Distributed Fault-Tolerant Topology Control in Static and Mobile Wireless Sensor Networks

Indranil Saha, Lokesh Kumar Sambasivan, Ranjeet Kumar Patro, Subhas Kumar Ghosh
Honeywell Technology Solutions Lab Pvt. Ltd.
151/1, Doraisanipalya, Bannerghatta Road, Bangalore 560 076, India
Email: {indranil.saha, lokesh.sambasivan, ranjeet.patro, subhas.kumar}@honeywell.com

***Abstract*— In wireless sensor networks, minimizing power consumption and at the same time maintaining desired properties in the network topology is of prime importance. In this work, we present a distributed algorithm for assigning minimum possible power to all the nodes in the wireless sensor network, such that the network is K-connected. In this algorithm, a node collects the location and maximum power information from all the nodes in its vicinity, and then it adjusts the powers of the nodes in its vicinity in such a way that it can reach all the nodes in the vicinity through K optimal vertex-disjoint paths. We prove that, if each node maintains K optimal vertex-disjoint paths to all the nodes in its vicinity then the resulting topology is globally K-connected, provided the topology obtained when all nodes transmit with their maximum power G_{max} is K-connected. This topology control algorithm has been extended to mobile scenario and the proof of connectivity in the mobile scenario has been presented. Simulation results show that significant power saving can be achieved by using this algorithm.**

I. Introduction

A Wireless Sensor Network (WSN) is composed of a large number of sensor nodes deployed arbitrarily in a two (or three) dimensional region. A limited-power battery fulfills the power requirement of a node. When this battery is completely discharged the node is no longer capable of transmitting or receiving any signal. So power is a valuable resource for sensor nodes. It is desirable that the nodes transmit with minimum possible power, so that the lifetime of the WSN is prolonged.

The main goal of the topology control is to assign power to all the nodes in the network, so that a few desired properties are maintained globally. One-Connectivity or simply Connectivity has been widely considered to be the required property that should be maintained in the WSN [10], [11], [12], [7]. Attempts have been made to assign minimum power to all the nodes in the network to ensure the global connectivity of the network. As low cost sensor nodes powered by limited-power battery build the sensor network, there is a high probability of a node failure. If between any two nodes there is only one path, then the absence of any one node in that path implies that there is no other way of communication between that pair of nodes. So, it is desired that there be more than one vertex-disjoint path between any pair of nodes, i.e., the more general problem of K-Connectivity be attended.

Minimum power assignment problem can be of two types: Homogeneous power assignment and heterogeneous power assignment. In homogeneous power assignment all nodes in the network are assigned the same power to maintain the connectivity (or K- connectivity) of the network. In heterogeneous power assignment problem, nodes present in the network are assigned minimum possible power so that the desired connectivity is maintained. Heterogeneous power assignment algorithms offer more energy-efficient network than the homogeneous power assignment algorithms. In this paper, we consider heterogeneous power assignment algorithm in wireless ad hoc networks.

A topology control algorithm should be fully distributed and asynchronous, and rely on local information only. Another important consideration of the topology control algorithm is the symmetry of the communication graph. As every node maintains some desired link to other nodes in the network, it is natural that resultant topology is not symmetric. Technical feasibility of implementation of wireless unidirectional link was supported by Pearlman et. al. [14], Bao and Garcia-Luna-Aceves [1], Kim et. al. [6], Prakash [15], and Ramasubramanian et. al. [16]. But, Marina and Das [13] showed that according to the performance, symmetric network topology is superior to the asymmetric one. However, the capability of forming a topology that consists of only bidirectional links is important for link level acknowledgments and packet transmissions/retransmissions over the unreliable wireless medium. Bidirectional links are also important for floor acquisition mechanisms such as RTS/CTS in IEEE 802.11. So it is desirable that the topology is composed only of the bi-directional links.

Another important aspect of the topology of the sensor network is the average node degree. The node degree is defined as the number of nodes within the transmission radius of a node. Average node degree is an good indication of the level of MAC interference, and better spatial reuse. The smaller the degree of a node, the less number of nodes its transmission may interfere with.

This paper has been organized as follows. In section II we recall the works that have been carried out in this field. In section III we describe the system model and the assumptions that we have considered to design the distributed algorithm, and formally define the problem. In section IV we present the proposed distributed algorithm and prove the connectivity result for the static sensor network. In section V we describe how the distributed algorithm can be easily extended to handle mobile scenarios and present the connectivity results in mobile scenarios. In section VI we present the simulation results to

show the performance of the proposed algorithm and compare it to the existing best algorithm for this problem. In section VII we finally conclude the paper.

