chosen. Usually the edges having the minimum pdf are chosen. The pdf is applied based on the number of nodes and the minimum weight.The pdfis calculated using Eq. (1), as in

$$pdf(i) = e^{-z*i} \qquad (1)$$

Here pdf is the probability distribution function,
z is the minimum weight to be entered,
i is the number of nodes to be entered.

From Eq. (1), it is determined that the value of pdf is inversely proportional to the value of z. So the minimum pdf decreases as the value of minimum weight increases. So the links having minimum pdf are chosen to form edges of the minimum spanning subgraph.

This distribution is used basically to represent the lifetime of some kind of equipment, or to represent the interarrival time between successive events when the events occur purely randomly (like demands for computer service, telephone calls at an exchange, etc.).

MGFSS is somehow related to Kruskal's algorithm, as Kruskal's algorithm [13] forms the minimum spanning subgraph which is 1-connected spanning subgraph but this algorithm creates spanning subgraph for value of k greater than 1 and edges are selected based on the pdf. Generally edges having the less pdf are selected to form an edge in minimum spanning subgraph.

The various ways to prove the correctness of MFGSS is different from that of the Kruskal's algorithm even though it is generalized form of it. The correctness of MFGSS is proved further but before that in its context two lemmas are given, these are required for the proofs to assumed propositions for proving the correctness of the proof.

Lemma 4.1. *Let x_1 and x_2 be the two vertices in a k-connected graph G. If x_1 and x_2 are k-connected after the removal of the edge (x_1, x_2), then G - (x_1, x_2) is still k-connected [14].*

Lemma 4.2. *Let G and G' be the two undirected graphs having same number of vertices. The G' will be k-connected since E (G') is subset of E (G), G is also k-connected [14].*

Theorem 4.3. *MFGSS can preserve k-connectivity of G, that is G' is k-connected if G is k-connected [14].*

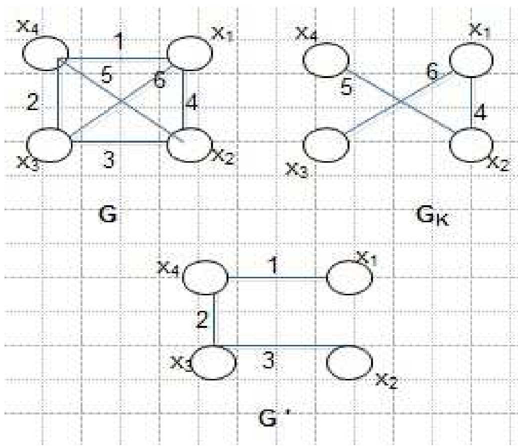

Fig. 2. Example proving k-connectivity of graph G'

Proof. As the edges are added into G' in ascending order, whether x is k-connected to y at the moment before (x, y) is inserted is dependent on the edges having less weight. Therefore every edge (x, y) ε E_0 = E(G) – E(G') satisfies that x is k-connected to y in G-{(x,y) ε E(G): w(x,y) >= w(x_0, y_0)}. So in this way it can be proved that G' preserves k-connectivity of G by applying lemma 2 to G' [14].

The proof can be explained with the help of diagrammatical representation given in Fig. 2.

To provide the proof for k- connectivity two lemmas are considered in [14] which describes that in the graph G even if removing one edge (x_1, x_2) from the graph, x_1 and x_2 remains k-connected then the graph after removal of this edge is also k-connected. The second one consider that every edge (x_1, x_2) in the graph $G_k$satisfies that x_1 is connected to x_2 and as it is having edges of weight greater than the edges present in the graph G', then G' is also k-connected . It can be seen thatgraph G_k is k-connected as removing any vertex at a time from graph does not disconnects the graph. The graph G is k- connected main graph and G' is the minimum spanning subgraph of graph G and G_k is the graph formed having edges of G that not present in G'.

In the proof given in [14] it is being proved that graph G' is k-connected as links are added in ascending order of weight. As it is seen that the graph $G_k$is having edges of weight large than the edges of the graph G_k and is k-connected based on which it is proved that G' is also k-connected. So according to the problem the minimum spanning subgraph of the graph is formed taking into account the edges having minimum pdf. Like in case if we consider that in G' link having minimum weight 3 fails, then we can choose the edge having weight 4 to form the minimum spanning tree as that link must be having minimum pdf. So the criteria will be to form tree based on pdf and consider that it is also k-connected on basis of proofs given in [14].

4.2. *Algorithm*

MFGSS Algorithm

Procedure: MFGSS
Input: G (V, E), a simple k-connected graph
Output: G' (V', E'), a k-connected spanning subgraph of the graph G
Begin

```
V' = V, E' = ϕ
for i to n
   z = minimum weight
     display the results according to equation (1)
arrange edges in ascending order of weight
for every edge (x1, x2) in the order
   if x1 is not k-connected to x2 in G'
      then add {(x1, x2)} to E'
            else if in the same connected component if
all nodes are present then
      exit
   endif
end
```
End

The MFGSS algorithm creates the minimum spanning tree from a connected graph and uses minimum probability distribution function as the edge weight.

