Algorithms for Fault-Tolerant Topology in Heterogeneous Wireless Sensor Networks *

Mihaela Cardei, Shuhui Yang, and Jie Wu
Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431, USA
E-mail: {mihaela@cse., jie@cse., syang1@}fau.edu

Abstract

This paper addresses fault-tolerant topology control in a heterogeneous wireless sensor network consisting of several resource-rich supernodes, used for data relaying, and a large number of energy-constrained wireless sensor nodes. We introduce the k-degree Anycast Topology Control (k-ATC) problem with the objective of selecting each sensor's transmission range such that each sensor is k-vertex supernode connected and the total power consumed by sensors is minimized. Such topologies are needed for applications that support sensor data reporting even in the event of failures of up to $k-1$ sensor nodes. We propose three solutions for the k-ATC problem: a k-approximation algorithm, a greedy centralized algorithm that minimizes the maximum transmission range between all sensors, and a distributed and localized algorithm that incrementally adjusts sensors' transmission range such that the k-vertex supernode connectivity requirement is met. Extended simulation results are presented to verify our approaches.

Keywords: Energy efficiency, fault tolerance, heterogeneous wireless sensor networks, topology control.

*This work was supported in part by NSF grants CCF 0545488, CNS 0422762, CNS 521410, CCR 0329741, CNS 0434533, CNS 0531410, and CNS 0626240.

1 Introduction

In this paper, we address topology control in heterogeneous wireless sensor networks (WSNs) consisting of two types of wireless devices: resource-constrained wireless sensor nodes deployed randomly in large numbers and a much smaller number of resource-rich supernodes placed at known locations. The supernodes have two transceivers, one to connect to the wireless sensor network, and another to connect to the supernode network. The supernode network provides better QoS and is used to quickly forward sensor data packets to the user. With this setting, data gathering in heterogeneous WSNs has two steps. First, sensor nodes transmit and relay measurements on multihop paths towards any supernode (see Figure 1). Then, once a data packet encounters a supernode, it is forwarded using fast supernode-to-supernode communication toward the user application. Additionally, supernodes could process sensor data before forwarding.

A study by Intel [14] shows that using a heterogeneous architecture results in improved network performance, such as a lower data-gathering delay and a longer network lifetime. Hardware components of the heterogeneous WSNs are now commercially available [6].

We model topology control as a range assignment problem for which the communication range of each sensor node must be computed. The objective is to minimize the total transmission power for all sensors while maintaining k-vertex disjoint communication paths from each sensor to the set of supernodes. In this way, the network can tolerate the failure of up to $k-1$ sensor nodes. In contrast with range assignment in ad hoc wireless networks, this problem is not concerned with connectivity between any two nodes. Our problem is specifically tailored to heterogeneous WSNs, in which data is forwarded from sensors to supernodes.

The contributions of this paper are the following: (1) we formulate the k-degree Anycast Topology Control (k-ATC) problem for heterogeneous WSNs, (2) we propose three solutions for solving the k-ATC problem: a k-approximation algorithm, a centralized greedy algorithm that minimizes the sensor maximum transmission range, and a distributed and localized algorithm, and (3) we analyze the performance of these algorithms through simulations.

The rest of this paper is organized as follows. In Section 2 we present related works on fault-tolerant topology control problems. Section 3 describes the heterogeneous WSN architecture, the network model, and introduces the k-ATC problem. We continue in Section 4 with our solutions for solving the k-ATC problem. Section 5 presents the simulation results, and Section 6 concludes our paper.

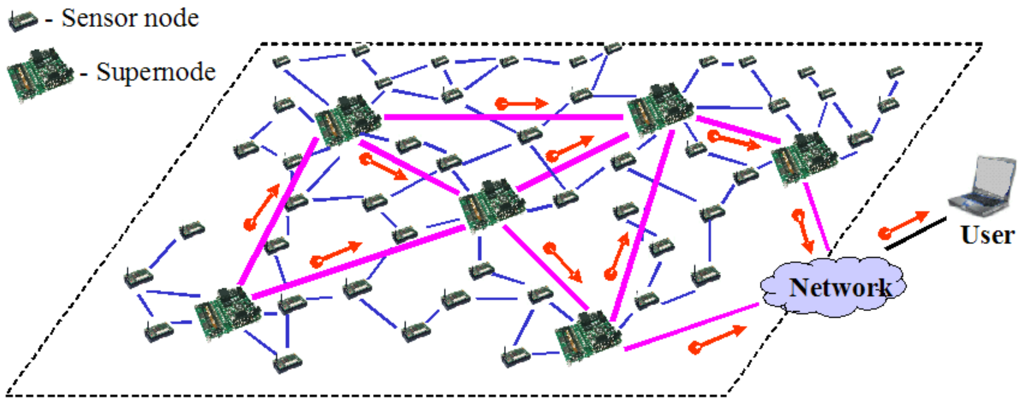


Figure 1: Heterogeneous WSNs.

2 Related Work

The benefits of using heterogeneous WSNs, containing devices with different capabilities, have been presented recently in literature. In [25] it is reported that when properly deployed heterogeneity can triple the average delivery rate and provide a 5-fold increase in the network lifetime.

The work in [19] introduces another type of heterogeneous WSNs called *actor networks*, consisting of sensor nodes and actor nodes. The role of actor nodes is to collect sensor data and perform appropriate actions. This paper presents an event-based coordination framework using linear programming and a distributed solution with an adaptive mechanism to trade off energy consumption for delay, when event data has to be delivered within a specific latency bound.

The majority of the existing work in fault-tolerant topology control studies the k-vertex connectivity, requiring the existence of k-vertex disjoint paths between any two nodes in the network. Such a requirement is more appropriate for ad hoc wireless networks, where any two nodes can be a source and a destination. In WSNs, data is transmitted from sensors to the sink(s), so maintaining a specific degree of fault-tolerance between any two sensors is not critical. However, it is rather important to have fault-tolerant data collection paths between sensors and sink(s) (or supernodes in our case).

A considerable amount of work ([1, 2, 11, 15] and [17]) has been done on the fault-tolerant topology control problem with the objective of minimizing the total power consumption while providing k-vertex connectivity between any two vertices. The majority of these algorithms are centralized and they propose approximation algorithms for various topologies. Calinescu et al. [2] propose an algorithm with a performance ratio of 4 for the 2-connectivity problem. Jia et al.

[15] propose a $3k$-approximation algorithm, $k \geq 3$, by first constructing the $(k-1)$th nearest neighbor graph and then augmenting it to k-connectivity by using one of the existing minimum edge weight k-connected algorithms. The Fault-Tolerant Cone-Based Topology Control (CBTC) algorithm proposed by Bahramgiri et al. [1] is a distributed and localized algorithm that achieves k-connectivity by having each vertex increase its transmission power until either the maximum angle between its two consecutive neighbors is at most $\frac{2\pi}{3k}$ or its maximal power is reached.

The works in [16] and [21] address the fault-tolerant topology control with the objective of minimizing the maximum power consumption. Ramanathan et al. [21] propose a centralized greedy algorithm for assuring biconnectivity ($k = 2$) that iteratively merges two biconnected components until only one remains. Li and Hou [16] introduce two algorithms for the k-connectivity problem, one centralized and the other distributed and localized. The algorithms examine edges in increasing order of their weight and select edges only if k-connectivity is not satisfied. These algorithms minimize the maximal power consumption between all k-vertex connected topologies.

There are also previous works addressing k-connectivity in a rooted graph. Frank and Tardos [8] study the k-connectivity from the root to any other node with the objective of minimizing the total weight of the edges. They propose a polynomial time optimal solution using a maximum cost submodular flow problem. Wang et al. [24] propose an approximation algorithm with ratio k for k-connectivity from any node to the root, and an approximation algorithm with ratio $O(n)$ for k-connectivity from the root to any node. However, these algorithms are centralized.