II. Background

The problem of maintaining K-connectivity in the network assigning approximately minimum possible power to all the nodes has been attended in a few previous works. Bahramgiri et. al. [2] used the cone-based topology control (CBTC) algorithm [10] to get K-connectivity in the global network. As the CBTC algorithm this algorithm also deals with homogeneous network which may not be the case in all practical purposes [9], [18]. A hybrid topology control framework, Cluster-based Topology Control (CLTC) algorithm for getting K-connected network has been proposed by Shen et. al. [20]. Their algorithm is not a fully distributed one. Chen and Son [4] present a fault-tolerant topology control by adding necessary redundant nodes to the network's simple communication backbone with a distributed algorithm. But it may not always possible to add redundant nodes to the existing sensor network. Li and Hou [8] presented the fault-tolerant topology control algorithm in which all nodes compute the spanning subgraph locally, where an edge is added to the local spanning subgraph if the two endpoints of the edge are not K-connected, and it has been proved that the global network is K-connected. Their algorithm considers heterogeneous power assignment, and the final topology contains only bi-directional links. The algorithm in [8] out-performs the algorithm presented by Bahramgiri et. al. [2].

Li et. al. [10] showed how cone-based algorithm can be adapted in network reconfiguration and mobile scenario. It is shown that if the topology ever achieves stability and the reconfiguration algorithm is executed, then network connectivity is maintained. Bahramgiri et. al. [2] adapted the same reconfiguration algorithm to preserve K-connectivity in case of network reconfiguration and mobile scenario. In [18], it is argued that mobility resilient topology control protocol should require little maintenance in the presence of mobility. In [18], Topology control protocols are classified into two types: P1 and P2. Protocol P1 builds the topology in the distributed manner and sets the nodes transmission power accordingly. In protocol P2 every node tries to maintain some number of neighbors in its vicinity according to some criteria. The algorithm presented by Li and Hou [8] is an example of protocol P1, whereas the algorithm presented by Bahramgiri et. al. [2] is an example of protocol P2. The reconfiguration procedure for protocol P1 is more complicated than that for protocol P2. Maintaining the MST in mobile scenario demands the algorithm to run frequently, as the absence of one edge from the topology graph may make the topology disconnected. On the other hand maintaining a number of neighbors at a particular cone as done in [2] is easier than protocol P1. Though the algorithm presented by Li and Hou [8] is very efficient for static network, but it is not advantageous in mobile scenario.

In this work, we have proposed a novel distributed algorithm for topology control in static sensor networks, that can be easily extended to mobile scenario. We have compared our work to that of Bahramgiri et. al. [2] and our algorithm outperforms the algorithm presented in [2] both in terms of average assigned power to the nodes, and average degree of the nodes.

III. System Model and Problem Definition

In this section, we describe our system model. In this model, each sensor node is equipped with an omnidirectional antenna. The transmitting power for a sensor node can be adjusted to a desired value. In ideal case, if a node transmits with power r^2 then all nodes in the sphere of radius r, with the node at center, can receive the transmission. However depending upon different kind of noise present in the transmission medium, the transmission power required for a node to reach up to a distance r is proportional to r^α, where $2 <= \alpha <= 5$, α is called *power attenuation exponent* [5].

In our system model, we assume that every node i knows its location (x_i, y_i). P_i^{max} is the maximum power available at node i at a given instant of time. The nodes present in the network may have different maximum powers. P_{ij} is the power needed to reach from node i to node j. If the Euclidian distance between node i and node j is r_{ij}, then $P_{ij} = r_{ij}^\alpha$. It is assumed that the transmission medium is symmetric, in that case, $P_{ij} = P_{ji}$. If for two nodes i and j, $P_i^{max} >= P_{ij}$ and $P_j^{max} >= P_{ij}$ then we consider that there is an edge between node i and node j, and we denote the edge by $\{i, j\}$. For any i, j, if $P_i^{max} \neq P_j^{max}$ and $P_i^{max} >= P_{ij} > P_j^{max}$, then there will be an arc from i to j, but no arc from j to i. So there will be an asymmetric link between i to j, which is, say, denoted by the directed link $\overrightarrow{L_{ij}}$. When that is the case, we consider that no edge is present between node i and node j. In this way, the topology when all nodes transmit with their maximum transmission power is an undirected graph composed of only bi-directional edges. We call this graph the *maximum topology*. Let it be denoted by $G_{max} = (V, E)$, where V is the set of nodes in the network and E denotes the set of all edges when all nodes are transmitting with their maximum power. The objective of the distributed topology control algorithm is to get *minimum power topology* G_k^* which is strongly K-connected, provided G_{max} is strongly K-connected.

Hajighayi et. al. [5] introduced the notion of power cost and normal cost of a topology graph. For an undirected graph $G = (V, E)$ with edge cost p_{ij}, the power cost of G is defined as

$$P(G) = \sum_{i \epsilon V,\ j|(i,j)\epsilon E} max\ p_{ij} \tag{1}$$

For a graph $G = (V, E)$ with edge costs p_{ij}, the normal cost of G is defined as

$$C(G) = \sum_{(i,j)\epsilon E} p_{ij} \tag{2}$$

Taking these two as the functions to be optimized, two different optimization problems have been defined. These problems

are called Undirected Minimum Power K-vertex connected Subgraph (K-UPVCS) problem, and Undirected Minimum Cost K-Vertex Connected Subgraph (K-UCVCS) problem.