The pdf is measured by entering the value of minimum weight for various number of nodes entered. The average of simulation results yielded shows that as the value of minimum weight increase the value of pdf decreases correspondingly. In Fig. 3, overall simulation result for minimum weight 0.2 is

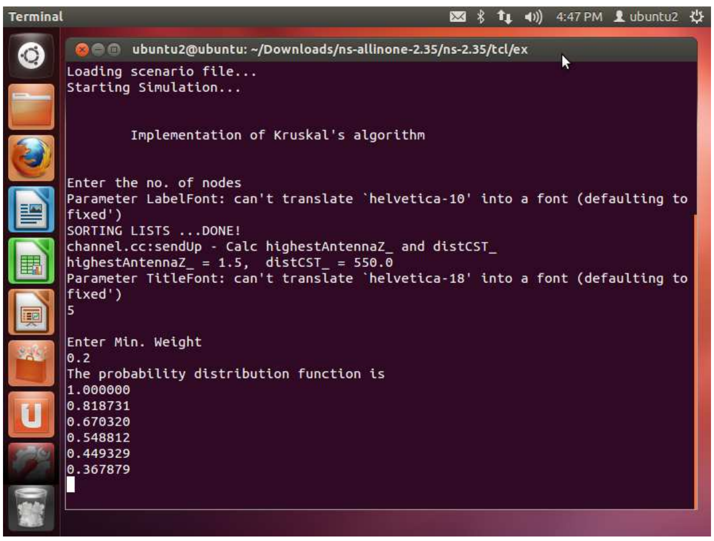

Fig. 3. Simulation results for minimum weight 0.2

shown and in Fig. 4, result for the minimum weight 0.8 is shown for 5 nodes entered.