Our work differs from [1, 2, 11, 15, 16, 17, 21, 24] by considering a different architecture and a different topology objective:

- We consider a heterogeneous WSN architecture with multiple supernodes and are concerned with providing k-connectivity from each sensor to the set of supernodes.
- [1, 2, 11, 15, 16, 17, 21] consider a homogeneous architecture and have as objective k-connectivity between any two nodes.
- [24] uses a heterogeneous architecture with only one root (or supernode) and study k-connectivity from the root to any node.

We use the framework in [24] to design our first centralized algorithm $MWATC_k$, thus achieving a performance ratio k. Additionally, we propose a centralized algorithm $GATC_k$ that minimizes the maximum transmission range and a distributed and localized algorithm $DATC_k$ that is feasible for practical deployment of large scale WSNs.

3 Problem Definition and Network Model

3.1 Heterogeneous Network Architecture

For networks that contain a large number of sensors (e.g. thousands of sensor nodes) it becomes infeasible to network sensors using a flat network. As data is forwarded hop by hop to the sink, it becomes inefficient and unreliable to travel a long way in the WSN, depleting the energy of the sensors participating in data relaying.

A solution that has received increasing attention recently is the use of heterogeneous WSNs that contain devices with different hardware capabilities. Three common types of hardware heterogeneity are mentioned in [25]: computational heterogeneity, where some nodes have increased computational power, link heterogeneity, where some nodes have long-distance highly reliable communication links, and energy heterogeneity, where some nodes have unlimited energy resources.

One architecture which has been recently explored in literature contains two types of wireless devices, as presented in Figure 1. The lower layer is formed by sensor nodes with size and weight restrictions, low cost (projected to be less than $1), limited battery power, short transmission range, low data rate (up to several hundred Kbps), and low duty cycle. The main tasks performed by sensor nodes are sensing, data processing and data transmission/relaying. The dominant power consumer is the radio transceiver [20].

The upper layer consists of resource-rich supernodes overlaid on the sensor network, as illustrated in Figure 1. Supernodes can have two radio transceivers, one for communication with sensor nodes and the other for communication with other supernodes. Supernodes have more power reserves, and better processing and storage capabilities than sensor nodes. Wireless communication links between supernodes have considerably longer ranges and higher data rates, allowing the supernode network to bridge remote regions of the interest area. Supernodes are more expensive, and, therefore, fewer are used than sensor nodes. One of the main tasks performed by a supernode is to transmit/relay data from sensor nodes to/from the sinks. Other tasks can include sensor data aggregation, complex computations, and decision making. Recently, hardware platforms usable for supernode development have become commercially available [6].

Various research works refer to resource-rich supernodes with different names: *gateways* by Intel research [14], *masters* by the Tenet architecture [10], *microservers* by work [22], and *macronodes* by work [23]. Two practical implementations of heterogeneous WSNs in habitat monitoring experiments are described in [18, 23]. In [18], the experiment monitors seabird nesting environment and behavior in a small island off the coast of Maine, while [23] investigates task decomposition and collaboration in two-tiered heterogeneous WSNs consisting of sensor nodes used for data sampling and supernodes (or macronodes) used to run the algorithms for target classification and localization.

The presence of heterogeneous nodes in a sensor network increases network lifetime, and decreases the average end-to-end delay. In heterogeneous WSNs, data transmission from motes to the sink usually contains two steps. First, motes send data packets to supernodes and then supernodes send the packets to the sink. Network lifetime is improved since a smaller number of sensors are involved in forwarding a data packet, thus saving energy resources. The average end-to-end delay decreases since supernode network communication has a higher data rate and since a packet is forwarded fewer times. A detailed survey on heterogeneous WSNs is presented in [3].

3.2 Anycast Topology Control Problem

In this paper we consider a heterogeneous WSN consisting of sensors and supernodes. The supernodes are pre-deployed in the sensing area, they are connected, and their main task is to relay data from sensor nodes to the user application. On the other hand, sensor nodes are deployed randomly in the area of interest. We assume that sensor nodes can adjust their communication ranges up to a maximum value R_{max}. When each sensor is using a maximum transmission range R_{max}, there exist at least k paths from any sensor node to the set of supernodes.

Our goal is to provide a reliable data-gathering infrastructure from sensors to supernodes. We model this as the objective to establish the transmission range of each sensor such that: 1) there exist k vertex-disjoint communication paths from each sensor to supernodes, and, 2) the total power consumed by all the sensor nodes is minimized. In this paper we do not address the supernode-to-supernode communication.

The first condition is needed to guarantee that data from every sensor reaches at least one supernode when up to $k-1$ sensor nodes fail. The second condition is needed to ensure an energy-efficient design, which is an important requirement in WSNs. We assume that once a packet with data from a sensor reaches a supernode, it will be relayed to the user application using a separate, more capable and less resource-constrained supernode network.

In this paper, instead of assuring the connectivity between any two sensor nodes, we want to provide communication paths from each sensor to one or more supernodes. A sensor can communicate with another sensor or with a supernode if the Euclidean distance between nodes is less than or equal to the sensor's communication range. We consider the path loss communication model where the transmission power of a sensor n_i is $p_i = r_i^\alpha$ for a transmission range r_i, where the constant α is the power attenuation exponent, usually chosen between 2 and 4. Our algorithms can also be used for a more general power model $p_i = r_i^\alpha + c$, where c is a technology-dependent positive constant [13]. The formal definition is given below:

Definition 1 (k-degree Anycast Topology Control (k-ATC) Problem)
Given a heterogeneous WSN with M supernodes and N energy-constrained sensors that can adjust their transmission ranges up to a maximum value R_{max}, determine the transmission range r_i of each sensor n_i such that

1. k-vertex supernode connectivity: there exist k-vertex disjoint communication paths from every sensor to the set of supernodes, and
2. the total power consumed over all sensor nodes is minimized, i.e. $\sum_{i=1}^{N} p_i$ = minimum.

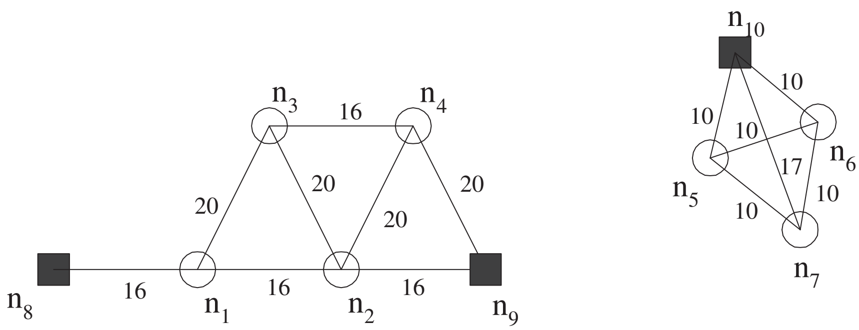
Figure 2 (a) shows an example of a heterogeneous WSN which is 3-vertex supernode connected. That means each sensor node has 3 vertex-disjoint paths to supernodes. For example, sensor n_3 has the following three vertex-disjoint paths to supernodes: $(n_3, n_1, n_8), (n_3, n_4, n_9)$, and (n_3, n_2, n_9).

Sensor nodes are prone to failure due to physical damage or energy depletion, and thus our goal is to provide a topology that is fault-tolerant to sensor node failures. The k-ATC problem applies to heterogeneous WSN applications where each sensor must have k-vertex disjoint data collection paths at all times. An example of such an application is when each sensor must periodically report its measurements and the data reporting must be fault-tolerant to the failure of up to $k-1$ sensor nodes.