Though we are dealing with heterogeneous networks we can use any of these two functions to optimize the power. In [21] Wieselthier et. al. introduced the concept of *Wireless Multicast Advantage* (WMA) and applied the energy saving potential of WMA to the minimum energy broadcast and multicast problems. Srinivas et. al. [19] showed that with WMA, the energy cost function becomes a function of a node-based metric, where it is enough to consider the power cost of the topology as the optimization function. In this paper, we consider the K-UPVCS problem. We define our problem as follows:

Assign minimum possible power to all the nodes (not necessarily equal) so that if all nodes transmit with their assigned power then the network will be globally K node-connected. Mathematically,

Objective : Minimize $\sum_{i \epsilon V} P_i$, where P_i is the assigned power to node i.

Subject to The graph G_k^* being K-connected.

IV. Distributed topology control algorithm

The algorithm presented here is a distributed algorithm that every node runs depending on its locally accumulated data. When all nodes finish running the algorithm, they are assigned with approximately minimum power and the resulting network topology becomes globally K connected. The algorithm runs in three phases. At any generic node i the algorithm is as follows:

Phase 1: **Information collection and Finding the vicinity topology**

Node i broadcasts a *Hello* message using its maximum transmission power P_i^{max}. The set of nodes that receive the *Hello* message and node i itself is referred to as the *vicinity nodes* of node i, denoted as V_i. *Hello* message includes the id of the transmitting node, its location and maximum power. The format of the *Hello* message from node i is as follows:

$$< \textbf{Hello}, i, (x_i, y_i), P_i^{max} >$$

Upon receiving such a *Hello* message, each node j in V_i replies to node i with an *Reply* message, with its location (x_j, y_j) and P_j^{max}. The format of the *Reply* message from node j to node i is as follows:

$$< \textbf{Reply}, j, i, (x_j, y_j), P_j^{max} >$$

If any node j in V_i has maximum power less than the power required to send a message to node i, i.e., $P_{ji} > P_j^{max}$, then, j must find a multi-hop path to reach i. In this case, its neighboring nodes help it by forwarding the *Reply* message.

After sending the *Hello* message a node waits for a predefined amount of time to get the reply messages. When that time is over node i computes its vicinity topology according to the gathered information. After getting the *Reply* messages from all the nodes in its vicinity, node i knows the location and maximum power of all the nodes in its vicinity. Having the knowledge of the locations and maximum transmission powers for itself and all its vicinity nodes, node i can derive the existence of the vicinity edges, and thus the vicinity graph. For any two nodes $j,\ k\ \in\ V_i$, {j, k} is defined as one of i's vicinity edges, if $P_j^{max} >= P_{jk}$ and also $P_k^{max} >= P_{jk}$. Consequently, node i constructs its local vicinity topology that includes all its vicinity nodes, itself and the discovered vicinity edges. If node i's vicinity topology is denoted as G_i and the collection of its vicinity edges is denoted as E_i, then we obtain a weighted, undirected graph $G_i = (V_i, E_i)$, where the weight of each edge, $w(i, j)$, is the power required to reach j from i on the edge $\{i, j\}$, equivalent to P_{ij}.

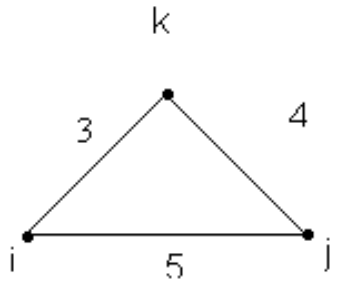

Fig. 1. Illustrating the suboptimal power assignment obtained by the algorithm in [12] for K=1.

Phase 2: **Construction of the minimum-power vicinity topology**

Node i finds out K vertex disjoint paths to all the nodes in V_i according to some optimality criteria. The optimality criteria should be such that the power assigned to the nodes is minimized. In [12] the path cost, i.e., the sum of the weights of the edges on the path has been considered to be the optimality criteria to choose a path between two nodes, and shortest path algorithm was used to find out the best path in between two paths. But in [17] it was shown that only considering the path cost may produce sub-optimal result, and in support the following example was presented. Consider a situation in which there are three nodes forming a triangle as shown in the Fig. 1. By running shortest path algorithm the nodes i, j and k will be assigned power 5, 5, 4, where in fact each node could have chosen 3, 4, 4 units of power respectively to maintain the reachability between each other.

In order to alleviate the effects like above, the following three metrics have been considered to choose optimal vertex disjoint paths from a node to the other nodes in its vicinity.

- The total cost of the path (C)

- Maximum edge cost in the path (X)
- Number of hops (N)

In [17] a function has been defined incorporating all these three parameters as follows:

$$F = C^c.X^x.N^n \tag{3}$$

A rigorous experiment has been carried out to find out the values of the exponents c, x and n in this function. In that work, the topology with asymmetric link was considered, and Equ (2) has been used as the function to be optimized. Here we have carried out similar kind of experiment for the optimization function given by Equ (1) and the topology with only symmetric links. We have found out that considering only the maximum edge cost in the path is giving the best result in terms of power assignment to the nodes. As power assigned to a node is equal to the maximum weight outgoing edge, it is logical to consider the maximum edge weight when we choose a path between two nodes.

