STREAM: Sensor Topology Retrieval at Multiple Resolutions

Budhaditya Deb, Sudeept Bhatnagar and Badri Nath

Computer Science Dept. Rutgers University.

***Abstract*- Large-scale dense sensor networks require mechanisms to extract topology information that can be used for various aspects of sensor network management. Many network properties can be inferred from a relatively low-resolution representation of topology. Different topology resolutions suffice for different management applications to perform at a desired level. In these cases, it is an overkill to retrieve the entire topology of large-scale dense networks particularly because sensor nodes are energy constrained. In this paper, we describe a distributed parameterized algorithm for Sensor Topology Retrieval at Multiple Resolutions (*STREAM*), which makes a tradeoff between topology details and resources expended. The algorithm retrieves network state at multiple resolutions at proportionate communication cost by adaptive spatial sampling. We also define various classes of topology queries and show how the parameters in the algorithm can be used to support queries specific to sensor networks. We show that the topology determined, albeit at a low resolution, is sufficient to approximate actual network properties.**

I. INTRODUCTION

Network Topology is an important attribute of the network state as it aids in network management and performance analysis. For any network, accurate knowledge of network topology is a prerequisite to many critical network management tasks, including proactive and reactive resource management and utilization, server siting, event correlation, root cause analysis, growth characteristics and even for use in simulation for networking research. In sensor networks[1], where redundant, cheap nodes operating on battery, are used to form a large-scale dense network, energy-efficiency is the key criterion guiding the design of network protocols. Accordingly any topology discovery algorithm should also be energy-efficient.

An important characteristic of sensor networks is that individual nodes may be inaccessible when they operate in doze mode or due to node failure or due to channel errors. Thus, a topology discovery algorithm may not be able to contact individual nodes and might only be able to retrieve partial topology. We show that different levels of partial topology suffice for determining different network properties. In fact, this observation opens up a new avenue for energy-efficient topology discovery where we can save a large amount of energy by aiming to retrieve partial topology (compared to retrieving complete network graph) without paying much in terms of inference of network characteristics.

In this paper, we describe a distributed parameterized algorithm for Sensor Topology Retrieval at Multiple Resolutions (*STREAM*), which makes a tradeoff between topology details and resources expended in large scale, dense networks. STREAM serves as a tool to retrieve network topology at any desired resolution with a communication cost proportional to the retrieved topology details. The rest of the introduction section provides the motivation, main contributions and overview of the algorithm.

A. The Need For Multi-Resolution Topology Discovery

Consider the following observations about large-scale dense networks:

- We found that many topological properties in dense networks can be inferred accurately using a low-resolution representation of topology.

  For example, if the network administrator needs to test the robustness of the network by verifying if each node in the network has at least three disjoint paths (to a monitoring node), using STREAM she only needs to recover 17% of the edges (in a network of 1000 nodes with average degree 25).[1] This results in reduction of 83% in energy consumed with respect to retrieving the complete network graph [2].

- Our simulation studies show that different topology resolutions are required for different applications to perform at a desired level.

  For example, on retrieving only 10% of the network edges (for a 1000 node network with average degree of 25) the average single source shortest path length increased by only 6% of the hops from the optimal. Similar bound (6% increase in hop length) for all pair shortest paths required 25% of the network edges. As the granularity of the recovered topology increases, many network properties converge towards the real values for the network.

- In some situations, network administrators may not be willing to spend more than a given amount of energy to retrieve certain network state. Given the energy budget we should be able to retrieve the maximum amount of information about the network. Note that the energy budget allotted to infer different network properties would also depend on the *importance* of the network property evaluated and how much the administrator is willing to tolerate in terms of the property degradation.

The above observations clearly highlight the need to have an ability to retrieve topology at multiple resolutions. STREAM provides us this ability.

B. Main Contributions

Our first contribution is a *distributed parameterized Algorithm* for Sensor Topology Extraction at ***Multiple resolutions (STREAM)***. We introduce the notion of *Minimal Virtual Dominating Set (MVDS*), which is the minimal set of nodes required for extracting topology at a desired resolution. The *MVDS* concept forms the basis for *adaptive spatial sampling* of the network. Retrieved network topology ranges from the backbone to the complete network graph.

[1] Typically, *Sensing Range << Communication range* (E.g. in heat sensors and motion sensors). A network barely sufficient to provide sensing coverage could be sufficiently dense for providing communication coverage.

[2] Sensor network specific properties such as field exposure and coverage [11],[12] also converge quickly towards the real values of the network even for at fairly low granularity of location information..

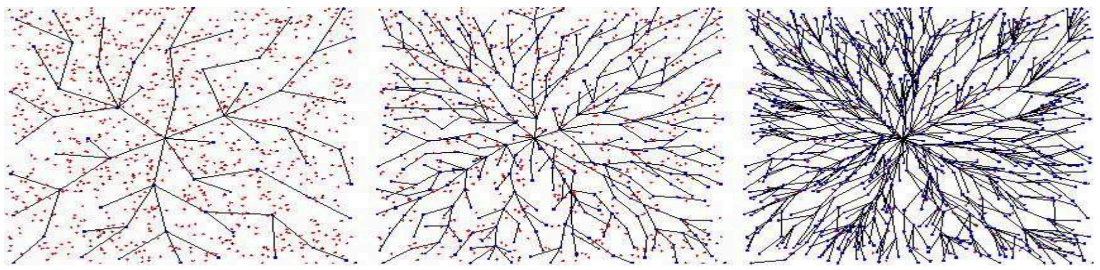

Figure 1: Topology of network with (1000 nodes in a 400x400 m^2 field, communication range 40m) retrieved at multiple resolutions. In all cases topology discovery is initiated at the center node. The nodes on the tree form the responding MVDS. The effectiveness of STREAM in selecting the MVDS is clearly depicted by the uniform distribution of the MVDS across the sensor field.

The heuristics to approximate the MVDS and its distributed implementation involves intelligent use of local timers. The timers are also responsible for creating an optimal topology aggregation tree for the MVDS and minimizing the message complexity of the process.

Our second contribution is a description of various types of topology discovery queries relevant for sensor networks and applications that can use these queries.

We give formal analyses to describe the expected behavior of the algorithm under various network conditions and use them to map the queries to the parameter values in STREAM.

C. Overview:

STREAM utilizes the broadcast property of wireless medium called the *Wireless Multicast Advantage*[27]. A node can detect the presence of its neighbors by eavesdropping on the communication channel. Thus by selecting a subset of nodes, approximate topology can be created by merging their neighborhood lists. The resolution of the topology depends on the cardinality and structure of the chosen set of nodes. For example, to construct a minimal backbone tree of the network, we only need to merge the neighborhood lists of the minimal dominating set of the network graph.

The algorithm runs in two stages. First a monitoring node, which requires the topology, sends a topology discovery request to all the nodes in the network by controlled flooding. The request contains two parameters called *virtual range* and *resolution factor*. These parameters are used to select a minimal set of nodes required to retrieve topology at a desired resolution. We define this set as the *Minimal Virtual Dominating Set (MVDS).* During this stage, the nodes are colored *red* or *black* such that each *red* node is a neighbor of some *black* node and the black nodes form the MVDS. Further, at the end of the first phase, a *black* node tree rooted at the monitoring node is set up. In the second phase, the *black* nodes reply back to the request with a subset of its neighborhood list, determined by the resolution factor. Each black node aggregates the data received from its children black nodes and sends it to its parent in the tree. Figure 1 illustrates the results of applying STREAM to select multi-resolution MVDS. The *MVDS* spatially samples the network to retrieve topology at a given resolution. The following factors make STREAM extremely suitable for sensor networks:

- The MVDS is created using message complexity of *N* (number of nodes in the network), i.e. each node sends *only one* packet and compares well to a centralized scheme. Such an algorithm using only one packet per node makes it very useful for energy constrained sensor networks.
- The cardinality of the MVDS is dependent only on the network field dimensions, the communication radius, and required resolution and is almost constant with respect to density of the network. Hence the message complexity does not increase with increase in density of the network.
- *The MVDS* tree rooted at the monitoring node, is optimal in the number of hops from monitoring node. Thus the response packets travel the minimum number of hops to reach the monitoring node.
- Local timer mechanisms to forward request packets and select the MVDS makes it possible to do the above using only one packet per node. The timers are completely local and do not require any *time synchronization.*
- The entire message complexity to retrieve topology is proportionate to the resolution retrieved. If *x%* of the topology (see definitions in section 2) is retrieved, the message complexity is $N(1+x+c)$, $(0<=c<=x<=1)$. Since the aggregation tree is also optimal, the algorithm is energy conserving and hence important for sensor networks.

Rest of the paper is organized as follows. In section 2 we describe in detail our proposed algorithm. In section 3 we analyze the algorithm and the properties of retrieved multi-resolution topologies. In section 4 we describe various classes of multi-resolution topology discovery queries relevant for sensor network. We assume uniform network topology, and zero channel errors and node failures for the analyses in sections 2, 3 and 4. In section 5 we describe the changes required to make the algorithm adaptive to any arbitrary network condition. In section 6 we summarize related work and conclude the paper in section 7.

II. Topology discovery Algorithms

We first give the basic assumptions and the topology resolution metrics. Then we list two trivial approaches for topology discovery and their probabilistic variations, which are based on the breadth first search. Finally we describe the STREAM algorithm.

A. Assumptions and Network Model

In this section, we describe the assumptions for purposes of describing the algorithms. Later, for complete evaluation of these algorithms, we relax some of the assumptions.

The sensor network topology is a connected disk graph with a fixed circular communication range *R* (the radius of the disk). Channels are error-free such that all packets are reliably transmitted. We also assume that node failure rate is zero i.e. all nodes are active during the execution of the algorithm

All the algorithms follow three basic stages of execution.