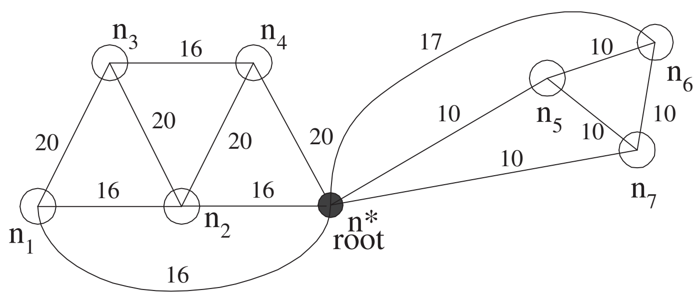
3.3 Network Model

We consider a heterogeneous WSN consisting of M supernodes and N sensor nodes, with $M \ll N$. We are interested in sensor-sensor and sensor-supernode communications only. That is, we do not model the supernode-to-supernode communication.

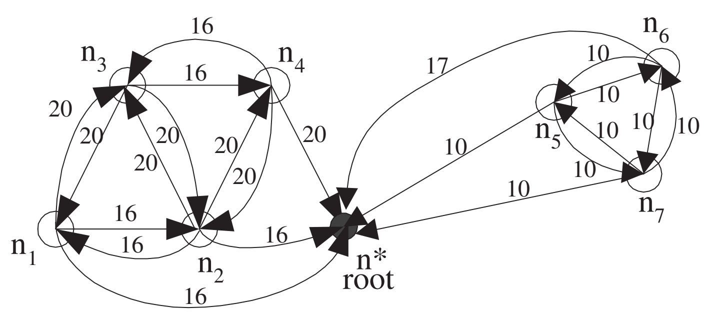
We represent the network topology with an undirected weighted graph $G = (V, E, c)$ in the 2-D plane, where $V = \{n_1, n_2, \ldots, n_N, n_{N+1}, \ldots, n_{N+M}\}$ is the set of nodes and E is the set of edges. The first N nodes in V are the sensor nodes and the last M nodes are the supernodes. When we refer in general to a *node* n_i, it means n_i can be either a supernode or a sensor node. If we specify the index i such that $1 \leq i \leq N$ then we are referring to a sensor node. If $i > N$ then n_i refers to a supernode. We define the set of edges $E = \{(n_i, n_j) | dist(n_i, n_j) \leq R_{max}\}$, where $dist()$ is the Euclidean distance function.



(a) Graph G with N = 7 and M = 3



(b) Reduced graph G^r



(c) Directed reduced graph G^r

Figure 2: Construction of the reduced graph G^r and its directed version $\overline{G}^r$, where "■" represents supernodes, "○" are sensor nodes, and "•" is the root.

The cost function $c(u, v)$ represents the power requirement for both nodes u and v to establish a bidirectional communication link between u and v. Then the cost function is defined as $c(u, v) = (dist(u, v))^\alpha$.

The directed graph $\overline{G} = (V, \overline{E}, c)$ of G is obtained by replacing each edge (u, v) in E with two directed edges (u, v) and (v, u) in $\overline{E}$. The two directed edges maintain the same cost as $c(u, v)$ in G.

We assume that each node has a unique id, such as the MAC address, and that each node is able to gather its own location information using one of the localization techniques for wireless networks, such as [4].

Definition 2 (Reachable Neighborhood)
The reachable neighborhood $\Gamma(n_i)$ is the set of nodes that node n_i can reach by using the maximum transmission range R_{max}, $\Gamma(n_i) = \{n_j \in V | (n_i, n_j) \in E\}$.

For example, in Figure 2 (a), the reachable neighborhood of node n_2 is $\Gamma(n_2) = \{n_1, n_3, n_4, n_9\}$.

Definition 3 (Weight Function)
Given two edges (u_1, v_1) and (u_2, v_2) in E, the weight function $w : E \rightarrow R$ satisfies $w(u_1, v_1) > w(u_2, v_2)$ if and only if:

- $dist(u_1, v_1) > dist(u_2, v_2)$, or
- $dist(u_1, v_1) = dist(u_2, v_2)$ AND $max\{id(u_1), id(v_1)\} > max\{id(u_2), id(v_2)\}$, or
- $dist(u_1, v_1) = dist(u_2, v_2)$ AND $max\{id(u_1), id(v_1)\} = max\{id(u_2), id(v_2)\}$ AND $min\{id(u_1), id(v_1)\} > min\{id(u_2), id(v_2)\}$.

The weight function w guarantees that two edges with different end nodes have different weights. The weight function definition in a directed graph is similar.

Definition 4 (k-vertex Supernode Connectivity)
The heterogeneous network is k-vertex supernode connected if, for any sensor node $n_i \in V$, there are k pairwise vertex disjoint paths from n_i to the set of supernodes (to one or more supernodes). Or equivalently, the heterogeneous network is k-vertex supernode connected if the removal of any $k - 1$ sensor nodes (and all the related links) does not partition the network. That is, for every sensor node n_i there will be a path from n_i to a supernode.

4 Solutions for the k-ATC Problem

In Section 4.1 we introduce the reduced graph, an auxiliary graph used in our solutions. We continue with three solutions for the k-ATC problem. We start with a k-approximation algorithm in Section 4.2, which also serves as a benchmark in our simulations. We continue with a centralized algorithm in Section 4.3 that has the important property of minimizing the maximum power assigned to all the sensors, thus balancing the energy consumption. In Section 4.4, we present an algorithm which is distributed and localized, properties which are important for a large scale WSN.

4.1 Reduced Graph

Given a graph $G(V, E, c)$ corresponding to a heterogeneous WSN and constructed as specified in Section 3.3, we construct its *reduced graph* $G^r(V^r, E^r, c^r)$ as follows. We substitute the set of supernodes with only one node called the *root*. Then $V^r = \{n_1, n_2, ..., n_N, n^*\}$, where the first N nodes are the sensor nodes, and the last node is the root. Edges between sensors remain the same, while an edge between a sensor and a supernode becomes an edge between the sensor and the root. The weight of the edges in G^r remains the same as in G. Figure 2 (b) shows an example of the reduced graph G^r for a heterogeneous WSN with 7 sensor nodes and 3 supernodes.

If a sensor is connected to more than one supernode, then only one edge is added in G^r with the cost corresponding to the distance to the closest supernode. This is because our objective is to pass the sensor data to at least one supernode while minimizing the energy consumption. The pseudocode for constructing the reduced graph is presented below.

We define the directed version $\overline{G}^r(V^r, \overline{E}^r, c^r)$ of the reduced graph as follows. Every undirected edge (n_i, n_j) in G^r between two sensors n_i and n_j is replaced with two directed edges (n_i, n_j) and (n_j, n_i) in $\overline{G}^r$. An edge in G^r between a sensor and the root, is replaced in $\overline{G}^r$ with only one directed edge from the sensor to the root. The reason is that in our problem we are concerned only with collecting sensor data to supernodes and we do not consider the communication out of supernodes. On the other hand, for a link between two sensors, we consider bidirectional communication since each sensor can forward data on behalf of the other sensor. The costs of the edges in $\overline{G}^r$ remain the same as in G^r. Figure 2 (c) shows an example of constructing the directed reduced graph $\overline{G}^r$.

The definitions for reachable neighborhood and weight function remain unchanged for the reduced graphs G^r and $\overline{G}^r$. Next, we define the k-vertex connectivity in the reduced graph G^r.