To get the optimal path by using the maximum edge cost of the path, we have modified the dijkstra's algorithm [3] and we call it *Get_Optimal_Path*. This algorithm finds out the path with minimum value of maximum edge weight from source node s to destination node j. Every node v in node s's vicinity topology G_s maintains an attribute X which holds the value of the maximum edge weight on the path from s to itself, and the path has minimum value of maximum edge weight among all the paths from s to v. Initially all the vertices are assigned infinity cost through X. The predecessors set of v is set to NIL through p. The X value of source node s is assigned zero. The structure Q contains all the vertices of G_s. The function $Extract\_Min$ removes the vertex with minimum X value from Q. The usual relaxation procedure used in Dijkstra's algorithm, i.e., $d[v] \leftarrow d[u] + w(u, v)$ (for nodes u and v, u is chosen by the $Extract\_Min$ and v is the adjacent node of u; in Dijkstra's algorithm d[v] is the shortest-path estimate of node v) is replaced by $max(X[u], w(u, v))$; in so doing we obtain the path in which the maximum edge cost is minimum. The *Get_Optimal_Path* algorithm is formally presented in Algorithm IV.1.

Algorithm $Get\_Optimal\_Path(G_s, s, d)$

forall *vertex* $w \epsilon V_s$ **do**
  $X[w] = \infty$
  $p[w] = NIL$
end
$X[s] = 0$
$Q = V[G]$
$S = NULL$
u = Extract_Min(Q)
while *(* $u \neq d$ || $Q \neq NULL$*)* **do**
  $S = S \bigcup u$
  forall *vertex* $v \epsilon Adj[u]$ **do**
    if *(*$X[v] = \infty$*)* **then**
      $X[v] = w(u, v)$
    else
      if $(X[u] < w(u, v))$ **then**
        $X[v] = w(u, v)$
      else
        $X[v] = X[u]$
        $p[v] = u$
      end
    end
  end
  u = Extract_Min(Q)
end
if *(* $u = d$*)* **then**
  Return (Path(s, d))
else
  Return $NULL$.
end

Algorithm IV.1: Algorithm for finding optimal path between node i and node j

Theorem 1: Algorithm *Get_Optimal_Path* run on a graph with source s and destination d returns the path with minimum value of maximum edge weight if exists, and returns NULL if no path exists between s and d.

Proof: We have to show that when d is obtained by the *Extract_Min* function, then the path obtained from s to d is the path with minimum value of maximum edge weight among all the paths from s to d. Let us show it by the method of contradiction. Let us consider that when the *Extract_Min* function returns the node d, the path returned by the algorithm is not the path with minimum value of the maximum edge weight. So, there is a better path in terms of the minimum value of maximum edge weight than the path returned by the algorithm. Let y is the vertex on the better path, which is not in the set S and directly connected to one of the node in the set S. As y is on the path whose maximum edge weight is less than the maximum edge weight of the returned path from s to d, so $X[y] < X[d]$. But as y is not in the set S, and d has been returned by the *Extract_Min* function, so $X[d] < X[y]$. Thus, the two inequalities are in fact equalities, giving $X[y] = X[d]$. It implies that our assumption that the returned path is not the best path in terms of minimum value of maximum edge weight is wrong. So, we can conclude that the algorithm *Get_Optimal_Path* run on a graph with source s and destination d returns the path with minimum value of maximum edge weight. ■

By using the vicinity graph, a node finds out the best path from it to a node in the vicinity. For any node j in the vicinity, node i first use the *Get_Optimal_Path* algorithm to select the path whose maximum edge cost is minimum among all the paths and store the path in an appropriate data-structure. Then that path is destroyed and next best path is considered. In this way, K vertex disjoint paths are obtained between the node i and its neighboring node.

Phase 3: **Transmission Power Assignment**

In this phase, node i needs to calculate the transmission power needed for itself and all nodes in V_i, to ensure that all

its minimum-power paths exist in the final minimum power network topology. Specifically, for node i itself and each node in set V_i, the transmission power is assigned as the power required to reach the furthest one-hop downstream nodes in node i's minimum-power vicinity graph G_{im}. Node i first assigns its own power, and then sends the minimum power required for other vicinity nodes with an explicit *Assigned Power*(AP) message. The format of the AP message from node i to node j is as follows:

$$< \textbf{assigned\_power}, j, i, power\_required_{ji} >$$

While assigning power to the nodes in the vicinity, a node should take care of the unidirectional links. It may be possible that in the minimum-power vicinity graph G_{im} of node i, directional link $\overrightarrow{L_{ij}}$ is present, but link $\overrightarrow{L_{ji}}$ is not. But, when node i assigns power to the nodes in its vicinity, it has to assign power to j such that link $\overrightarrow{L_{ji}}$ also exists. As maximum topology is an undirected graph, it is guaranteed that if $\overrightarrow{L_{ij}}$ exists, then power can be assigned to node j such that $\overrightarrow{L_{ji}}$ would also exist. In this way node i maintains all the links in its vicinity to be bi-directional. Upon receiving the AP message, a vicinity node j compares the power requirement from i with its current power setting. If i requires a stronger transmission power at node j, node j increases its power accordingly. Otherwise, it discards the AP message. Note that its existing setting is assigned by itself or any other nodes that have executed the algorithm earlier than node i and propagated the AP message.