- A *monitoring node,* also called an *initiating node,* requiring the topology of the network initiates a *topology discovery request*.
- This request diverges throughout the network reaching all active nodes.
- A response action is set up which converges back to the initiating node with the topology information. Nodes respond to the topology request with topology information (e.g. adjacency lists).

We assume that the request divergence is through controlled flooding[3] so that each node forwards the discovery request exactly once. Since wireless is a broadcast medium of communication, a node can collect its neighborhood list by eavesdropping on the communication channel[4].

The resolution of the topology is defined as:

– *Edge Resolution:* Ratio of the number of *retrieved* edges and the actual number of edges in the network.

– *Node Resolution:* Ratio of the number of *retrieved* nodes and the actual number of nodes in the network.

B. Trivial Approaches

We describe approaches and their probabilistic variations for topology discovery.

– *Direct Response:* When a node receives a topology discovery request, it forwards this message and immediately sends back a response with its neighborhood list along the reverse path. In the probabilistic variation, nodes decide to reply back with some probability p and report all their edges.

– *Aggregated Response:* A node receives a packet, it forwards the request immediately but waits to aggregate the information from its children before sending its own response. In the probabilistic variation, all nodes respond but report each of their edges with probability p.

The probabilistic variations can be used to retrieve topology at multiple resolutions. In the first case, since nodes respond back randomly, aggregation is not guaranteed. In the second case, since all nodes are reporting back, the number of responses is same as its non-adaptive counterpart, albeit with a reduced per- response cost. In the probabilistic variations, no guarantees can be provided that *each* node will be covered. Also bounds on some graph properties such as shortest path routes computed on multi-resolution topologies cannot be guaranteed using the probabilistic algorithms. In our proposed approach we provide such guarantees with lesser overhead.

C. STREAM Algorithm

STREAM selects a subset of nodes to reply to the topology discovery query with their neighborhood information. The number of these nodes determines the resolution of retrieved topology. The overhead is proportional to the resolution retrieved. The key to the STREAM algorithm is to have a mechanism to control the cardinality of the responding set. The conceptual framework, which allows STREAM to do this, is described below:

Consider a graph $G(V, E(R))$ where an edge exists between nodes v_i and v_j if their distance is at most R.[5] For wireless networks, having communication range as *R* defines the network graph. We define the following terms on this graph:

– *Virtual Edge Set (E(r)):* The subset of E(R) such that each edge in the set has endpoints at most distance r apart. In this case, r is called the Virtual Range.

– *Virtual Graph G(V, E(r)):* The sub-graph of G which has only edges from E(r).

– *Minimal Virtual Dominating Set MVDS(r):* A minimal dominating set on G (V, E(r)).

STEM is a coloring algorithm, which creates an *MVDS(r)* based on the *virtual range r*. Since a decrease in *r* causes a decrease in the number of virtual edges, the cardinality of *MVDS(r)* increases as *r* decreases. STREAM selects the nodes in *MVDS(r)* to respond to topology discovery queries with *r* providing a resolution control parameter. Note that finding a minimum dominating set of unit disk graph is NP-Complete [26]. STREAM is a distributed approximation algorithm to find *MVDS.*

1) STEM Request Propagation Phase

The request propagation in STREAM is similar to the other approaches using controlled flooding. The algorithm takes three user-specified parameters that control the topology resolution from the minimal backbone tree to the complete network graph. The parameters are defined as follows:

– *Virtual Range ($r \in [0,R]$):* The virtual range controls the cardinality of the MVDS as described earlier.

– *Resolution Factor ($f \in [0,1]$):* A member of the MVDS reports this fraction of edges originating from it. For example, for f=0.4 a node should return 40% of its neighborhood list.

– *Query Type (Q):* This defines the set of queries, which STREAM can support. The query type maps to specific filters and aggregating functions that may be required for general-purpose multi-resolution information retrieval.

Thus, each query is of the form: STREAM ***(r, f, Q).***

[3] A probabilistic flooding as described in [15] can be used as well.

[4] Note that each node must send at least one packet for other nodes to know its existence. A topology discovery request from each node ensures: 1) All nodes receive a packet; 2) Nodes have complete neighborhood lists.

[5] Such a graph is referred to as a unit disk graph in literature (for R=1)

To find the *MVDS* we use four colors. As the topology discovery request propagates, different nodes are colored according to their definitions given below:

- ***White:*** Yet undiscovered node, or a node which has not received any topology discovery request packet.
- ***Black*:** A node in the *MVDS* which replies to topology discovery request with its neighborhood set. After becoming *black*, a node discards all other request packets.
- ***Red*:** A node which is *virtually dominated* by at least one *black* node, i.e., it is inside the virtual range *r* of the *black* node. After becoming *red*, a node discards all other request packets. The node is said to be *attached-to* the corresponding black node.
- ***Blue:*** A node which receives a packet from a *red* or *blue* node or a node which is within communication range of a *black* node but outside its *virtual range*. It waits for a time period for some other node in its virtual range to become *black*. Otherwise it itself becomes a *black* node.

The request packet contains the following fields:

– Sending Node ID and Node Color

– Black Node it is attached to

– Number of Hops from Monitoring Node.

– Virtual-range, resolution factor, query type

Each node uses two timer functions defined as follows:

1. *Request Forwarding Timer (R_F_T (distance)):* The time between receiving a discovery request and forwarding it, is the forwarding delay. The parameter distance is the distance between the receiving node and the sending node[6]. The timer is inversely proportional to the distance i.e. the farthest node forwards the earliest. The timer is also designed such that a node h hops away from the monitoring node forwards before a node h+1 hops away.[7]
2. *Black Node Formation Delay (B_F_T(distance)):* This is the time period after which a blue node changes to black. The timer is proportional to its deviation from 2r distance from the black node at previous hop. Thus a node which is closest to the 2r distance becomes black first.

We note that the above timers work without *any global time synchronization* using only local knowledge.

Initially all nodes are *white*. The *initiating node* is colored *Black* and initiates the process by broadcasting a topology discovery request. As the request propagates, each node is colored *black*, *red* or *blue* according to the coloring algorithm. Figure 2 illustrates the coloring sequence. Figure 3 describes the action taken by a node on receiving a discovery request. The algorithm is described as follows.

- The node that initiates the topology discovery request is colo*red black* and broadcasts a topology discovery request packet.
- Lines 1-13 describe the operations when nodes receive packet from a *black* node. All *white* and *blue* nodes within virtual range of a *black* node become *red*. Other nodes that are in communication *range* but outside *virtual range* become *blue*. After ***R_F_T****(distance)* time a node forwards the discovery request if it has not already done so. All *blue* nodes start a timer to become *black* with a *Black Node Formation delay* function ***B_F_T****(distance).*
- Lines 14-18 describe operations when a node receives a packet from a *red* or *blue* node. When a *white* node receives a packet it becomes *blue.* It then starts a timer ***B_F_T****(distance),* to become *black.*
- If any *blue* node receives a packet from a black node in its *virtual range*, it cancels its B_F_T and becomes *red*. Once nodes are *red* or *black*, they ignore other topology discovery request packets. Figure 3 illustrates the algorithm with an example.

Ideally a minimum number of nodes should reply back to provide a desired resolution (in this case the MVDS). Since the problem of finding minimum dominating set is known to be NP-complete (MVDS is a minimum dominating set on the *virtual graph*), we use a greedy approach for finding the set of nodes to reply back, i.e. at each step a node that covers the maximum number of *yet* uncovered nodes is chosen to become *black*. It is not possible to know the best candidate to become *black* without global knowledge of neighborhood sets. We use timer mechanisms (based on only local knowledge) to approximate the greedy approach.

The timer mechanism achieve the following:

– Maximal number of nodes become blue so that the probability of having the best candidate to become black is higher. (using R_F_T delay)

– The better candidates among the candidate blue nodes become black with a lower delay. (using B_F_T delay)

Consider the example in Figure 4. When any node forwards, a node with a lower overlap with a covered region is expected to reach a larger number of uncolored nodes. The communication range of node *a* has a lesser overlap with that of *b* as compared to that of *c.* When *b* is further away from *a* than *c,* it is expected to cover a larger unexplored region. R_F_T ensures that *b* forwards before *c.*

The second condition is achieved with a black node formation delay at each blue node proportional to its nearness to 2r distance from a black node in previous step. This means that a blue node closest to 2r distance from previous black node would become black the earliest. Such a black node is expected to have minimum overlap of virtual region.

Note that at the end of the coloring phase, each node is either red or black if the network is connected. This is because

[6] We assume that the distance between nodes can be approximated by using GPS, signal strengths, etc

[7] Each node decides a time period $Epoch_{Start}$, to $Epoch_{End}$ within which they should forward the request. Suppose the node decides to forward with delay T_{delay} in its Epoch. Then the next hop node should forward after the $\mathrm{Epoch}_{\mathrm{End}}$ of previous hop. To ensure this, in the request packet we pass the value of ($Epoch_{End}$ - $Epoch_{Start}$ - T_{delay}.) which is added onto the T_{delay} of the next hop. We note here that here we don't need the actual time of epoch $Epoch_{Start}$, and $\mathrm{Epoch}_{\mathrm{End}}$ but only the time period $Epoch_{End}$ - $Epoch_{Start}$. The above scheme would ensure that a node at i^{th} hop would always forward *before* $(i+1)^{th}$ hop.