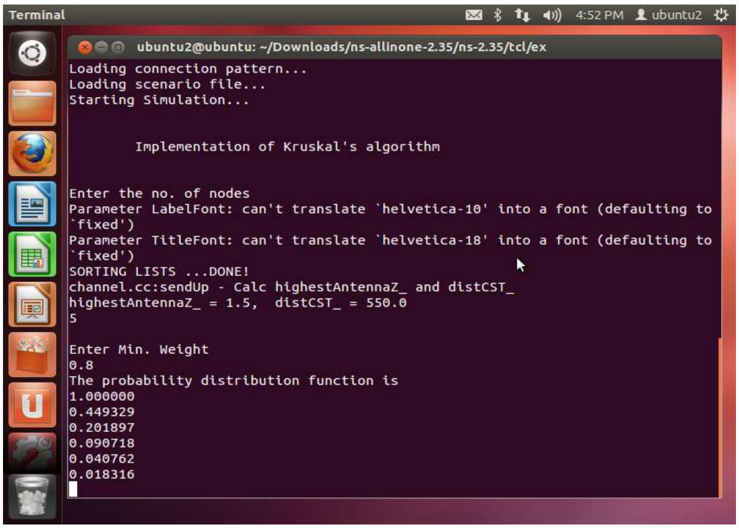

Fig. 4. Simulation results for minimum weight 0.8

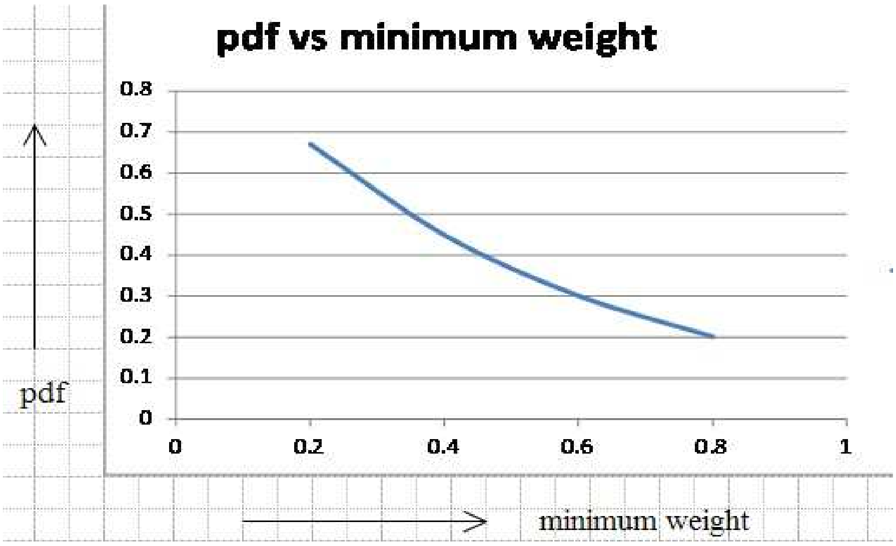

Fig. 5. Graph showing pdf values w.r.t minimum weight for node 2

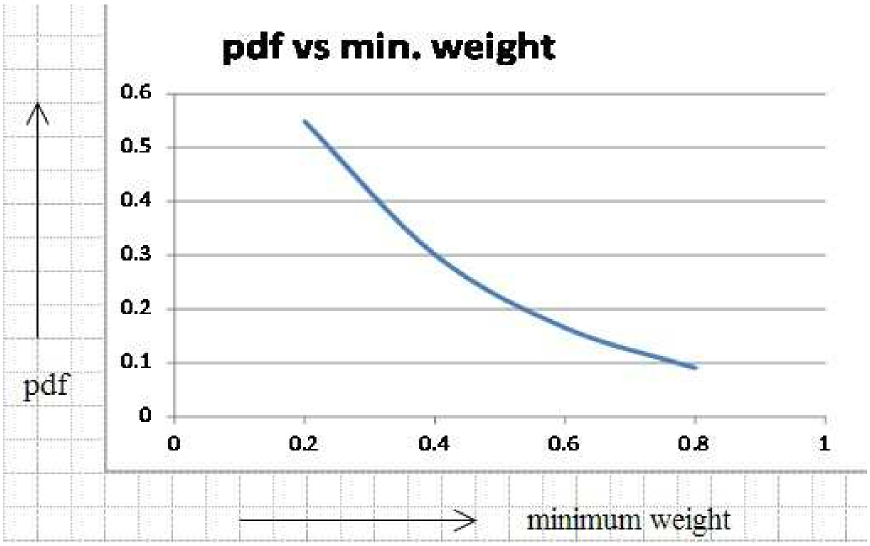

Fig. 6. Graph showing pdf values w.r.t minimum weight for node 3

The graph in Fig. 5, shows the pdf values against the minimum weights defined for the node 2 and in Fig. 6 for node 3.

5. CONCLUSION AND FUTURE SCOPE

We consider the centralized fault tolerant global spanning subgraph algorithm which can form the k-connected minimum spanning subgraph to maintain to maintain the link connectivity. The conclusion drawn is that link connectivity can be maintained by choosing the link having minimum pdf in case the link to be added having minimum weight fails to form the k-connected subgraph. The edges are chosen then to form subgraph based on edges having minimum values of pdf. The future considerations will be to provide better network performance and to remove the various other problems related to maintaining connectivity in the network by using much of the information related to the network.

References

[1] J. M.D. Penrose, "On K-Connecitivity for a Geometric Random Graph.", Random Structures and Algorithms, 1999.

[2] C. Bettstetter, "On the Minimum Node Degree and Connectivity of a Wireless Multihop Network.", ACM Symp. Mobile Ad Hoc Networking and Computing (MOBIHOC), 2002.

[3] X.Y. Li, P.J. Wan, Y. Wang, and C.W. Yi, "Fault Tolerant Deployment and Topology Control in Wireless Networks.", ACM Symp. Mobile Ad Hoc Networking and Computing, 2003.

[4] L. Li, J.Y. Halpern, P. Bahl, Y.-M. Wang, and R. Wattenhofer, "Analysis of a Cone-Based Distributed Topology Control Algorithm for Wireless Multi-Hop Networks.", ACM Symp. Principles of Distributed Computing, 2001.

[5] R. Ramanathan and R. Rosales-Hain, "Topology Control of Multihop Wireless Networks Using Transmit Power Adjustment.", 2000.

[6] G. Calinescu and P.J. Wan, "Range Assignment for High Connectivity in Wireless Ad Hoc Networks.", International Conf. Ad-Hoc, Mobile, and Wireless Networks (ADHOC-NOW), 2003.

[7] M. Khazaei and R. Berangi, "A multi-path routing protocol with fault tolerance in mobile ad hoc networks." , International CSI Computer Conference, 2009.

[8] R.E. Ahmed, "A Fault-Tolerant Routing Protocol for Mobile Ad Hoc Networks.", Journal of Advances in Information Technology, 2011.

[9] R.S. Shaji, Dr.R.S. Rajesh and B. Ramakrishnan, "A Fault-tolerant Scheme for Routing Path Re-establishment for reliable communication in Heterogeneous Networks.",

[10] International Journal of Scientific & Engineering Research, 2010.

[11] R.S. Shaji, Dr.R.S. Rajesh and B. Ramakrishnan, "SFUSP: A Fault-tolerant Routing Scheme for path establishment among Mobile Devices in Pervasive Spaces.", Journal of Computing, 2010.

[12] Adeluyi, O.J. Lee, "SMiRA: A bio-inspired fault tolerant routing algorithm for MANETs.", International Conference on ICT convergence (ICTC), 2012.

[13] Meghanathan, Natarajan, "Applications of Graph Theory Algorithms in Mobile Ad hoc Networks", 2008.

[14] J.B. Kruskal, "On the Shortest Spanning Subtree of a Graph and the Travelling Salesman Problem." ,1956.

[15] Li, Ning, and C. Jennifer, "Localized fault-tolerant topology control in wireless ad hoc networks.", IEEE Transactions on Parallel and Distributed Systems, 2006.