Now we shall present two theorems to prove the desired connectivity of the network. Theorem 1 shows that if the maximum topology G_{max} is connected then our algorithm for $K = 1$ gives a connected minimal topology G^* (say). We prove this theorem in the same line as done in [12] to prove the connectivity result. By using Theorem 2, Theorem 3 shows that for any K, if all nodes run the above algorithm individually, the resulting topology G_k^* (say) will also be K-Connected, provided G_{max} is K-Connected.

Theorem 2: The Algorithm *K-connected_Minimal_Topology* gives the optimal connected topology G^* for $k = 1$, provided the graph obtained when all nodes transmit with their maximum power G_{max} is Connected.

Proof: Let us consider two generic nodes u and v in the network. There may be two cases: (a) node v is in the vicinity of node u and (b) node v is not in the vicinity of node u.

Case (a): When node u constructs its minimum-power vicinity graph G_{um}, it finds out the optimal path from itself to all the nodes in V_u. As node v is in the vicinity of node u, obviously there exists a path from node u to node v.

Case (b): As the maximum topology G_{max} is connected, so there exists a path from node u to node v. Let us consider such a path where in between node u and node v, node y_1, y_2, . . ., y_n are present. So, obviously $y_1 \epsilon V_u$, $y_2 \epsilon V_{y_1}$ and so on. According to the logic of case(a), in G^*, u is connected to y_1, y_1 is connected to y_2 ,. . . , y_n is connected to v. So u and v are connected in G^*. The above logic proves that if G_{max} is connected then G^* is also connected. ■

Theorem 3: If there are K optimal vertex-disjoint paths from each node to all the nodes in its vicinity, then between any two nodes in the global network there exists K vertex disjoint paths i.e., the resulting topology G_k^* is globally K-Connected, provided the graph obtained when all nodes transmit with their maximum power G_{max} is K-Connected.

Proof: We shall prove the theorem by the method of proof by contradiction. Let us suppose that G_k^* is not K-connected. So there exists at least one set of $K-1$ nodes, by removing which we can get a graph that is not connected. Let's denote this graph by G''. Let G' be the graph obtained by removing the same set of $K-1$ nodes from G_{max}, which were removed in forming G'' from G_k^*. As G_{max} is K-connected, so G' is connected. Let G^* be the graph obtained by running the proposed algorithm with $K = 1$ on the remaining set of nodes, i.e., the set obtained after removing $K-1$ nodes. According to Theorem 2, G^* is connected because the graph G' is connected.

As G^* is connected and G'' is not connected, at least one edge of G^* will not be present in G'' (Note that G_k^* and G^* are constructed in the same manner). Let us suppose that the edge $\{u, v\}$ is one of such edges in G^*, which is not present in G''. The presence of the edge $\{u, v\}$ in G^* implies that {u,v} is the optimum path from u to v. So if $K-1$ vertices were not removed from the graph G_k^*, then the edge $\{u, v\}$ would be at least the K-th optimal path from u to v in G_k^*. So the edge $\{u, v\}$ is one of the K vertex disjoint optimal paths from u to v in G_k^*. By removing the set of $K-1$ nodes from G_k^* we can destroy at most $K-1$ vertex disjoint paths. But the direct edge $\{u, v\}$ will still be present, since it is one of the K optimal vertex disjoint paths from u to v and also removal of a set of $K-1$ nodes can not destroy the $\{u, v\}$. (Note that u and v are nodes selected from remaining set, so they would not have been removed.)

So the edge $\{u, v\}$ will be present in G''. So our assumption that the edge $\{u, v\}$ is not present in G'' is incorrect. This implies that all edges present in G^* are also present in G''. So G'' is connected. Thus G_k^* is K-Connected.

So the network is globally K-Connected. ■

V. Dealing with Mobility

In wireless Ad hoc network the structure of the network changes time to time. A node may be added to the network, a node may die due to the lack of the power or, due to mobility a node may change its position. To deal with these situations we present the following algorithm by using the Neighbor Discovery Protocol presented in [10]. We call this algorithm *K-connected Mobile Resilient Topology Control* (K-MRTC). In this protocol, three basic events have been defined: $join_u(v)$, $leave_u(v)$ and $change_u(v)$. In the $join_u(v)$ event, node v which was not in the vicinity of node u previously appears in the vicinity of node u. In the $leave_u(v)$ event, node v which was previously a neighbor of node u, disappears from the vicinity of node u. We only consider the join and the

leave events, for in case of a change event, after changing the position when a node comes to the stable state, it is equivalent to some leave events followed by some join events. We assume that when a node is mobile the network may not be K-connected temporarily, but when the network comes to a stable condition, network K-connectivity is preserved.