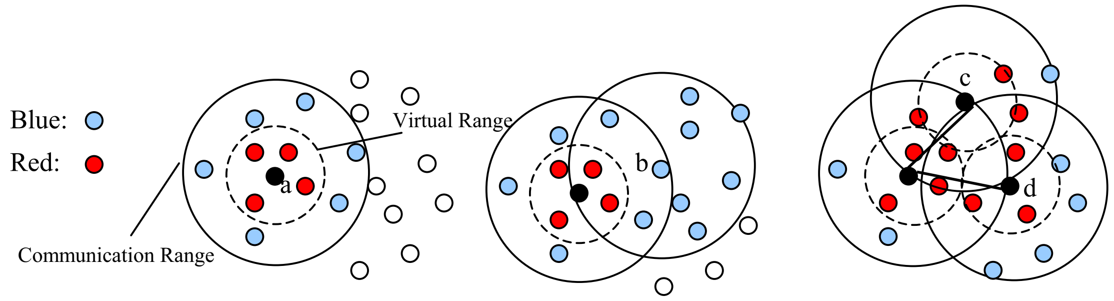

Figure 2: Illustration of the Coloring Algorithm. Node *a* becomes *black* and forwards a topology discovery request. Nodes within its virtual range *r* are colored *red* and nodes outside its virtual range but within its communication range *R* are colored *blue*. Node *b*, which is farthest from node *a* forwards the topology discovery request the earliest since forwarding delay is inversely proportional to the distance. Since node *b* was blue, all its *white* neighbors also become blue. All blue nodes start a timer to become *black*. Nodes *c* and *d* become *black* earlier than other blue nodes since they are closer to *2r* distance from previous *black* node *a*. They color their neighbors similar to earlier step. The *black* nodes *c* and *d* are children *black* nodes of node *a.*

```
RECEIVE_REQUEST_PACKET( recvColor, distance, r )
1.  if ( (recvColor==BLACK) & ( selfColor==WHITE) )
2.  if (distance < r)
3.  selfColor = RED
4.  FORWARD_REQUEST_PACKET(R_F_T(distance))
5.  if (distance>r)
6.  selfColor= BLUE
7.  B_F_T(distance)
8.  FORWARD_REQUEST_PACKET(R_F_T(distance))
9.  if ((recvColor==BLACK) & ( selfColor==BLUE ) & ( distance<r ))
10. selfColor=RED
11. cancel (B_F_T)
12. if (a request packet has not been Forwarded )
13. FORWARD_REQUEST_PACKET (R_F_T(distance))
14. if ( (recvColor==RED) OR ( recvColor==BLUE ) )
15. if (selfColor == WHITE)
16. selfcolor = BLUE
17. B_F_T(distance)
18. FORWARD_REQUEST_PACKET(R_F_T(distance))
```

Figure 3:Coloring Algorithm when node receives a topology request Packet

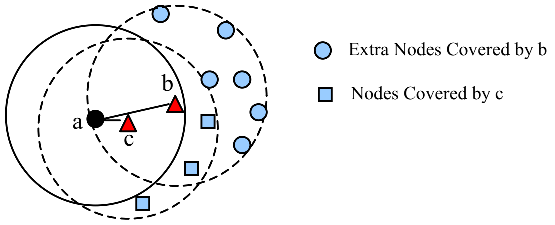

Figure 4: Illustration for the delay heuristic.

a blue node either becomes black if its B_F_T expires or becomes *red* if a node in its virtual range becomes black. Moreover the timer mechanisms try to reduce the cardinality of the black node set. Hence the black node set gives the required *MVDS(r).*

2) STEM Response Phase

During the first phase, the initiating node becomes the root of the *black* node tree where the parent *black* nodes are at most two hops away from their children *black* nodes. Each node has the following information at the end of this period:

- Each node knows its *parent black node*, which is the last *black* node from which the topology discovery was forwarded.
- Each *black* node knows the *default node* to which it should forward packets in order to reach the *parent black node*. This node is essentially the node from which it had received the topology discovery request.
- All nodes have their neighborhood information by eavesdropping on the communication channel.

Using the above information, the response action is described below:

- When a node becomes *black*, it sets up an *acknowledgement timer* (described later) to reply to the discovery request. Each *black* node waits for this time period during which it receives responses from its children *black* nodes.
- It aggregates all topology information from its children and adds *f fraction* of edges from its own region.
- When its time period for acknowledgement expires, it forwards the aggregated neighborhood list to the *default node* to its parent *black* node.
- In the general framework for information retrieval, the parameter *query-type*, is used to filter or aggregate the information. However for topology discovery we just merge the neighborhood lists.

For the algorithm to work properly, timeouts of acknowledgements should be properly set. The *acknowledgement timer* of a *black* node should always expire before its *parent black node* so that each *black* node forwards only after receiving responses from its children. For this we set a timeout value inversely proportional to the number of *hops* a *black* node is away from the monitoring node (the number of hops is obtained from the discovery request packet). We need an upper bound on the number of hops between extreme nodes. If the extent of deployment region and communication range of nodes is known initially, the maximum number of hops can be easily calculated. However, if that information is not available to the nodes, we can assume that the topology discovery runs in stages where it discovers only a certain extent of area at each stage. In this work we assume that the initiating node has knowledge of the node deployment area. Again we see that no global time synchronization is required.

III. ANALYSIS OF STREAM

In this section we state some of the analytical results related to STREAM. All the results assume that the node density is uniform and that the network is connected. We verify our analytical results with simulations, for which we modified the NS-2 simulator to incorporate details of STREAM.

A. *Request Propagation and Black Node Tree*

Proposition 1. Each node receives the first request packet in minimum number of hops.

Proof: Consider a node which is *h* hops away from the initiating node. The timer mechanisms ensure that a packet that has traveled *h-1* hops is forwarded before one which has traveled *h* hops away. Thus the first packet the node receives is from a node *h-1* hops away.

Proposition 2. If the network has symmetric links, the aggregation tree is optimal in the number of hops. As a consequence every black node and intermediate node is optimal number of hops from the sink in the retrieved network graph.

Proof: The *next hop to parent* node for any node is the one from which it received the request packet first. Since the first packet is received in the optimal number of hops (proposition 1), by sending the topology response packet to the *next hop to parent* node, the initiating node can be reached in optimal number of hops. Since the aggregation tree is included in the retrieved topology, by default, every node in the black node tree is optimal number of hops away from the monitoring node in the retrieved network graph.

Next we consider the nature of request propagation in our algorithm. Reference [29] shows that the number of hops in a *broadcast percolation* scenario for high-density networks can be well approximated by the following:

$$H = ceil(D/R) \qquad \text{... 1}$$

D = distance of a node from initiating node.

Hence, from *proposition* 1, we can assume that the request percolates as a wave encompassing a circular region of radius *iR* in the i^{th} step. This is because each node at *h* hops forwards request before any node at *h+1*. We use the above approximation later to evaluate the expected overhead of our algorithm.

B. *Expected number of Black nodes*

To find number of black nodes (*B)* for a given *communication range R* and *virtual range r,* we associate with each node a probability to become black based on the definition of node colors.

Let p = probability of each node to become black.

Each node would become black if no other node within its *virtual* range becomes black, i.e., if the expected virtual degree is *n.* Then for a given node, all its *n* virtual neighbors should not be black. Then the following equation is satisfied:

$$p = (1-p)^n \qquad \text{... 2}$$

$$n = \frac{N\pi r^2}{A} - 1, n \geq 1, \quad 0 < p < 1 \qquad \text{... 3}$$

N = Total number of nodes.

A = Area of the sensor field

We solve the above equation for *p* to get the probability of each node becoming black. Equation 2 is valid only when the expected number of nodes within a virtual region is greater than 1. As the virtual range reduces, this value can become less than 1. This means that each node has less than one node in its virtual range in the expected.

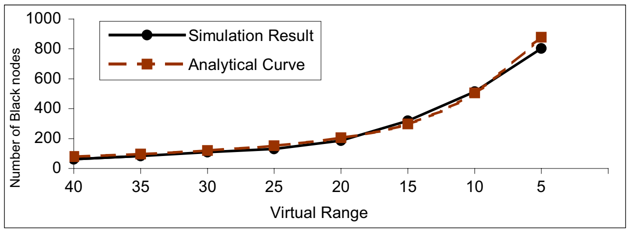

Figure 5: Comparison of Analytical expectation of the number of black nodes and simulations. Both simulation results are for 400x400m^2 field with 1000 nodes for a communication range 40m. The results show the number of black nodes formed for different virtual ranges

Let E = Expected virtual degree of a node

$$E = \sum_{x=1}^{N-1} x \binom{N-1}{x} \left(1 - \frac{\pi r^2}{A}\right)^x \left(\frac{\pi r^2}{A}\right)^{N-x-1} \qquad \text{... 4}$$

The total number of virtual edges in the network is half the sum of the expected degrees of all nodes, which is *0.5NE.* Since the number of *virtual* edges is less than the number of nodes (recall *n<1* in this case), nodes with no edges would always become black. Out of the rest of the nodes, only half would become black with the remaining being neighbors of these nodes. The expected number of black nodes is given by:

$$B = \left(N - \frac{NE}{2}\right) + \left(\frac{NE}{4}\right) = N - \frac{NE}{4} \qquad \text{... 5}$$

Thus the expected number of black nodes is:

$$B = \begin{cases} pN & for \quad n > 1 \\ N - \dfrac{NE}{4} & for \quad n \leq 1 \end{cases} \qquad \text{... 6}$$

Note that Equation 5 does not take into account the heuristics and edge effects. However the heuristics in STREAM make a black node configuration with lower overlap in virtual regions a more likely event. Hence the expected number of black nodes formed is lesser than that estimated by equation 5. Figure 5 plots the average number of *black* nodes formed in simulations compared to the number of *black* nodes from the analytical expectation of the number. The analytical expectation is very close to the simulations since the edge effects and the heuristics tend to negate each other.

Figure 6 and Figure 7 show comparison of STREAM against centralized *log(n)-approximate* solution provided by the greedy algorithm [15] for set cover to find the black nodes. The nodes for this simulation are uniformly spread in 200m x 200m field and the virtual range is equal to the communication range. Figure 6 shows the impact of increasing the number of nodes in the field. Figure 7 shows the effect of communication range for 1000 nodes in the field. In both cases, STREAM performs almost as well as the centralized solution which has global knowledge. This result shows that the heuristics used in STREAM timers perform well.