Definition 5 (k-vertex Connectivity in a Reduced Graph)

Algorithm 1 Construct Reduced Graph (G(V, E, c), N, M)

$V^r := \{n_i | n_i \in V \ and \ i \leq N\} \cup \{n^*\}$;
$E^r := \phi$;
for each edge $(n_i, n_j) \in E$ **do**
  if $(i \leq N)$ AND $(j \leq N)$ **then**
    $E^r := E^r \cup (n_i, n_j)$ and $c^r(n_i, n_j) := c(n_i, n_j)$;
  else if $((i \leq N)$ AND $(j > N))$ OR $((i > N)$ AND $(j \leq N))$ **then**
    $u := min(i, j)$ and $v := max(i, j)$;
    if $(n_u, n^*) \notin E^r$ **then**
      $E^r := E^r \cup (n_u, n^*)$ and $c^r(n_u, n^*) := c(n_u, n_v)$;
    else if $((n_u, n^*) \in E^r)$ AND ($c^r(n_u, n^*) > c(n_u, n_v))$ **then**
      $c^r(n_u, n^*) := c(n_u, n_v)$;
    end if
  end if
end for

The reduced graph G^r is k-vertex connected to the root if for any sensor node $n_i \in V^r$, $i \leq N$, there are k-vertex disjoint paths from n_i to the root n^*. Or equivalently, the reduced graph G^r is k-vertex connected if the removal of any $k-1$ sensor nodes (and all the related links) does not partition the network.

Lemma 1 A heterogeneous WSN is k-vertex supernode connected if and only if the corresponding reduced graph is k-vertex connected to the root.

Proof: Let us consider any sensor node n_i. Assume that the network is k-vertex supernode connected. Then there are k-vertex disjoint paths between n_i and the set of supernodes. By replacing each supernode in the path with the root n^*, we obtain k-vertex disjoint paths between n_i and n^* in the reduced graph G^r.

Similarly, if G^r is k-vertex connected, then for any sensor node n_i there are k-vertex disjoint paths between n_i and n^*. Then for any such path $(n_i, n_{i_1}, \ldots, n_{i_j}, n^*)$ we can take an equivalent path in G by replacing n^* with a supernode n_q, $q > N$, such that $(n_{i_j}, n_q) \in E$ and $c(n_{i_j}, n_q) = c^r(n_{i_j}, n^*)$. These paths in G are k-vertex supernode connected. $\Box$

Definition 5 and Lemma 1 also apply to the directed reduced graph $\overline{G}^r$.

4.2 $MWATC_k$: Minimum Weight-Based Anycast Topology Control

The $MWATC_k$ algorithm proposed in this section uses an algorithm proposed by Frank and Tardos [8] to solve the Min-Weight k-OutConnectivity problem.

The Min-Weight k-OutConnectivity problem is defined as follows. Given a directed graph G and a distinguished vertex r, the objective is to find a directed spanning subgraph of G such that:

1. the sum of the weight of the selected edges is minimized, and

2. there are k-vertex disjoint paths between r and any other vertex in the graph.

The main differences between Min-Weight k-OutConnectivity problem and the problem proposed in this paper are that (1) we are concerned with *InConnectivity*, that is to provide disjoint paths from each vertex to r, and (2) our objective is to minimize the sum of powers assigned to each node, rather than the sum of weights of all edges.

Frank and Tardos [8] propose an optimal solution for the Min-Weight k-OutConnectivity problem solvable in polynomial time, using a solution for the maximum cost submodular flow problem. Let us call this solution FT in our paper.

Wang et al [24] apply the FT algorithm and obtain an approximation algorithm with performance ratio k for the Min-Power k-InConnectivity problem. Here, the objective is to minimize the sum of the power of each node when there are k-disjoint paths from each node to the root. We use the same framework for our k-ATC problem.

Algorithm 2 Algorithm $MWATC_k$
Input: $G(V, E, c)$, a k-vertex supernode connected graph
Output: power assignment p_i for each sensor node n_i

1: Construct the reduced graph $\overline{G}^r(V^r, \overline{E}^r, c^r)$ of G;
2: Construct $\overline{G}'^r$ by reversing the direction of each edge in $\overline{G}^r$ and keeping the weight of each edge the same;
3: $\overline{G}'_{FT}$:= FT($\overline{G}'^r, k, n^*$);
4: Construct $\overline{G}_{FT}$ by reversing each edge in $\overline{G}'_{FT}$ and keeping the weight of each edge the same;
5: **for** $i := 1$ to N **do**
6: $\quad p_i := max\{c^r(n_i, n_j) | (n_i, n_j) \; is \; an \; edge \; in \; \overline{G}_{FT}\}$;
7: **end for**

In this algorithm, we first construct the reduced graph $\overline{G}^r$ and then reverse the edge directions in order to transform from the requirement of k-InConnectivity to the requirement of k-

OutConnectivity. Next, we apply the FT algorithm [8] that optimally solves the Min-Weight k-OutConnectivity problem. The result of the FT algorithm is a directed subgraph of $\overline{G}'^{r}$. We reverse the edge directions one more time to transform back to the k-InConnectivity requirement. The power of each sensor node is assigned such that it will cover all of its 1-hop neighbors in the resulting subgraph.

The complexity of $MWATC_k$ is dominated by the running time of the FT algorithm. Gabow [9] has given an implementation for the FT algorithm that runs in time $O(k^2n^2m)$ where n and m are the number of vertices and number of edges in the graph. Thus, the complexity of the $MWATC_k$ algorithm is $O(k^2N^2\overline{E}^r)$.

Theorem 1. $MWATC_k$ is an approximation algorithm with performance ratio k for the k-ATC problem.

Proof: Let OPT^r be an optimal solution for the Min-Power k-InConnectivity problem in the reduced graph $\overline{G}^r$ and OPT be an optimal solution to the k-ATC problem in the graph G. From the way we construct $\overline{G}^r$ starting from G, we observe that any solution to the k-InConnectivity problem in $\overline{G}^r$ is also a solution to k-ATC problem in G, and vice versa.

Let SOL^r be a solution using the $MWATC_k$ algorithm for the k-InConnectivity problem in $\overline{G}^r$, with $SOL^r = \sum_{i=1}^{N} p_i$. Let SOL be the corresponding solution to k-ATC problem in $\overline{G}$ where the power assigned to each node is the same as in SOL^r.

Since we used the FT algorithm, the solution SOL^r has a performance ratio k to the Min-Power k-InConnectivity problem in a rooted graph (in our case $\overline{G}^r$). The formal proof for the k-approximation ratio is presented in [24]. Then we have the following inequality:

$SOL = SOL^r \leq k \times OPT^r = k \times OPT$

and thus $MWATC_k$ is a k-approximation algorithm. $\square$

4.3 $GATC_k$: Fault-Tolerant Global Anycast Topology Control

In this section we present a centralized greedy algorithm, $GATC_k$, that builds a k-vertex supernode connected subgraph and then assigns to each vertex the minimum power needed to cover all of its 1-hop neighbors.

This algorithm has the property that it minimizes the maximum transmission power for all the sensor nodes, among all other k-vertex supernode connected subgraphs. This property is important since it balances the power consumption among all sensor nodes. The algorithm is presented below.

The algorithm $GATC_k$ starts from the k-vertex supernode connected graph G, constructs its

Algorithm 3 Algorithm $GATC_k$

Input: $G(V, E, c)$, a k-vertex supernode connected graph

Output: power assignment p_i for each sensor node n_i

1: Construct the directed reduced graph $\overline{G}^r(V^r, \overline{E}^r, c^r)$ of G;
2: Let $\overline{G}_k := (V_k, \overline{E}_k, c^r)$ with $V_k := V^r$ and $\overline{E}_k := \overline{E}^r$;
3: Sort all edges in $\overline{E}_k$ in decreasing order of weight (using Definition 3);
4: **for** each edge (u, v) in the sorted order **do**
5:     $\overline{E}'_k := \overline{E}_k \setminus \{(u, v)\}$;
6:     **if** u is k-vertex connected to the root in the graph $(V_k, \overline{E}'_k)$ **then**
7:         $\overline{E}_k := \overline{E}'_k$;
8:     **end if**
9: **end for**
10: **for** i := 1 to N **do**
11:     $p_i := max\{c^r(n_i, n_j) | n_j \in V_k \; and \; (n_i, n_j) \in \overline{E}_k\}$;
12: **end for**

reduced graph G^r, and then transforms it to a directed graph $\overline{G}^r$ as explained in Section 4.1. Based on Lemma 1, G^r and $\overline{G}^r$ are k-vertex connected to the root. We examine all edges in $\overline{G}^r$ in decreasing order and remove an edge (u, v) if after its removal, sensor node u remains k-connected to the root. Then the algorithm computes the power p_i for each sensor node n_i such that n_i can directly communicate with any other node joined by an edge in $\overline{E}_k$.