Our algorithm works as follows. Any node i, which is in the stable condition, broadcasts a *Beacon* message with its maximum power periodically. This *Beacon* message is for all the nodes j, where i is in the vicinity of node j, $i \epsilon V_j$. As we are considering heterogeneous network, it may be possible that i is not able to reach all nodes j for which $i \epsilon V_j$, even though node i use the maximum power to transmit the *Beacon* message. In this case, its neighboring nodes help it by forwarding the *Beacon* message. If a node does not receive the *Beacon* message from one of its neighbor within a time interval T, then it assumes that the node is no more its neighbor.

To handle join event:

When a node i is added to the network for the first time, or the node becomes stable after mobile condition, it broadcasts the *Hello* message with maximum transmitting power P_i^{max}, and build its vicinity graph G_i in the same way it is done in the first phase of the algorithm in the static case. All the nodes send *Hello* message periodically. The nodes which were already in the vicinity of a node don't reply to this *Hello* message. The new node replies the Hello messages with *Reply* messages. When a node gets reply from a new node, it finds out K-optimal vertex disjoint paths to the new node, keeping the paths to other nodes in its vicinity intact. Note that it is done only to reduce the computational complexity. In this way, we are sacrificing the paths from a node to other nodes in its vicinity, which pass through the new node and more optimal than the existing paths.

To handle leave event:

A node i maintains the list of k paths from it to all the nodes in V_i. If it finds that a node is no more its neighbor, it finds out the nodes for which this node contributed to form one of the vertex disjoint paths. Due to the lack of presence of this node only one of the K vertex disjoint paths for some neighboring nodes has been destroyed. For those nodes it compensates that path by finding out a new optimal vertex-disjoint path.

The connectivity results in the mobile scenario have been proved in Theorem 4, 5 and 6. To prove the results we have assumed that after the join or leave event when the nodes come to the stable condition, the resulting topology will be K-connected provided the graph G_{max} is K-connected. Note that G_{max} is the graph obtained if all the nodes transmit with their maximum power when the network comes to the stable condition after the change in the network.

Theorem 4: If a newly added node maintains K optimal vertex-disjoint paths to all the nodes in its vicinity, and all the other nodes maintains all the paths they were previously maintaining, and K vertex-disjoint paths to the newly added node (if the node is in vicinity), then there exist K vertex disjoint paths between any two nodes in the resultant network.

Proof: We prove this theorem in the same line as we did in theorem 3. Let us suppose that G_{ka}^* is not K-connected. So there exists at least one set of $K-1$ nodes, by removing which we can get a graph that is not connected. Let's denote this graph by G''. Let G' be the graph obtained by removing the same set of $K-1$ nodes from G_{max}, which were removed in forming G'' from G_{ka}^*. As G_{max} is K-connected, so obviously G' is connected. Let G_a^* be the graph obtained by running the algorithm *K-connected_Minimum_Topology* with $K = 1$ on the remaining set of nodes, i.e., the set obtained after removing $K-1$ nodes. According to Theorem 1, G_a^* is connected because the graph G' is connected.

As G_a^* is connected and G'' is not connected, at least one edge of G_a^* will not be present in G'' (Note that G_{ka}^* and G_a^* are constructed in the same manner). Let us suppose that the edge $\{u, v\}$ is one of such edges in G_a^*, which is not present in G''. The presence of edge $\{u, v\}$ in G_a^* implies that $\{u, v\}$ is the optimum path from u to v. If we consider the vicinity of any node j in whose vicinity, a node k has been newly added we see that we have not considered the paths from j to all other nodes except k in its vicinity, passing through node k. In this way we may miss one of the K optimal vertex-disjoint path from j to any other node. But the edge $\{u, v\}$ is not obviously one of such paths, as those paths will have more than one edge (minimum path from j to i through k is $j-k-i$). So if $K-1$ vertices were not removed from the graph G_{ka}^*, then the edge $\{u, v\}$ would be at least the K-th optimal path from u to v in G_a^*. So the edge $\{u, v\}$ is one of the K vertex disjoint optimal paths from u to v in G_{ka}^*. By removing the set of $K-1$ nodes from G_{ka}^* we can destroy at most $K-1$ vertex disjoint paths. But the edge $\{u, v\}$ will still be present, since it is one of the K optimal vertex disjoint paths from u to v and also removal of a set of $K-1$ nodes can not destroy the edge $\{u, v\}$. (Note that u and v are nodes selected from remaining set, so it would not have been removed.)

So the edge $\{u, v\}$ will be present in G''. So our assumption that $\{u, v\}$ is not present in G'' is not correct. This implies that all edges present in G_a^* are also present in G''. So G'' is connected. Thus G_{ka}^* is K-connected.

So the network is globally K-connected. ■

Theorem 5: If a node disappears from the vicinity of another node, then the algorithm K-MRTC ensures the K-connectivity of the resultant topology.

Proof: When a node finds that a node is no more in its vicinity, it finds out at most one path to some neighbors to which it had one path through the leaving node. Now, if we consider that the node has been died due to the lack of power, then in the new graph all the nodes maintain K-optimal vertex disjoint paths to all the nodes in its vicinity. So according to Theorem 3, the topology is globally K connected. ■

Theorem 6: If a node changes its position, then the algorithm K-MRTC ensures the K- connectivity of the resultant topology.