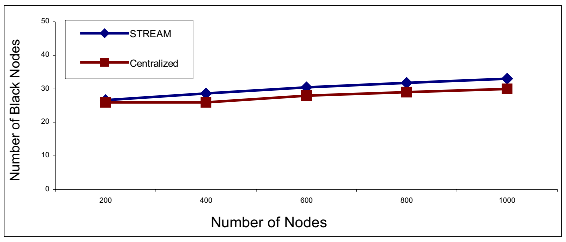

Figure 6: Number of *Black* nodes vs. total number of nodes for STREAM and Centralized greedy algorithm. The communication range is 30m and sensor field dimensions are is 200x200m^2

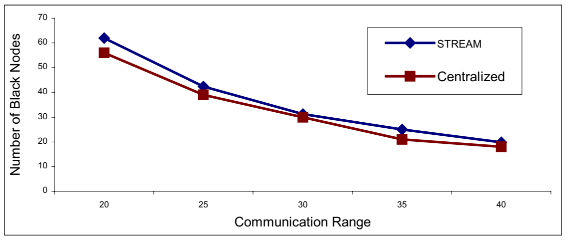

Figure 7: Number of *Black* nodes vs communication range for STREAM and centralized greedy algorithm. The size of sensor field is 200x200m^2 and the field has 1000 nodes.

C. *Retrieved Topology Resolution*

STEM takes two parameters, which control the resolution of the returned topology. Thus for a given *virtual range(r)* and *resolution-factor(f),* the returned topology resolution is computed as follows:

The number of black nodes formed for a *virtual range r* is given by equation 5. Each of these black nodes formed, returns *f%* of its edges. Recall that two or more black nodes may be formed in the same communication range if *virtual range* is smaller than R. Since we assume edges to be symmetric, we should account for the edges which may be reported by both its endpoints.

p = probability of a node to be black, from equation 5.

Thus both nodes associated with any edge have probability *p* of becoming black. When a black node reports any edge with probability *f* (the resolution factor), the actual probability of that edge being reported is:

$$x = p^2\left(f^2 + 2f(1-f)\right) + 2pf(1-p) \qquad \text{... 7}$$

Thus if d= average node degree, then

e= number of edges reported back = Nxd.

Figure 8 shows the plot of expected topology resolution as a function of STREAM parameters *virtual range(r)* and *resolution-factor (f).* The contours at the base are the *iso-resolution* curves obtained by intersection of a resolution plane with the curve. Each contour shows the set of *(r,f)* pairs which return the same resolution. We show in section V, how these can be used for selecting the parameter values for specific queries.

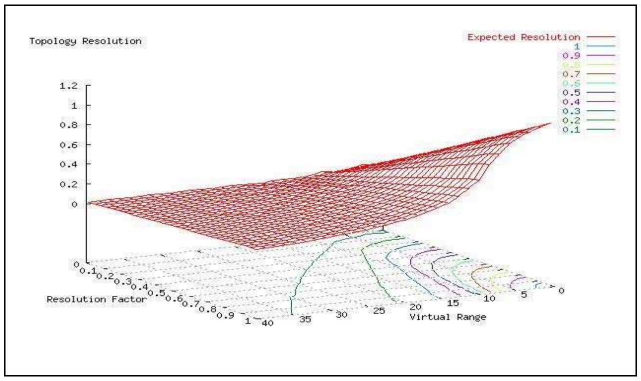

Figure 8: Expected Resolution of topology as a function of parameters Virtual Range and Resolution Factor. Each contour on the base shows the range of parameter values for topology returned at a particular resolution.

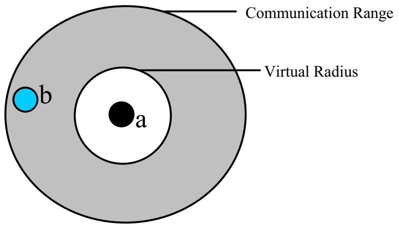

Figure 9: To Compute the Probability of an intermediate node to be black

D. *Overhead*

First we consider the packet overhead of the algorithm. For the initial phase of the algorithm, since all nodes forward the topology discovery request at most once, the overhead is *N.* During the acknowledgement phase, a subset of nodes are involved namely the black nodes and the intermediate nodes between two black nodes. For a given virtual radius, the expected number of black nodes is given by equation 5.

To compute the expected number of intermediate nodes, we consider the probability of the intermediate node to be also black. Consider Figure 9 with a black node *a* and an intermediate node *b*. We see that *b* is black with probability *p* only if it falls outside the virtual range of *a.* Thus the probability *P* that an intermediate node is *black* is give by:

$$P = p\frac{R^2 - r^2}{R^2} \qquad \text{... 8}$$

Hence the total communication overhead of the acknowledgement phase is equal to the expected number of black nodes plus expected number of intermediate nodes which are blue. Since each black node except the monitoring node has one intermediate node, the total expected overhead is given by:

$$O_P = N + Np\left(2 - p\frac{R^r - r^2}{R^2}\right) \qquad \text{... 9}$$

Next, the overhead incurred by STREAM is analyzed in terms of total bytes transmitted. The energy spent is equivalent to the trend shown by the byte overhead. Let,

C = constant packet header overhead per packet.

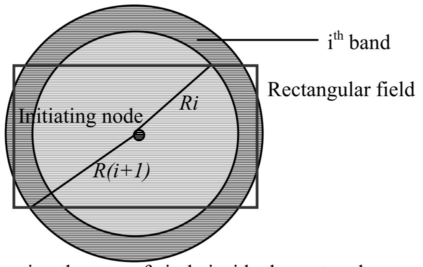

Figure 10: Computing the area of circle inside the rectangle.

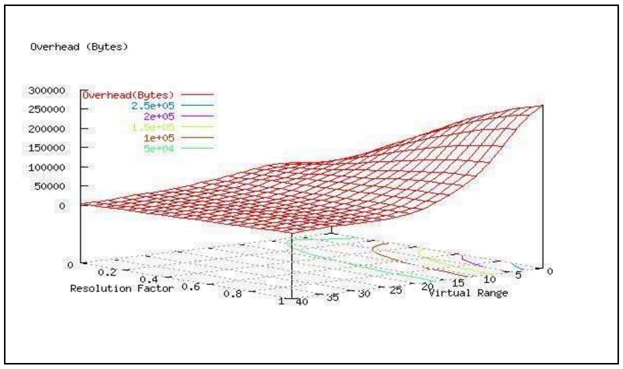

Figure 11: Expected Byte overhead of STREAM for different values of parameters (, N=1000, R=40m, 400x400m). Constant packet overhead C=5 bytes, Overhead per edge information C_f = 2 bytes. The contours show the range of parameter values which require equal overhead to return topology.

C_f = Overhead per edge information sent.

$B_i(r)$ = number of Black nodes at i hops from the initiating node, a function of virtual range.

Each black node sends *f%* of the edges in its neighborhood. This information travels to the initiating node which is *i* hops away. Thus the expected byte overhead O_T of the response process is given by:

$$O_T = C_f x d \sum_{i=1}^{H} iB_i + C\sum B_i \qquad \text{... 10}$$

Where,

H =maximum number of hops from initiating node

d = average node degree

x is given by equation 5

Equation 9 uses the maximum hop distance *H* in computing the overhead. *H* is not known in advance at the initiating node. However, from section A, we can approximate H by the following:

$$H = ceil(D/R) \qquad \text{... 11}$$

D = distance of farthest node from initiating node.

From the assumption in equation 10, we see that any black node i hops away from the initiating node lies in the band between the circles of radii *R(i+1)* and *Ri* centered at the initiating node. If we know the area of this i^{th} band, the expected number of black node in that region can be computed. For exposition we consider the sensor field to be a rectangle with the initiating node somewhere inside the rectangle as shown in Figure 10.

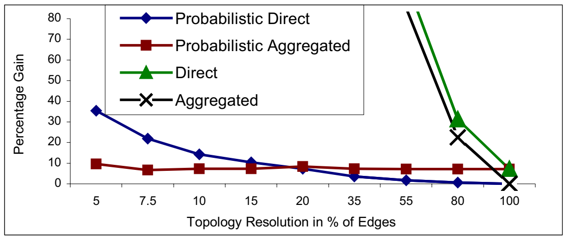

Figure 12: Relative Gain in Overhead of STREAM over Aggregated, Direct and their probabilistic variations.

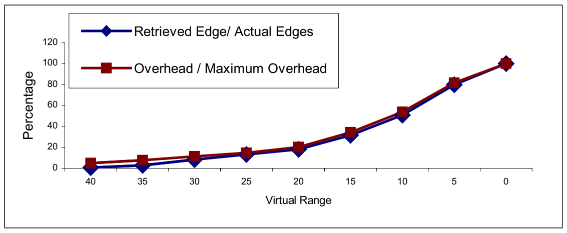

Figure 13:. Graph shows the relative overhead (with respect to maximum overhead) incurred in retrieving topologies at different resolutions. (N=1000 nodes, communication range 40m, and field size 400x400 m^2)

The i^{th} band is shown using the two circles, which cut the rectangle at maximum of eight points. If the area for each of the circles inside the rectangle is $A_{R(i+1)}$ and A_{Ri}, the number of black nodes in the band between the two circles (dark shaded region inside the rectangle) is given by:

$$B_i = \left(A_{Ri} - A_{R(i-1)}\right)\frac{B}{A} \qquad \text{... 12}$$

Note that calculating this is trivial once we get the intersection points of the circles with the sides of the rectangle. We can use the above procedure to find the number of black nodes for a sensor field of any general polygonal shape. In this work we only consider field to be a square region of dimensions 400x400 m^2.