By using network flow techniques [7], a query on whether two vertices are k-connected in a graph (V, E) can be answered in $O(E + V)$ time for any fixed k. Therefore, the complexity of $GATC_k$ is $O(\overline{E}^r(\overline{E}^r + V^r)) = O((\overline{E}^r)^2)$.

Theorem 2 (Correctness). If G is k-vertex supernode connected then the power assigned by $GATC_k$ to each sensor node guarantees a k-vertex supernode connected topology. Thus $GATC_k$ preserves the k-vertex supernode connectivity of G.

Proof: Since G is k-vertex supernode connected, the graphs G^r and $\overline{G}^r$ are k-connected to the root (see Lemma 1). We start from a graph $\overline{G}_k := \overline{G}^r$ and remove edges. We prove that the resultant graph $\overline{G}_k$ remains k-connected at the end of line 9 in algorithm $GATC_k$.

We show that if $\overline{G}_k$ is k-vertex connected to the root before the removal of an edge (u, v), then it remains k-vertex connected to the root after the edge removal as long as u remains k-vertex connected to the root. To show that $\overline{G}_k$ is k-vertex connected to the root, we show that after the removal of any set C of vertices, $|C| \leq k - 1$, the remaining sensor nodes are still connected to

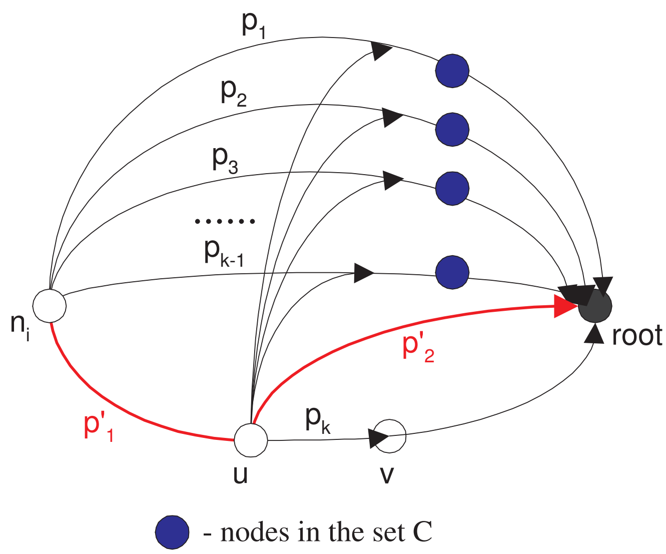


Figure 3: Case when $k-1$ nodes are removed from the paths p_1, p_2, . . ., p_{k-1}.

the root.

Let us take any sensor node n_i. Before the removal of (u, v), n_i has k-vertex disjoint paths to the root, say $p_1, p_2, \ldots, p_k$. If (u, v) is not on any path $p_1, p_2, \ldots, p_k$, then the removal of (u, v) does not affect n_i's connectivity. Let us assume now that (u, v) belongs to one of the paths, let us say $(u, v) \in p_k$. If $|C| < k-1$, then after the removal of C and edge (u, v), n_i is still connected to the root.

Consider now the case $|C| = k-1$ when any $k-1$ vertices are removed from the graph. The only critical case is when one vertex is removed from each path $p_1, p_2, \ldots, p_{k-1}$ and edge (u, v) is removed from the path p_k. This case is illustrated in Figure 3. Node n_i is still connected to u along the path p_k and we will call this path p'_1 which is a subpath of p_k. Vertex u is k-vertex connected to the root after the removal of (u, v) so there are k-vertex disjoint paths between u and the root. Since $|C| = k-1$, only $k-1$ such paths can be broken, so after the removal of C, there will still exist one path between u and the root, let us call it p'_2. Then $p'_1 + p'_2$ will give us a path between n_i and the root.

Therefore, we conclude that $\overline{G}_k$ remains k-vertex connected to the root after the removal of (u, v) as long as u remains k-vertex connected to the root. $\square$

Theorem 3. The maximum transmission range (or equivalently power) among all the sensor nodes is minimized by $GATC_k$.

Table 1: $DATC_k$ notations.

f_i	1 if sensor node n_i decided its final power, otherwise 0
r_i	Current transmission range of sensor node n_i
p_i	Current transmission power level of sensor node n_i, $p_i = r_i^{\alpha}$
$\Gamma(n_i)$	$\{n_j \mid dist(n_i, n_j) \leq R_{max}\}$
p_i^{max}	Transmission power of node n_i needed to reach the farthest neighbor in $\Gamma(n_i)$
p_i^{min}	Transmission power of node n_i needed to reach the closest k neighbors in $\Gamma(n_i)$
$\overline{G}_{n_i}$	n_i's localized topology view; directed graph $\overline{G}_{n_i} = (V_{n_i}, \overline{E}_{n_i})$ where $V_{n_i} = \Gamma(n_i)$ and $\overline{E}_{n_i} = \{(n_u, n_v) \mid n_u, n_v \in V_{n_i}$ AND $dist(n_u, n_v) \leq r_u\}$
$\Gamma'(n_i)$	$\{n_j \mid dist(n_i, n_j) \leq r_i\} \cup \{n_j \mid (r_i < dist(n_i, n_j) \leq R_{max})$ AND (n_i is k-vertex connected to n_j in $\overline{G}_{n_i}$) $\}$

Proof: We show this property by contradiction. Let (u, v) be the first edge that is not removed from $\overline{E}_k$ as we examine the list of decreasingly ordered edges by weight. Then u will have the maximum range between all the senor nodes in $\overline{G}_k$.

Assume by contradiction that there exists a topology $\widetilde{G}$ that has the maximum transmission range from all the sensor nodes less than $c^r(u, v)$. Then the induced topology $\widetilde{G}$ does not contain any edge with cost greater than or equal to $c^r(u, v)$. Since $GATC_k$ could not remove the edge (u, v) from $\overline{E}_k$, it results that, without the edge (u, v), u is not k-connected to the root, thus violating the connectivity correctness of $\widetilde{G}$. $\Box$

4.4 $DATC_k$: Fault-Tolerant Distributed Anycast Topology Control

$DATC_k$ is a distributed and localized algorithm that efficiently assigns the power level of each sensor node such that k-vertex supernode connectivity is preserved. The main algorithm notations are introduced in Table 1.

Each node n_i starts by constructing its localized neighborhood $\Gamma(n_i)$ based on *Hello* messages exchanged between neighbors with communication range R_{max}. Each sensor node n_i starts a distributed process to decide its final transmission power p_i, as presented next in the $DATC_k$(i) algorithm.

Sensor node n_i computes p_i^{max} and p_i^{min}, the power needed to reach the farthest neighbor in $\Gamma(n_i)$ and the first k neighbors in $\Gamma(n_i)$, respectively. Each sensor node n_i uses an iterative process

Algorithm 4 Algorithm $DATC_k$(i)

$p_i := p_i^{min}$;
if $p_i^{min} = p_i^{max}$ **then**
    $f_i := 1$;
else
    $f_i := 0$;
end if
Broadcast(i, p_i, f_i);
while $f_i = 0$ **do**
    compute Δp_i, the minimum incremental power needed to cover at least one neighbor in $\Gamma(n_i) - \Gamma'(n_i)$;
    start timer t;
    if broadcast message received from a neighbor n_j before t expires **then**
        update $\Gamma'(n_i)$ and Δp_i;
        if $\Gamma'(n_i) = \Gamma(n_i)$ **then**
            $f_i := 1$;
            Broadcast(i, p_i, f_i);
            Return;
        end if
    end if
    if timer t expires **then**
        $p_i := p_i + \Delta p_i$;
        update $\Gamma'(n_i)$;
        if $\Gamma'(n_i) = \Gamma(n_i)$ **then**
            $f_i := 1$;
        end if
        Broadcast(i, p_i, f_i);
    end if
end while
Return;

to establish its final power, starting from p_i^{min}. The final power p_i selected by node n_i will be between p_i^{min} and p_i^{max}. In order for a node to be k-vertex connected, it must have at least k disjoint neighbors. Therefore, its transmission power must cover the k closest neighbors resulting in $p_i \geq p_i^{min}$.