Proof: Theorem 4 ensures that the addition of one node in the vicinity of a node does not affect the connectivity. Theorem 5 states that a node finds out the link in its vicinity

to make up the loss of links by a leaving node from its vicinity and thus maintain the connectivity. In the stable condition after the changing of position of a node, the incident can be considered as a combined effect of this two, and when the node come to a static condition these two cases can be executed separately. ■

VI. Performance Evaluation

To evaluate the performance of our topology control algorithm we perform extensive simulation. In the simulation, we consider only the static set of nodes. Random networks have been generated in a fixed grid size of 400×400. Number of nodes has been varied from 100 to 200, the power attenuation exponent has been taken as 2, and in every case the nodes have been randomly assigned power in the range of 22500 and 40000 units. For $\alpha = 2$, these maximum energy range corresponds to the maximum radius range of 150 to 200 unit. The average radius and the average node degree for K = 2 and K = 3 are shown in Fig. 2 and Fig. 3 respectively. Each data point on the graph is the average of 10 simulation runs.

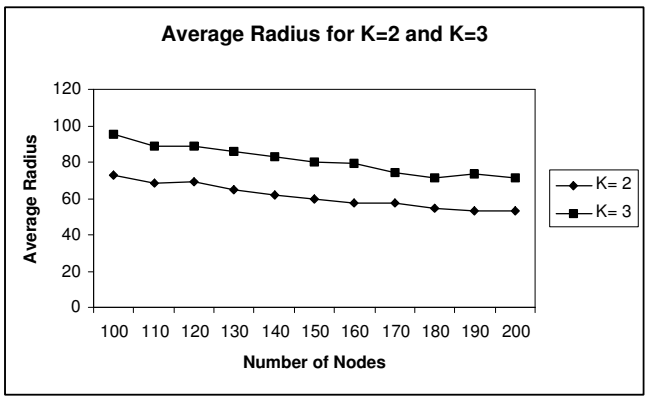

Fig. 2. Average Radius for $K = 2$ and $K = 3$

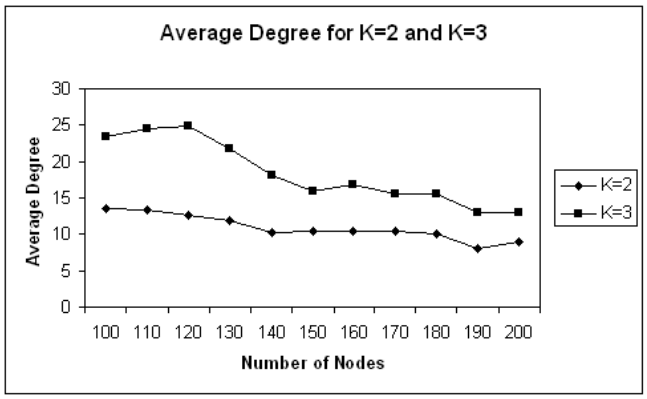

Fig. 3. Average Degree for $K = 2$ and $K = 3$

The graphs for average radius shows that the average power assigned to each node in the sensor network decreases with the increase in number of nodes in the same area. This means that increase in node density helps decrease the assigned power to the nodes. It is in congruence with the fact that more the number of nodes in the vicinity, more possibilities there are to get paths with smaller edge weight to reach another node. Regarding average degree, the nature of the graphs are a bit random, but the trends of average degrees of a node is to decrease with the increase in the number of nodes in the same area.

We compare the performance of our algorithm (we call it path-based algorithm) to that of the Bahramgiri's algorithm (cone-based algorithm [2]) as it is the only work that presents the algorithm for topology control in mobile scenario. We compare the performance of two algorithms in static scenario as it is also indicative of performance comparison in mobile scenario. In the simulation, Bahramgiri et. al. considered 200 nodes with maximum power 260 placed randomly in a grid of 400×400 . To compare our work with them we considered the same parameters and the result of comparison is presented in Table I. The results for the cone-based algorithm have been taken from [2]. For path-based algorithm, every data is the average of 10 simulation runs. Average degrees have been rounded to the nearest integer.

Connectivity	*Algorithm*	*Average Degree*	*Average Radius*
2	Cone-based	15	158.388
2	Path-based	9	54.589
3	Cone-based	22	184.025
3	Path-based	13	71.330

TABLE I
COMPARISON BETWEEN CONE-BASED ALGORITHM AND PATH-BASED ALGORITHM FOR 2 AND 3 CONNECTIVITY

From Table I it is evident that path based algorithm outperforms cone based algorithm in terms of both average degree and average radius.

VII. Conclusion

In this work, the complete focus is on developing a distributed algorithm for getting K-connectivity in the sensor network along with minimizing the power assigned to each node. Every node runs the algorithm based only on the locally accumulated data, and it has been proved that upon convergence, the network becomes globally K-connected. The algorithm does not require a change in the primary deployment of the sensor network. Also this algorithm can be applied in the mobile scenario efficiently, as very little maintenance is required in mobile scenario. We have presented the proof of correctness of K-connectivity in the mobile scenario.