Figure 11 shows the plot of STREAM overhead against different *virtual-ranges* and *resolution-factors.* Figure 12 and Figure 14 show the actual overhead incurred by STREAM. In evaluating the byte overhead, we assume a constant packet header of 5 bytes and an additional 2 bytes of information per edge reported in the packet. Total bytes transmitted during the entire operation characterizes overhead.

Figure 12 shows the percentage extra overhead incurred by direct response and aggregated response and their probabilistic variations over STREAM to retrieve equal resolution topologies. The probabilistic variations perform better than their non-adaptive counterparts, but are significantly worse than STREAM. Moreover, STREAM guarantees that each node would be reported whereas the probabilistic variations cannot provide such a guarantee.

Figure 13 shows the relative overhead for different virtual ranges. Observe that the relative overhead curve (as a percentage of the overhead incurred for retrieving the complete topology) closely follows the resolution. The relative difference at low resolutions is due to the constant overhead of the coloring phase of the algorithm.

Required Node Resolution	Returned Node Resolution		
	Average	*Maximum*	*Minimum*
0.8	0.813	0.832	0.791
0.6	0.621	0.646	0.605
0.4	0.439	0.456	0.416
0.2	0.238	0.269	0.213

TABLE I. PERFORMANCE OF NODE-CONSTRAINED QUERY ON A RANDOM 1000 NODE TOPOLOGY IN A 400X400M^2, R =40M.

IV. MAPPING QUERIES TO PARAMETRS.

In this section we describe various types of queries that would be relevant in the context of sensor networks. While this is not meant to be an exhaustive list of queries, it would encompass a broad range of applications. STREAM algorithm can be invoked using three parameters: virtual range, resolution factor and query type. STREAM takes the following form:

– STREAM (r, f, Q)

To handle various queries we frequently face the question: *Given the required topology resolution what values of parameters should be chosen?* In general the choice of parameter values would depend on the following factors:

- *Overhead:* Choose parameter values, which require the minimum overhead for a given resolution.
- *Properties of Multi-resolution Topologies:* Given a resolution what property needs to be maintained in the retrieved graph.

Note that, the retrieved topology at a desired resolution might be required to possess certain properties. The trade-off here is between the overhead and maintenance of graph properties. For example, to test whether each node has two edge-disjoint paths to the sink, the retrieved topology should have minimum node degree of two. However, exploring these tradeoffs is orthogonal to the focus of this paper and is part of our future work.

In this work, we try to *minimize the overhead* while getting a desired resolution. This is discussed in detail in the *Overhead Constrained Query,* at the end of this section. We use simulations (averaged over 20 runs with randomly chosen initiating nodes) to show the effectiveness of STREAM in dealing with different types of queries.

A. Node Constrained Query:

The node-constrained query is of the form: *“Return a representation of the network with x% of the nodes”.* Such a query helps in determining the approximate density distribution of the network while not requiring all the nodes to be returned.

For this query, we construct the minimal backbone topology by setting the parameter *virtual range (r)* equal to communication range *R* and *resolution-factor (f)* equal to the desired fraction *x.* The following lists various node-constrained queries:

– STREAM(R, x, Node-Constrained-Query).

– STREAM(R, 0, Node-Constrained-Query)

Required Edge Resolution	Virtual Radius	Returned Edge Resolution		
		Average	*Maximum*	*Minimum*
0.8	6.38	0.710	0.715	0.706
0.6	9.03	0.556	0.561	0.546
0.4	11.93	0.424	0.427	0.421
0.2	20.45	0.179	0.188	0.175

TABLE II. PERFORMANCE OF EDGE CONSTRAINED QUERY ON A RANDOM 1000 NODE TOPOLOGY IN A 400X400M^2, R =40M.

– STREAM(r, 0, Exposure-Query)

When the parameter *Node-Constrained-Query* is passed, each black node finds its neighbor nodes, which have *not* been reported by its children black nodes. Out of these unreported neighbors, it picks *x%*, of nodes and aggregates with the children’s information before sending it upstream. The 2nd query is used to find the virtual backbone of a network. In the query-type *Exposure-Query,* each node just returns its location. The number of nodes responding is controlled by virtual range *r.* As has been observed in [13], the minimum exposure path does not change significantly after a certain node density. STREAM can return the minimal number of nodes to calculate the exposure with desired precision.

Table 1 gives the performance of node-constrained query using STREAM for the different required resolutions in the simulation setup described earlier.

B. Edge Constrained Query:

The form of an edge-constrained query is: *”Return x% of the edges in the network”.* This type of query is useful in determining the connectivity properties of the network. We evaluate properties such as shortest paths lengths, number of edge, node disjoint paths, etc. for different resolutions of topology retrieved.

To extract topology at required resolution, we have to find the appropriate values for *virtual range r*. The *resolution-factor* is set to 1. Note that setting resolution-factor to 1 ensures that at least one edge pertaining to each node is reported back. *Let,*

d = average node degree

E= expected total number of edges in the topology.

$$d = \frac{N\pi R^2}{A} - 1 \quad and \quad E = \frac{Nd}{2},$$

Since every black node reports all edges originating from it, the total number of edges reported back only depends on the number of black nodes chosen to reply back. To get the required fraction of edges (x) the following condition must be satisfied.

$$\frac{Bd}{2} = xE \qquad \Rightarrow B = \frac{2xE}{d}$$

And also from equation 5,

$$B = pN \Rightarrow p = \frac{B}{N} = x$$

This gives the virtual radius as follows:

$$r = \sqrt{\frac{A}{N\pi}\left(1 + \frac{\log(x)}{\log(1-x)}\right)} \quad \text{... 13}$$

Now for equation 12 to be consistent with Equation 5, the expected number of nodes in virtual range should be greater than 1. Thus r is valid only if:

$\frac{\log(p)}{\log(1-p)} > 1$, Otherwise r is given by equation 6, as:

$$r = \sqrt{\frac{4A(1-p)}{N\pi}} \quad \text{... 14}$$

The topology discovery query takes the following form:

– STREAM(r, 1, Edge-Constrained-Query).

The edge-constrained query in essence is central to the multi resolution topology extraction problem we are trying to solve. The overhead for this type of query illustrates the exact nature of trade-off between the resolution of topology retrieved and communication overhead expended. From equation 9, we see that to get *x%* of the edges the overhead is:

$$O_{Total} = N + Nx\left(2 - x\frac{R^r - r^2}{R^2}\right) \quad \text{... 15}$$

This is because we get p equal to x and find the value of r using equation 13. Thus we see that the highest resolution topology takes a packet overhead of *2N*, and the lowest resolution topology takes the packet overhead of *N(1+2p)* where *2Np* is the total number of nodes which are black or intermediates of black for virtual range equal to the communication range.

The edge-constrained query also illustrates the convergence properties of the multi-resolution network graph. Let us consider the average deviation in hops for shortest paths from monitoring node to every other node as compared to the actual values in the network graph. From proposition 2 in section A we know that every black node and intermediate node is optimal number of hops away from the sink in the retrieved graph. All other nodes are at most two hops away in the retrieved graph since they are always neighbor of some black node. Hence the average deviation is bounded by the number of blue nodes which are not intermediate nodes. For required edge resolution x, this is given by:

$$N_{blue} = N - Nx\left(2 - x\frac{R^r - r^2}{R^2}\right) \quad \text{... 16}$$

$$H_{deviation}(x) = 2\left(1 - x\left(2 - x\frac{R^r - r^2}{R^2}\right)\right) \quad \text{... 17}$$

Thus if we retrieve *x%* of the edges using edge-constrained query the average deviation is bounded by equation 17. Thus we see that STREAM can give guarantees on hop deviation and 100% node coverage which is not possible with probabilistic variations. Moreover the overhead incurred is lesser which makes STREAM suited for sensor networks.

Table 2 shows the mapping of required edge-resolution to the value of parameter r for the simulation setup described at the beginning of the section IV. Using the above queries, we see that STREAM is able to retrieve edges at resolution close to the desired values. Figure 12 showed that multi-resolution topology is retrieved at proportionate cost.

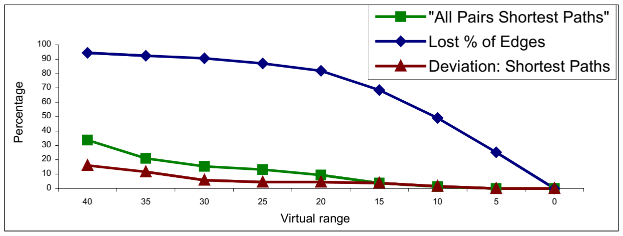

Figure 14: Effect of Edge Resolution: Deviation from optimal All-pairs and single source shortest path lengths, (N=1000, R=40, field size 400x400 m^2)

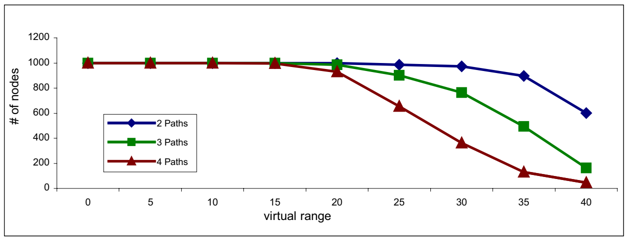

Figure 15: Effect of Edge Resolution on number of Node Disjoint Paths to Sink. (N=1000, R=40, field size 400x400 m^2)

Figure 14 shows the effect of edge resolution on the length of single source shortest path lengths (rooted at initiating node) and all-pairs shortest path lengths. The graph shows the relative degradation in the single source shortest path lengths (SSSP) and all pair shortest path lengths as the topology resolution increases. We see that even for low resolutions, paths do not deviate significantly from the optimal. Figure 15 shows the number of nodes which have 2, 3 and 4 node and edge disjoint paths for different resolutions.