The goal of the algorithm is to find a minimum transmission power p_i of node n_i, $p_i \in [p_i^{min}, p_i^{max}]$, such that each node n_j in $\Gamma(n_i)$ is either within communication range r_i of node n_i or there exist k-vertex disjoint paths between n_i and n_j. When this condition is met, node n_i declares its current power estimate as its final power assignment, by setting f_i to 1.

Every node n_i maintains p_j value of each neighbor $n_j \in \Gamma(n_i)$. We assume that a node n_i has a complete topological view of its 1-hop neighborhood, and this is a directed, asymmetric graph $\overline{G}_{n_i}$ where nodes have different communication ranges. The edge set of this topology changes over time (new edges are added) as n_i receives advertisements from its neighbors. A node n_i can compute the connectivity between any two 1-hop neighbors if nodes broadcast their location or their 1-hop neighbors in the *Hello* messages.

The algorithm executes in at most $|\Gamma(n_i)| - k$ rounds (or iterations). In each round, power level p_i is minimally incremented with Δp_i such that at least one node in $\Gamma(n_i) - \Gamma'(n_i)$ is added to $\Gamma'(n_i)$. As specified in Table 1, $\Gamma'(n_i)$ represents the set of neighbors that are either within the range r_i of n_i or those nodes that can be reached from n_i through k-vertex disjoint paths. The value Δp_i can easily be computed since node n_i maintains the distance and location information for all nodes in $\Gamma(n_i)$. The algorithm is completed when $\Gamma(n_i) = \Gamma'(n_i)$.

All broadcast messages that are sent to advertise new power level updates are sent with power level $p_{max} = R_{max}^{\alpha}$. If, during the back-off interval, a broadcast message is received from a neighbor in $\Gamma(n_i)$, then $\Gamma'(n_i)$ and Δp_i are updated before continuing the back-off waiting. When node n_i decides to broadcast its advertisement, it updates its power level p_i and neighboring set $\Gamma'(n_i)$ in lines $20 - 21$ of algorithm $DATC_k$.

The rounds should be designed to have each node advertise its new power estimate once, in the event that the node did not establish its final power yet. Ideally, nodes send the broadcast without colliding with their neighbors' advertisement. To avoid simultaneous updates among neighbors, a back-off scheme is used. Each node backs-off a time inversely proportional to its calculated gain before sending a broadcast. The gain can be computed, for example, as $p_{max} - (p_i + \Delta p_i)$. In this case, nodes with a smaller power level will advertise earlier, thus helping the nodes with larger transmission power. This approach could help to balance power consumption among sensor nodes.

The complexity of the $DATC_k$ algorithm run by each node n_i is polynomial in the total number

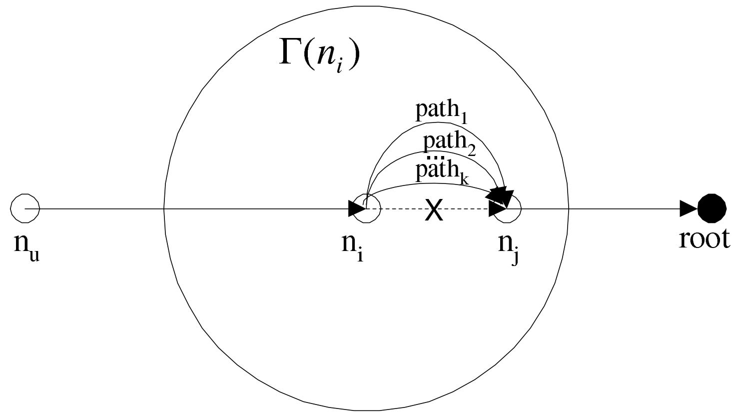


Figure 4: An node n_i does not need to reach n_j directly if there are k disjoint paths between n_i and n_j.

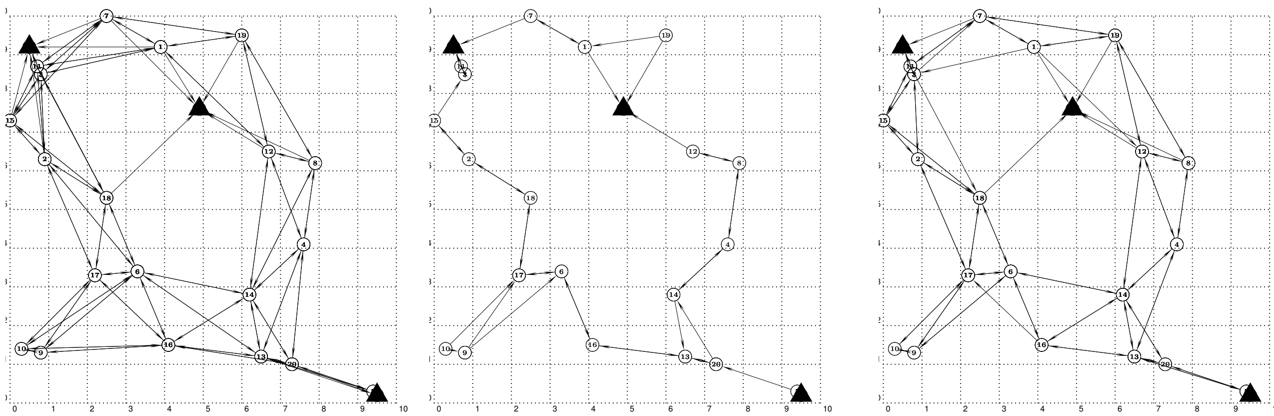
of nodes $N+M$. Let us denote the maximum node degree as Δ, that is $\Delta = max_{i=1..N}|\Gamma(n_i)|$. The complexity of $DATC_k$ is $O(\Delta^5)$. This is because, for a node n_i there are at most $O(\Delta)$ rounds, the time to update Δp_i is at most $O(\Delta^3)$, and during the back-off at most Δ neighbor updates can be received.

The message complexity of a sensor node n_i can be summarized as follows. Assuming an ideal MAC protocol with no collisions and retransmissions, sensor n_i transmits at most $1 + \Delta - k = O(\Delta)$ messages. A *Hello* message is transmitted at the beginning of the protocol for neighbor discovery. Then the algorithm has at most $\Delta - k$ rounds and at most one message is transmitted in each round. Since each sensor has at most Δ neighbors within the communication range and each transmits $O(\Delta)$ messages, the number of messages received by sensor n_i is $O(\Delta^2)$.

Theorem 4 (Correctness). If G is k-vertex supernode connected then the power level assignment provided by the $DATC_k$ algorithm guarantees a k-vertex supernode connected topology.

Proof: For simplicity of our discussion, let us consider G's reduced graph G^r and its directed version $\overline{G}^r$, both being k-connected to the root.