As future work, we would like to simulate our algorithm in mobile scenario. Applying the same notion of our algorithm in 3-dimensional sensor network is another possible extension of this work.

References

[1] L. Bao, J. Garcia-Luna-Aceves. "Channel Access Scheduling in Ad hoc Networks with unidirectional Links." *Proc Discrete Algorithms and*

Methodsfor Mobile Computing and Communications (DIALM), pp. 9 - 18. 2001.

[2] M. Bahramgiri, M. T. Hajiaghayi, and V. S. Mirrokni. "Fault-tolerant and 3-dimensional distributed topology control algorithms wireless multi-hop networks." *11th IEEE International Conference on Computer Communications and Networks (ICCCN)*, pp. 392-398, 2002.

[3] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein. "Introduction to Algorithms", 2nd Ed. MIT Press, Cambridge, MA. U.S.A., 2001.

[4] Y. Chen; S. H. Son."A Fault Tolerant Topology Control In Wireless Sensor Networks," *3rd ACS/IEEE International Conference on Computer Systems and Applications*, 2005.

[5] M. T. Hajiaghayi, N. Immorlica, V. S. Mirrokni. "Power Optimization in Fault-Tolerant Topology Control Algorithms for Wireless Multi-hop Networks." *9th annual international conference on Mobile computing and networking*, Sept, 2003.

[6] D. Kim, C. Toh, Y. Choi. "On Supporting Link Asymmetry in Mobile Ad hoc Networks." *Proc. IEEE Globecom*, pp 2798 - 2803. 2001.

[7] L. Li, J. Y. Halpern. "A Minimum-Energy Path-Preserving Topology Control Algorithm." *IEEE Transaction on Wireless Communication*, Vol. 3, No.3, pp. 910-921, May, 2004.

[8] N. Li, J. C. Hou. "FLSS: A Fault-Tolerant Topology Control Algorithm for Wireless Networks". *ACM Mobicom*, pp. 275-286, 2004.

[9] N. Li, J. C. Hou. "Topology Control in Heterogeneous Wireless Networks: Problems and Solutions." *Proc. IEEE INFOCOM*, 2004.

[10] L. Li, J. Halpern, V. Bahl, Y. Wang, and R. Wattenhofer. "Analysis Of A Cone-Based Distributed Topology Control Algorithms For Wireless Multi-Hop Networks." *Proc ACM Symposium on Principle of Distributed Computing (PODC)*, August 2001.

[11] N. Li, Jennifer C. Hou, and L. Sha. "Design and Analysis of an MST-Based Topology Control Algorithm," *IEEE INFOCOM*, 2003.

[12] J. Liu and B. Li. "Distributed Topology Control in Wireless Sensor Networks with Asymmetric Links," *GLOBECOM 2003 - IEEE Global Telecommunications Conference*, no. 1, pp. 1257-1262, Dec 2003.

[13] M. Marina, S. Das. "Routing Performance in the Presence of Unidirectional Links in Multi-hop Wireless Networks". *Proc. ACM Mobihoc*, pp. 12 - 23. 2002.

[14] M. Pearlman, Z. Hass, B. Manvell. "Using Multi-hop Acknowledgements to Discover and Reliably Communicate over Unidirectional Links in Ad hoc Networks." *Proc. Wireless Communications and Networking Conference (WCNC)*, pp 532 - 537. 2000.

[15] R. Prakash. "A Routing Algorithm for Wireless Ad hoc Networks with Unidirectional links." *ACM/Kluwer Wireless networkork* 7, 6, pp 617 - 625, 2001.

[16] V. Ramasubramanian, R. Chandra, D. Mosse. "Providing a Bidirectional Abstraction for unidirectional Ad hoc networks." *Proc. IEEE INFOCOM*, pp. 1258 - 1267, 2002.

[17] I. Saha, L. K. Sambasivan, R. K. Patro, S. K. Ghosh. "Distributed Fault Tolerant Topology Control in Wireless Ad-hoc Sensor Network". *In Proceedins of 3rd IEEE International Conference on Wireless and Optical Communications Networks (WOCN)*, Bangalore, India, 2006.

[18] P. Santi. "Topology Control in Wireless Ad hoc and Sensor Networks". *ACM Computing Surveys*, Vol. 37, No. 2, pp. 164-194, June, 2005.

[19] A. Srinivas, E. Modiano. "Minimum Energy Disjoint Path Routing in Wireless Ad-hoc Networks." *Mobicom*, 2003.

[20] C. C. Shen, C. Srisathapornphat, R. Liu, Z. Huang, C. Jaikaeo, E. L. Lloyd. "CLTC: A Cluster -Based Topology Control Framework For Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, Vol. 3, No. 1, pp. 18-32, Jan-Mar, 2004.

[21] J. E. Wieselthier, G. D. Nguyen, A. Ephremides. "On the Construction of Energy-Efficient Broadcast and Multicast Trees in Wireless Networks." *Proc. IEEE INFOCOM*, pp. 585 - 594, 2000.