We note that the savings in topology discovery increases as the density of the graph increases, i.e., lesser percentage of edges is required as density increases for similar deviation from optimal behavior. Also as density increases lesser percentage of nodes become black due to which the relative message complexity decreases. In general, since sensor networks are expected to be dense, we can attain significant savings by discovering topology at the required resolution, using STREAM.

C. *Overhead Constrained Query*

This query has the form: *"Return the best resolution topology using a given Overhead budget."* The total overhead in topology discovery is due to request forwarding (network-wide flooding) and the discovery acknowledgement. Probabilistic forwarding may be required to cut down on the energy spent on the forwarding phase. To support this query, we use the analytical results on expected retrieved resolution and the overhead for given values of parameters r and f (Figure 8 and Figure 11).

First we consider the packet overhead of the algorithm. For the initial phase of the algorithm, since all nodes forward the topology discovery request at most once, the overhead is *N*. During the acknowledgement phase, a subset of nodes are involved namely the black nodes and the intermediate nodes between two black nodes. For a given virtual radius, the expected number of black nodes is given by equation 5.

To compute the expected number of intermediate nodes, we consider the probability of the intermediate node to be also black. Consider Figure 9 with a black node *a* and an intermediate node *b*. We see that *b* is black with probability *p* only if it falls outside the virtual range of *a.* Thus the probability *P* that an intermediate node is *black* is give by:

$$P = p\frac{R^2 - r^2}{R^2} \qquad \text{... 8}$$

Hence the total communication overhead of the acknowledgement phase is equal to the expected number of black nodes plus expected number of intermediate nodes which are blue. Since each black node except the monitoring node has one intermediate node, Hence total expected overhead is given by equation 8.

If O_M = Maximum bytes allowable,

$O_T(r, f)$ = Total overhead for a given r and f.

$Re(r, f)$ = Retrieved resolution for given r, f.

Then we have to find a pair *r, f*, which satisfies the following:

$$\begin{aligned} &Max(\mathrm{Re}(r,f)) \\ &s.t.\ \ O_T(r,f) \le O_M \\ &0 \le f \le 1, \qquad 0 \le r \le R \end{aligned} \qquad \text{... 18}$$

The *iso-overhead* contours in Figure 8 provide a range of parameter values which return topology at a given overhead. The contour for the given query is computed by intersecting the surface O_T with the hyper-plane defined by equation 14. In general, the analytical computation of parameters by this method is not required if we assume uniform random graphs. Setting *resolution-factor* to 1 and then selecting the minimum possible virtual range for the given overhead would almost always retrieve the maximum resolution. This is because by setting *f=1,* we are maximizing the amount of information in each packet and thus minimizing the share of constant packet overhead in the total overhead. However, other considerations may come into account when selecting the parameter values as discussed at the beginning of the section.

Similarly, given a required topology resolution, to select parameters such that overhead is minimized is done as follows. We choose the largest possible virtual range and with resolution factor equal to 1. The underlying intuition here is that the average number of hops traveled by packets is nearly the same for all parameter values. Thus to reduce the overhead, the number of packets has to be reduced.

V. STREAM UNDER ARBITRARY NETWORK CONDITIONS

In this section we analyze the performance of STREAM under arbitrary network conditions involving non-uniform topologies, sleeping nodes, channel error rates and node failures. We also propose methods which makes STREAM adaptive to these varying network conditions.

Required Edge Resolution	Edge Resolution on Topology 1 assuming uniform	Returned edge resolution with local scheme	Edge Resolution on Topology 2 assuming uniform	Returned edge resolution with local scheme
0.8	0.66	0.706	0.503	0.71
0.6	0.497	0.56	0.353	0.56
0.4	0.38	0.411	0.23	0.43
0.2	0.160	0.184	0.087	0.208

TABLE III. PERFORMANCE OF EDGE CONSTRAINED QUERY ON NON UNIFORM TOPOLOGIES OF 1000 NODES IN A 400X400M^2, R =40M.

A. Non-Uniform Topology

We had assumed the sensor network topology to be comprised of uniformly distributed nodes. Although this was useful to analyze the behavior of STREAM, practical sensor networks may not be uniform and we may not have any information about the underlying distribution. For STREAM to be useful in practice, it should be able to function properly even without knowledge of node distribution.

Consider the network to have any arbitrary distribution where a node *i* has some degree d_i whose distribution is not uniform across the network. The key to getting the desired resolution in a uniformly distributed network was that having a constant virtual range across the network sufficed. When the topology is non-uniform, each node has to compute its own virtual range based on the local network density and node degree.

Let x = required edge resolution.

Consider the circular range of the i^{th} node with d_i neighbors. Let p_i be the probability of nodes inside this region to become black.

Expected number of black nodes in the region = p_i d_i.

Assuming that node distribution is uniform *inside the communication range*[8], the number of edges reported for this region should be *a fraction, x* of the total expected edges in this region. Hence, we have to choose a virtual range so that p_i=*x.* Let the virtual range for the i^{th} node be r_i.

$$r_i = \sqrt{\frac{R^2}{d_i}\left(1 + \frac{\log(x)}{\log(1-x)}\right)} \qquad \text{... 19}$$

And for $\dfrac{\log(p)}{\log(1-p)} < 1$, we have

$$r_i = \sqrt{\frac{4R^r(1-x)}{d_i}} \qquad \text{... 20}$$

Thus every node computes its own virtual radius according to its local information. The coloring scheme and the algorithm remain the same while using these local values of virtual range.

We evaluate this scheme using two different non-uniform topologies. In the first one we consider a square field of dimension 400x400 m^2 and divide it up in four quadrants of size 200x200 m^2. We put 100, 200, 300 and 400 nodes in the

[8] While the node distribution across the network is non-uniform, the distribution inside a node's communication range can be considered to be uniform when the range is much smaller than the network dimensions.

first, second, third and the fourth quadrant respectively resulting in different densities in each quadrant. In the second case we generate a gaussian distribution of 1000 nodes around the center of a 400x400 m^2 field and variance 75m.

Table 3 shows the results of performing edge-constrained queries of the two non-uniform topologies. We see that by assuming the topology to be uniform and computing a single global value of virtual range results in large deviations in the returned edge resolution compared to the required edge resolution. By using local virtual ranges the returned resolution is very near to the required resolutions and performs as good as with the uniform topology case.

B. *Impact of Sleeping Nodes*

One of the distinguishing features of sensor networks is that nodes would sleep periodically. We compute the fraction of sleeping nodes that would be reported by the responding active nodes. Note that active nodes cache data about its neighbors and would respond with information about sleeping nodes as well. This discussion assumes that the network of active nodes is connected. Let,

x=fraction of nodes sleeping.

(1-x) N = N' is the number of active nodes.

d = average degree of each node.

p =probability of node to become black (equation 5)

p_s=probability of a sleeping node being reported

b_x=expected black neighbors of each node:

$$b_x = p\sum_{i=1}^{d}\binom{d}{i}\, i(1-x)^i(x)^{d-i} \qquad \text{... 21}$$

Each active node is always reported because the network of active nodes is assumed to be connected. The probability of a sleeping node being reported is the probability of that any one of its b_x black neighbors report it. Since black nodes report *f fraction* of their edges, the probability is given by:

$$p_x = \begin{bmatrix} 1-(1-f)^{b_x} & b_x \ge 1 \\ b_x f & b_x < 1 \end{bmatrix} \qquad \text{... 22}$$

We tested the performance of STREAM with varying percentage of sleeping nodes. The simulations are set up for with the parameter values *r=R* and *f=1*. The results in Figure 16 show that STREAM is able to retrieve information about nearly all the nodes if the number of active neighbors of each node is around *8*. The results also imply that if the active node set is connected, then we get information about almost all nodes.

C. *Channel Errors and Node failures.*

Sensor networks are envisioned to be deployed in hostile territories and hence would have significant channel errors and node failures. To make our proposed algorithm useful in practical scenarios, we have to ensure that it is fault tolerant to these errors. We first discuss the effect channel errors and node failures during the request propagation phase.

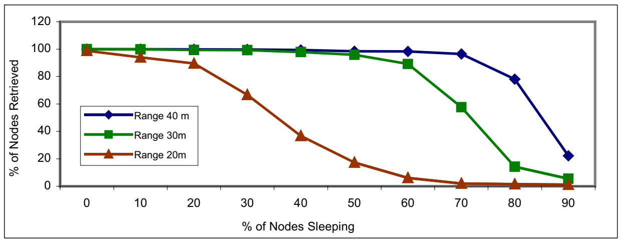

Figure 16: Effect of sleeping nodes on the Retrieved Topology. The network consists of 1000 node in a field of 400x400m^2. Simulations are carried out for three different communication ranges with varying % of nodes sleeping.

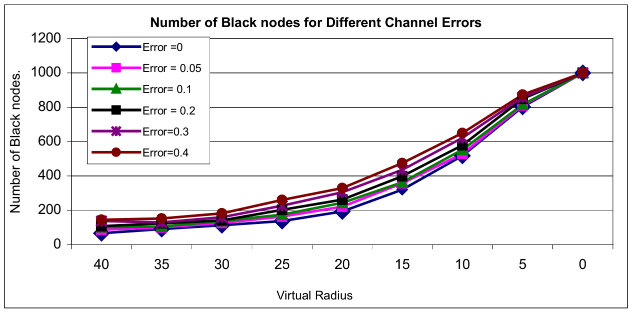

Figure 17: The number of black nodes formed for different channel error rates. 1000 node deplyed in 400x400 m^2 field with R=40m.