Our proof is by induction. The starting graph $\overline{G}^r$ is the base case, corresponding to a transmission power p_i^{max} for any sensor n_i. We remove edges from this graph when we set the power of a node n_i to a value less than p_i^{max}. For the inductive step, let us assume that the current graph is k-connected to the root and that an edge (n_i, n_j) is removed, or equivalently n_i's final range assignment $r_i < dist(n_i, n_j)$. In conformity with the $DATC_k$ algorithm, this happens when n_i remains k-vertex connected to n_j after the removal of (n_i, n_j). This is illustrated in Figure 4, where sensor n_i does not have to reach n_j directly since there are k other disjoint paths between n_i and



(a) Original topology (b) $GATC_k$ (c) $DATC_k$

Figure 5: Examples of $DATC_k$ and $GATC_k$ ($k = 2, M = 3$).

n_j in $\Gamma(n_i)$.

We show that any sensor node n_u maintains its k-vertex connectivity to the root after the removal of (n_i, n_j). For this, we show that the removal of any set C of vertices, $|C| \leq k-1$ and $n_u \notin C$, does not affect the connectivity of n_u to the root.

Before the removal of (n_i, n_j), n_u has k-vertex disjoint paths to the root, let us say $p_1, p_2, \ldots, p_k$. If (n_i, n_j) is not on any path $p_1, p_2, \ldots, p_k$, then n_u's connectivity is not affected. Assume now that (n_i, n_j) belongs to one of the paths, let us say $(n_i, n_j) \in p_k$. If $|C| < k-1$, then after the removal of C and edge (n_i, n_j), n_u is still connected to the root.

Let us now consider $|C| = k-1$. The only critical case is when one vertex is removed from each path $p_1, p_2, \ldots, p_{k-1}$ and edge (n_i, n_j) is removed from the path p_k. Node n_u is still connected to n_i along the path p_k and we will call this path (which is subpath of p_k) p'_1. Node n_j is still connected to the root along the path p_k and we will call this path (which is subpath of p_k) p'_3. Vertex n_i is k-vertex connected to the node n_j, so after the removal of C, only $k-1$ such paths can be broken. It follows that n_i is still connected to n_j and we will call this path p'_2. Then $p'_1 + p'_2 + p'_3$ will give us a path between n_u and the root.

Therefore, we conclude that the $DATC_k$ algorithm assigns power levels to nodes in such a way that guarantees a k-vertex supernode connected topology. $\square$

Figure 5 (a) shows a sample network with 20 sensor nodes and 3 supernodes ($k = 2, M = 3$). Figure 5 (b) is the resultant topology after applying $GATC_k$ and Figure 5 (c) is the one after $DATC_k$. We can see that $GATC_k$ can reduce the transmission ranges of the sensor nodes more significantly than $DATC_k$.

Table 2: $DATC_k^h$ notations.

$\overline{G}_{n_i}^h$	n_i's localized h-hop topology view; directed graph $\overline{G}_{n_i}^h = (V_{n_i}^h, \overline{E}_{n_i}^h)$ where $V_{n_i}^h = \{n_j \in V \| min\text{-}hops(n_i, n_j) \leq h\}$ and $\overline{E}_{n_i} = \{(n_u, n_v) \| n_u, n_v \in V_{n_i}^h$ AND $dist(n_u, n_v) \leq r_u\}$
$\Gamma'^h(n_i)$	$\{n_j \| dist(n_i, n_j) \leq r_i\} \cup \{n_j \| (r_i < dist(n_i, n_j) \leq R_{max})$ AND (n_i is k-vertex connected to n_j in $\overline{G}_{n_i}^h$) $\}$

4.5 Extension of $DATC_k$ to h-hop Neighborhood

In the $DATC_k$ algorithm discussed above, a sensor node n_i makes decisions based on the information from its 1-hop neighbors, which is the set $\Gamma(n_i)$. In deciding whether to incrementally increase its power such that to directly cover a neighbor, node n_i checks whether in its local view there are k-disjoint paths to that particular neighbor. If such k disjoint paths are identified, node n_i does not need to cover its neighbor directly. Otherwise, n_i will increase its power such as to cover that neighbor directly.

In this section we extend the $DATC_k$ algorithm such that each sensor node maintains topological information about its h-hop neighborhood, and we call this extension $DATC_k^h$. The h-hop neighborhood is maintained by requiring each broadcast message to be forwarded h-hops, using a time-to-live equal to h. By using an h-hop neighborhood, usually for small h, the algorithm is still localized and the main advantage is that a larger neighborhood is used to search for k disjoint paths. Therefore, smaller node power assignments are expected. The trade-off is a higher message complexity, since each update message is forwarded h hops. Simulation results are presented in Section 5.

The algorithm $DATC_k^h(i)$ has the same pseudocode as $DATC_k(i)$ with the observation that the $Broadcast()$ messages are sent over h hops. Also, the last two definitions from Table 1 have to be updated as presented in Table 2.

5 Simulation

In this section we present the results of our simulation. We analyze and compare the performance of $MWATC_k$, $GATC_k$, $DATC_k$, and $DATC_k^h$ with various parameters. We use CPLEX [5] to implement $MWATC_k$ in a small scale network. The other two approaches are tested on a custom simulator using C++ in a large scale network.

5.1 Simulation Environment and Settings

The sensors are deployed in a $100m \times 100m$ area. The supernodes are uniformly deployed in this area. The following parameters and their trade-offs are considered in the simulation:

1. The network size N. We vary N to examine the scalability of the proposed algorithms. In the small-scale network, the network size is varied from 10 to 50. In the large-scale network, it is in the range of 100 to 500.

2. The number of supernodes M. We set M to 1 and 3 for small scale networks and between 2 and 10 in large scale networks.

3. The value of k. We use 2 and 4 as the values of k in the simulation. We also set k to be 1% of N to study the case when k is a percentage of the number of nodes.

4. The power attenuation exponent α. We use 2 and 4 as the values in the simulation.

5. The number of hops h of the local neighborhood in $DATC_k^h$. We use 1 to 3 as the values of h.

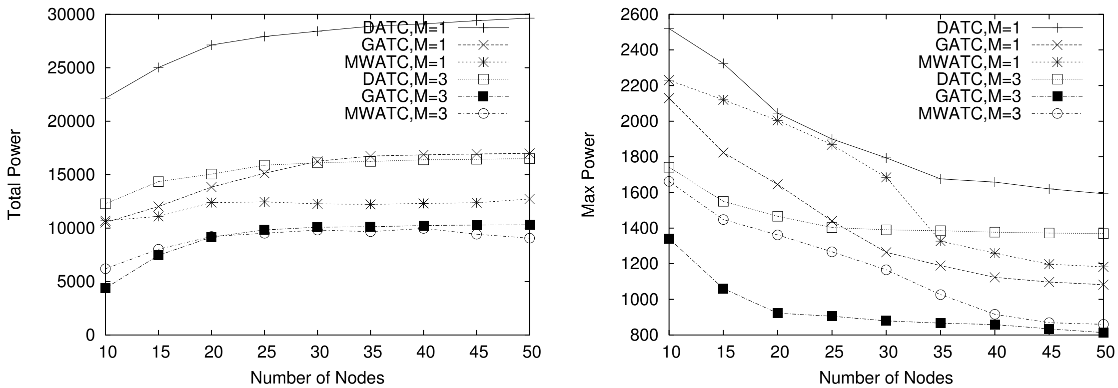
6. The initial sensor transmission range R_{max}. In order to guarantee that the WSN is k-vertex supernode connected, we set the initial sensor transmission range in a small-scale network to be $50m$, and in a large-scale network $20m$ for $k = 2$ and $40m$ for $k = 4$.

A sample network is discarded if it is not k-vertex supernode connected with its initial settings. For each tunable parameter, the simulation is repeated 100 times. The performance metrics are as follows:

1. The total power consumption. This is the summation of power consumption of each sensor (according to its final transmission range).

2. The maximum transmission power among all the sensors. This is to measure the balance of energy consumption among all the sensors. We also compute the standard deviation of energy consumption of the nodes in the network to show the balance degree.

3. The reduction ratio of both total power consumption and maximum power consumption. We use the initial sensor transmission range to calculate the original power consumption.