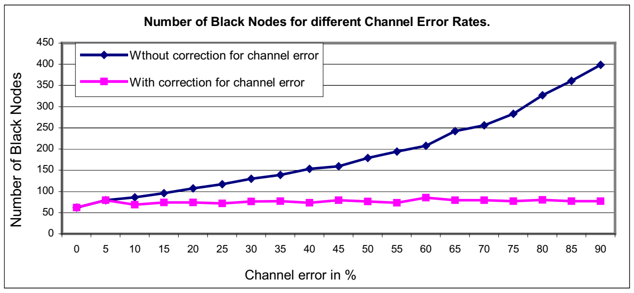

Figure 18: The number of black nodes formed for different channel error rates. 1000 node deplyed in 400x400 m^2 field with R=40m, and r=40.

Channel errors affect the request propagation phase in the following ways:

- *Increase in number of black nodes:* Since packets may be lost when a black node forwards a request, a node may become black even if it is inside the virtual range.
- *Proposition 1 and 2 are no longer valid.* This means that request divergence may not have a wave like propagation.
- *Timers may not work*: since nodes do not receive a topology discovery request packet in optimal hops.
- All nodes may not receive a request packet. However for dense networks, it is negligible even for high channel errors.

Figure 17 shows the effect of channel errors on the number of black nodes. A node becomes black if no other node in its virtual range is black or if packets from one or more black nodes in its virtual range are lost due to channel errors. Thus equation 2 can be modified to incorporate the channel errors to get the analytical model for number of black nodes:

$$p = \sum_{i=0}^{d} e^i p^i (1-p)^n \qquad \text{... 23}$$

Figure 19 gives the average hop deviation from the optimal for received request packets. One may choose a value or virtual radius according to equation 23 to get required resolution. We may also select timers such that it incorporates the expected deviation related to the error rate. However computing the values involves numerical solution of the equations involved and may be too complex for sensor nodes. We choose a simple method by which the number of black nodes and hop deviation is reduced significantly.

We see in Figure 17 and 19 that increase in number black nodes and hop deviation is negligible for channel error rates lesser than 10% for the topology scenario considered. This means that at such channel error rates propositions 1 and 2 are valid with high probability and equation 5 is a good approximation to the number of black nodes formed.

Let $0 < c < 0.1$ be some channel error rate for which the effect is negligible. Suppose we send multiple copies of the request packet per node, neighbor nodes would receive the packet with higher probability because of this redundancy. I.e. the apparent channel error would be lower. In particular if the apparent channel error is reduced such that it is close to *c,* then the number of copies *m* required for this is given by:

$$m = \frac{\log(c)}{\log(e)} \qquad \text{... 24}$$

However if each node sends *m* request packets instead of 1 the overhead would increase significantly. Instead if a node sends *m* copies only if becomes a black node, overhead is significantly lowered since black nodes are much lesser in number. We can use the scheme of only black nodes forwarding multiple packets if it is able to reduce the effect of channel errors significantly.

We assume that nodes have knowledge of their local channel error rates. We use $c=0.05$ to compute the value of *m.* In Figures 18 and 19 we show the effect of this scheme on the number of black nodes and hop deviation for different channel error rates. We see that the number of black nodes formed is stabilized even for channel error rates as high as 90%. The average hop deviation is also reduced by this method. In particular the hop deviation for black nodes is significantly reduced.

Since we are able to stabilize the number of black nodes and average hop deviation for black nodes for every channel error rates, the analysis in section 3 is a good approximation for expected number of black nodes, retrieved resolution of topology and some graph properties. Hence the parameters for edge and node constrained queries can be mapped as described in section 4. The bounds on the properties of retrieved topology are also maintained with high probability. Since *m* increases sub-linearly with increase in *e* the overhead increases sub-linearly if the number of black nodes is small compared to the total number of nodes in the network.

The effect of node failures during the request propagation phase is equivalent to the effect of sleeping nodes. I.e. nodes do not participate in the topology discovery if they are dead or sleeping. However a node may die in between the topology request phase and the acknowledgement phase.

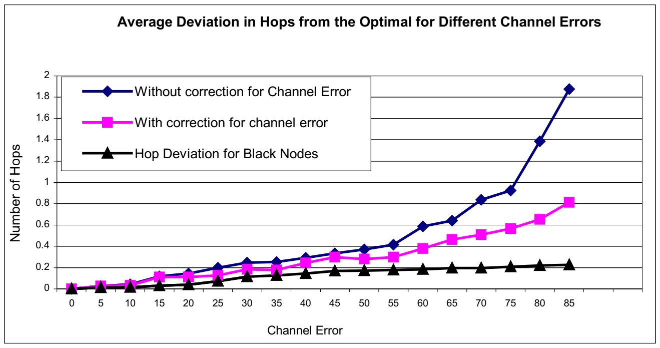

Figure 19: The average hop deviation of the received request packets for different channel error rates. 1000 node deplyed in 400x400 m^2 field with R=40m, and r=40.

During the request propagation phase, black nodes sends back their aggregated topology information upstream. Due to channel errors topology information is lost if an acknowledgement packet is lost. Similarly if any node in the black node tree dies, the information from its sub-tree would not reach the sink. A *passive acknowledgement scheme* from DSR [10] is used to account for channel errors. We utilize the high density of sensor networks to account for node failures. These are described as next.

After a black node sends its aggregated edge information to the next hop to its parent black node it eavesdrops on the communication channel to see if the next hop node has forwarded its packet. If it does not hear within a certain time period it forwards the packet again.

Since the next hop may be dead, the number of retransmissions cannot be allowed to go on indefinitely. We have to limit the number of retransmissions. From equation 24, we know that if a packet has been retransmitted *m* times, then with probability *1-c* it should have reached the next hop. If the black node does not receive the passive acknowledgement after *m* tries we may infer with high confidence that the next hop node is dead.

After *m* retransmissions, a black node selects another node from its neighborhood with a gradient towards the sink (if the black node is *h* hops away from the sink, it selects a node which is *h-1* hops away from the sink). We note that since the network density is high there are multiple such nodes with high probability. Each node just has to keep that information which it gathers during the request propagation phase.

Let *k* be the number of neighbors of a node with a gradient towards the sink. If the node failure rate is *f* and local channel error rate is *e,* the probability that a packet can be forwarded successfully to a node next hop is given by:

$$P(success) = 1 - \left(1 - (1-e)(1-f)\right)^{mk} \qquad \text{... 25}$$

We see that by the above method information from active black nodes can be sent to the sink with high probability. For example with *e=0.3, f=0.1*, *m=3 and k=2,* gives *P(success)= 0.000378.* Thus we see that even when we choose a maximum retransmission number of 3 with only two nodes next hop, packets from downstream are lost with very low probability.

Virtual range	Expected resolution to be retrieved	Retrieved Edge Resolution, *e=0.15,f=0.05*	# of nodes Retrieved *e=0.15, f=0.05*	Retrieved Edge Resolution, *e=0.3,f=0.1*	# of nodes Retrieved *e=0.3, f=0.01*
40	0.07	0.16	982.46	0.182	978.952
30	0.15	0.262	980..5	0.28	963.047
20	0.219	0.304	995.57	0.326	994.5
10	0.62	0.706	999.95	0.731	999.578
5	0.81	0.859	999.71	0.91	999.6

TABLE IV. PERFORMANCE OF EDGE CONSTRAINED QUERY WITH DIFFERENT CHANNEL ERROR AND NODE FAILURE RATE, 1000 NODES IN A 400X400M^2, R =40M.

However the neighborhood information of a dead black node is lost. To compensate for this loss, when an intermediate node infers after *m* transmissions that its black parent is dead, it adds its own neighborhood list to a packet when forwarding it upstream. Thus we are able to retrieve information about nodes which are in range of both the black node and the intermediate node. The only problem with this method is that there is no way to compensate for the death of a node in the last hop.

We have to also take care of the timers for nodes which forward when some node in the black node tree dies. Without the timers, we cannot guarantee that each node will forward request packets at most once and if at all it does, it is able to aggregate all packets from downstream. Without channel errors and node failures only the black nodes needed to maintain the request acknowledgement timers. However, after receiving the topology request if each node starts an acknowledgement timer in case it is required, the layered aggregation semantics can still be maintained. Thus we have to incur extra overhead at each node to reduce communication overhead, to account for channel errors and node failures. However since the main resource constraint in sensor networks is energy, this is still favorable.

This completes the description of our algorithm. We reexamine the overhead of our algorithm based on the channel errors and node failures. Assuming uniform topology, the expected virtual radius at each node would be approximately equal to the one computed from equation from equation 19 and 20. Also from Figure 18 we see that the expected number of black nodes is constant for a given virtual range for different channel errors.

Let, e = channel error rate. f = node failure rate and x = required edge resolution. Then the expected number of black nodes = x. m is computed using equation 24. Assuming all nodes are active during the request propagation phase overhead is given by:

$$O_{request} = N(1-x) + Nxm \qquad \text{... 26}$$

During the acknowledgement phase, we know that there are maximum of *m* retransmissions and maximum of *k* nodes next hop. Thus each node in the acknowledgement tree successfully forwards a packet with very high probability with a maximum overhead of *mk* packets. We know even if a black nodes die, there is another node which would forward the packets from downstream Thus the total number of nodes forwarding a packet remains the same as before but each may forward multiple copies of a packet. Hence overhead of the acknowledgement phase is given by:

$$O_{request} = (Nfx + I)mk \qquad \text{... 27}$$

Table 4 gives the performance of the edge constrained query for two pairs of channel error and node failure rate. In the first case we look at mild conditions of *15%* channel error and *5%* node failure. In the second case we look at an adverse condition of 30% channel error and *10%* node failure rate. The node failure rate is a relatively high number since it is unlikely that *10%* of the nodes would die within a time period of topology discovery. We see that retrieved edge resolution is always higher than the expected resolution although the average number of nodes which are included in the retrieved topology is much lesser than 1000. This is because with high failure rate, many black nodes die. To compensate for this, other intermediate nodes insert their own neighborhood lists. Thus although the edge resolution increases, the results mean that the edges are not distributed evenly among nodes.

We also see that at low resolution there is a large difference in the expected topology and the retrieved topology and the difference reduces for higher resolutions. This is because at higher resolutions percentage of blue intermediate nodes is very low and the extra edges added by them correctly compensates for the missing information due to dead black nodes. This result is also favorable for sensor networks since we would ideally like to have better control on retrieved as resolution increases because retrieved overhead directly effects the expended overhead of the process.

VI. RELATED WORK

Researchers have proposed different mechanisms for topology discovery of data networks [3],[7],[8],[9] primarily using probing techniques. Using routing tables for aggregating topology information is not feasible because traditional routing tables may not be available if data centric model of routing is used [5]. Further, in ad-hoc deployments, routing tables are often inaccurate or incomplete. Probing techniques used for Internet mapping [3] is not possible in sensor networks since nodes operate in various levels of doze mode, may often be disconnected and the incurred overhead is unsuitable for energy constrained sensor networks.

In [26], given a set of network endpoints, end-to-end Bayesian probing is used to infer properties of IP networks using correlations. Our work focuses on *selecting a representative set* of nodes in a sensor networks to estimate network properties.

In [15] authors propose a mobile agent based framework to distribute topology information. Here, the optimal number of agents required is half the number of nodes. The overhead with this approach would not scale for sensor networks where the number of nodes is large. Reducing the agents would increase the expected time that any agent takes to reach the initiating node.

References [18],[19] introduced the concept of virtual backbones for wireless networks. References [17], [18], [19], [20], [21], [22], [23] describe routing in ad hoc networks using minimal connected dominating sets. Using each of the above methods, a simple topology discovery algorithm can be designed to query the dominating nodes, which provide their neighborhood lists. However, our algorithm differs significantly from the above in its *multi-resolution* nature.

Hence, the topology returned is not limited to the minimal backbone. In fact, the topology that each of the above can provide is only a special case of lowest resolution topology recovered by our algorithm.

In [28] a conceptual framework called DIMENSIONS was introduced for multi-resolution data-access in sensor networks. STREAM provides a framework to spatially sample a sensor network to select a subset of representative nodes for the network at a given resolution. The algorithm proposed in [28] focus compression of data at each node using spatial and temporal correlations and maybe used in conjunction with STREAM after using STREAM to first select the subset.

VII. CONCLUTIONS AND FUTURE WORK

In this paper, we have described an algorithm for sensor topology extraction at multiple resolutions (STREAM). STREAM may also be used as a general-purpose multi resolution information retrieval algorithm. The main issues are generalizations of the sampling parameters to select the subset.

For topology discovery the node selection process finds a suitable MVDS. In general, the node selection could depend on any characteristic of the information sought. For example, if the desired information has spatial gradient and neighborhood correlations, then such information can determine the *virtual-range.*

The parameters *resolution factor query-type* and determines the filtering and compression of neighborhood done at the sampled node set. The *resolution-factor* determines the number of neighbors about which a reporting node responds. Each node can collect information about its neighboring nodes by eavesdropping. The responding node can choose a fraction *f* of these neighbors and apply the filter specified by the query-type parameter. Even, in case of topology discovery a more sophisticated aggregation scheme as proposed in [16] could be used. A different type of aggregation could be used for energy information as proposed in [4]. Finally use of schemes like smart messages as proposed in [24] which would carry specific code along with the query to support infrequent queries could save valuable memory resources in sensors.

STREAM opens up a wide design space for multi-resolution information retrieval. This would require knowledge of the properties of queries and the corresponding aggregation/filter functions to compress that property. We intend to explore this design space for different types of information in future.

REFERENCES

[1] J.M. Kahn, R.H. Katz, K.S.J. Pister, *Next century challenges: Mobile networking for 'smart dust*', Proc. MOBICOM, 1999, Seattle, 271-278.

[2] G.J. Pottie and W.J. Kaiser, *Wireless Integrated Network Semsors.* Communications of the ACM, vol. 43 no. 5 (2001), 51-58.

[3] A. Downey, "*Using pathchar to estimate Internet link characteristics*, Proc. SIGCOMM 1999, Cambridge, MA, pp. 241-250, Sept. 1999.

[4] Jerry Zhao, Ramesh Govindan and Deborah Estrin, "Residual Energy Scans for Monitoring Wireless Sensor Networks"*, IEEE Wireless Communications and Networking Conference, 2002.*

[5] C. Intanagonwiwat, R. Govindan and D. Estrin; *Directed diffusion: a scalable and robust communication paradigm for sensor networks*; in Proceedings of Mobicom '00, 2000.

[6] Cormen, Leiserson, Rivest (1990) *Introduction to Algorithms*, MIT Press, McGraw Hill.

[7] Bruce Lowekamp, David R. O'Hallaron, and Thomas R. Gross, *"Topology Discovery for Large Ethernet Networks,"* In Proceedings of ACM SIGCOMM August 2001 (San Diego, California), ACM Press,.

[8] Y. Breitbart, M. Garofalakis, C. Martin, R. Rastogi, S. Seshadri and A. Silberschatz. *" Topology Discovery in Heterogeneous IP Networks"* In Proceedings of IEEE INFOCOM, 2000.

[9] Nancy Miller and Peter Steenkiste, *"Collecting Network Status Information for Network-Aware Applications"*In Proceedings of IEEE INFOCOM 2000.

[10] David B Johnson and David A Maltz, *"Dynamic Source Routing in Ad Hoc Wireless Networks",* Mobile Computing, Volume 353, Kluwer Academic Publishers, Edited by Imielinski and Korth.

[11] Seapahn Meguerdichian, Farinaz Koushanfar, Gang Qu, Miodrag Potkonjak, *"Exposure In Wireless Ad Hoc Sensor Networks."* Proc.. of 7th Annual International Conference on Mobile Computing and Networking, MOBICOM 2001.

[12] Seapahn Meguerdichian, Farinaz Koushanfar, Miodrag Potkonjak, Mani Srivastava, *"Coverage Problems in Wireless Ad-Hoc Sensor Networks,"*In Proc. of IEEE INFOCOM 2001.

[13] Bhaskar Krishnamachari, Stephen B. Wicker, and Ramon Bejar, *"Phase Transition Phenomenon in Wireless Ad hoc Networks", Symposium on Ad-Hoc Wireless Networks, GlobeCom2001*, San Antonio, Texas, November 2001.

[14] L. Li, Z. Haas and J.Y. Halpern*,"Gossip-Based Ad Hoc Routing", IEEE INFOCOM,* June 2002.

[15] Romit RoyChouldhury, S. Bandyopadhyay and Krishna Paul*, "A Distributed Mechanism for Topology Discovery in Ad hoc Wireless Networks using Mobile Agents",* Proc. of Workshop On Mobile Ad Hoc Networking & Computing (MOBIHOC 2000).

[16] Baruch Awerbuch and Yuval Shavitt, "*Topology Aggregation for Directed Graphs.",* IEEE/ACM Transactions on Networking, Vol.9, N0.1, Feb 2001.

[17] B. Das, V Bhargavan, *"Routing in Ad Hoc Networks Using Minimum connected dominating Sets".* IEEE International Conference on Communications (ICC '97), Jun, 1997.

[18] A. Ephremides, J.E. Wieselthier, and D.J. Baker, "A design concept for trliable mobile radio networks with frequency hopping signaling." Proceeding of IEEE, 75, 1 (Jan 1987) 56-73.

[19] M. Gerla, J.T. C. Tsai, *"Multicluster, mobile, multimedia radio networks,"* ACM J. Wireless Networks , 1, 3 (1995) 255-265.

[20] S. Guha and S. Khuller, "*Approximation Algorithms for Connected Dominating Sets,"* European Symposium on Algorithms. 179-193, 1996.

[21] P. Sinha, R. Sivakumar and V. Bharghavan, "*CEDAR: a Core-Extraction Distributed Ad hoc Routing Algorithm"* In proceedings of IEEE Infocom 1999.

[22] Suman Bannerjee, Samir Khuller, "*A Clustering Scheme for Hierarchical Control in Wireless Networks",* In Proceedings of IEEE INFOCOM 2001.

[23] B. Das, R. Sivakumar and V. Bharghavan. "*Routing in ad hoc networks using a virtual backbone*." In Proc. IEEE (IC3N'97).

[24] Cristian Borcea, Deepa Iyer, Porlin Kang, Akhilesh Saxena, Liviu Iftode, *"Cooperative Computing for Distributed Embedded Systems"*, International Conference of Distributed Computing Systems, 2002.

[25] B. Clark , C. Colbourn and D. Johnson, *"Unit disk graphs",* Discrete Mathematics, vol. 86, pp. 165--177, 1990.

[26] A. Bestavros and J. Byers and K. Harfoush, *"Inference and Labeling of Metric-Induced Network Topologies"* In Proceedings of Infocom'02: The IEEE ICCC*, June 2002.*

[27] J.E. Wieselthier, G.D. Nguyen, and A. Epremedes, *"On the construction of energy-efficient broadcast and multicast trees in wireless networks"* In IEEE INFOCOM 2000, Tel Aviv, Israel 2000.

[28] Deepak Ganesan and Deborah Estrin, "*DIMENSIONS:Why do we need a new Data Handling architecture for Sensor Networks"* , In Proceedings of the ACM Workshop on Hot Topics in Networks, HotNets-2002.

[29] Yuan-Chieh Cheng and Thomas G. Robertazzi, "*Critical Connectivity Phenomenon in Mulithop Radio Models",* In IEEE Transactions on Communications, Vol. 37, No. 7, July 1989