(a) Total power

(b) Maximum power

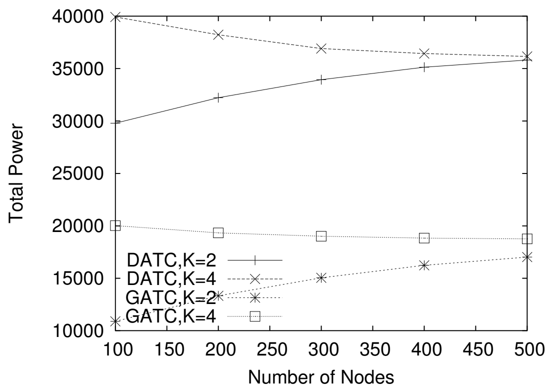
Figure 6: $MWATC_k$, $GATC_k$, and $DATC_k$ in the small scale network.

5.2 Simulation Results

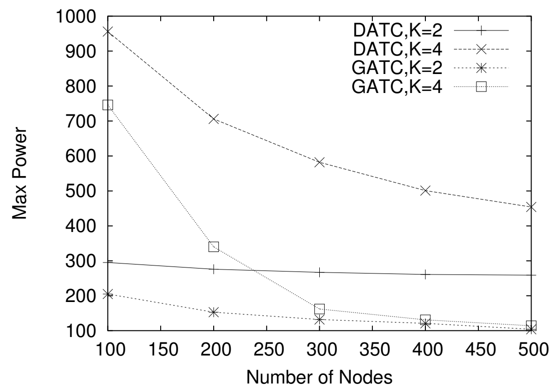
Figure 6 shows the comparison of $MWATC_k$, $GATC_k$, and $DATC_k$ in a small-scale network, where N varies from 10 to 50, M is 1 or 3, k is 2, and α is 2. In Figure 6 (a) we compare the performance of $GATC_k$ and $DATC_k$ with $MWATC_k$ which we proved has a performance ratio of k. We observe that $GATC_k$ performs close to $MWATC_k$, while the distributed algorithm $DATC_k$ has its total power doubled in general. When M is 3, less power is needed than when M is 1. Thus more supernodes scattered in the network help to preserve the k-vertex supernode connectivity.

With the increase in the number of sensors, the total power increases. However, as shown in Figure 6 (a) the rate of increase of power is lower than that of sensors. This is because with more sensors, the total power tends to increase, but the power consumption for each sensor is reduced. Figure 6 (b) is the maximum power comparison. With the increase in the number of sensors, the maximum power decreases for all approaches. $GATC_k$ has the smallest maximum power and $DATC_k$ has the largest one for both $M = 1$ and $M = 3$. When M is larger, the maximum power is smaller for all approaches. These simulations verify our theoretical result that $GATC_k$ minimizes the maximum transmission range between all sensors.

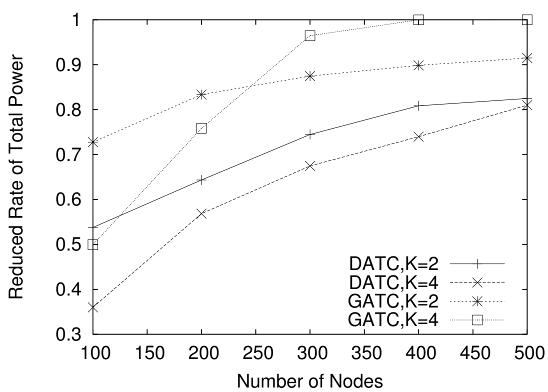
Figure 7 is the comparison of $GATC_k$ and $DATC_k$ in a large scale network, where N varies from 100 to 500, M is 3, α is 2, and k is 2 or 4. Figure 7 (a) is the total power consumption comparison. We can see that $GATC_k$ has better performance than $DATC_k$, and the power consumption is small when k is 2. When k is 2, the power consumption increases with the number of sensors. However, when k is 4, the power consumption decreases slightly. This is because when



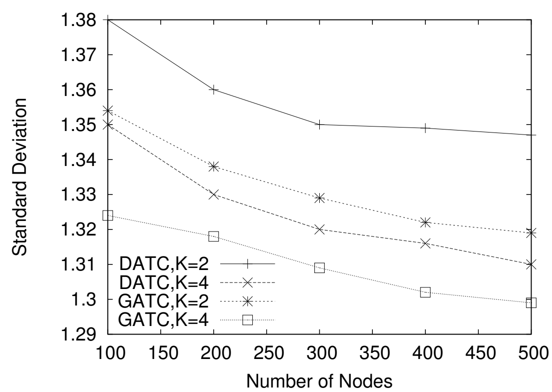
(a) Total power



(b) Maximum power



(c) Reduced rate of total power

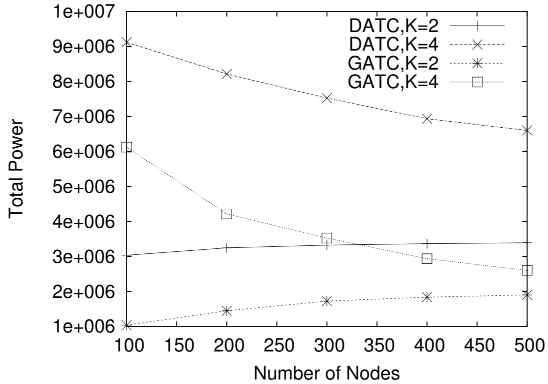


(d) Standard deviation

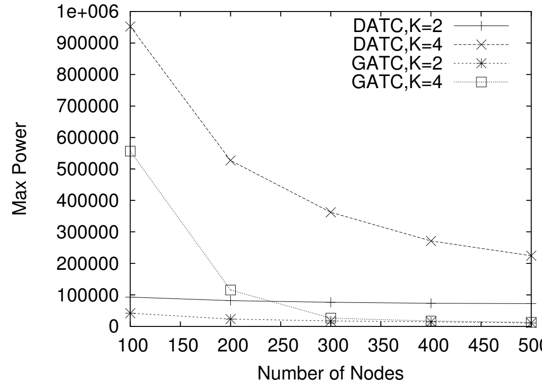
Figure 7: Comparison of $GATC_k$ and $DATC_k$ in the large-scale network.

k is large, the increased number of sensors increase power consumption and helps each sensor to reduce its transmission power. The latter effect is more significant than the former one. Figure 7 (b) is the maximum power comparison. With the growth of the number of sensors, the maximum power decreases for both approaches. $GATC_k$ has smaller maximum power than $DATC_k$. When k is 4, a larger maximum power is needed.

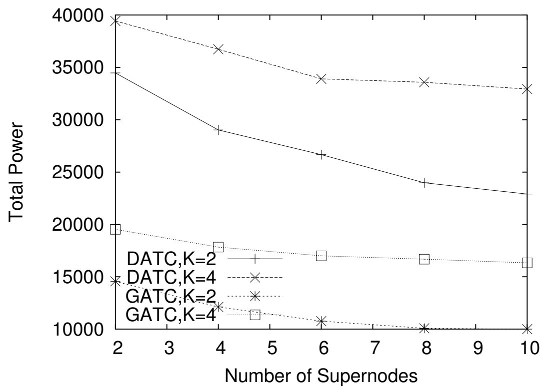
Figures 7 (c) is the corresponding reduced rates of the total power consumption. We compute the reduced rate of the total power consumption as $1 - (p_1 + p_2 + \ldots + p_N)/(p_{max} \times N)$. $GATC_k$ has larger reduction rate than $DATC_k$ in terms of total power. All of the reduction rates increase with the number of sensors. The increase of power consumption in both $GATC_k$ and $DATC_k$ is small with the growth of the number of sensors, while the initial power consumption increases linearly. Figure 7 (d) is the standard deviation of the energy consumption of each node in the network. $GATC_k$ has a more balanced energy consumption than $DATC_k$. A larger k results in a



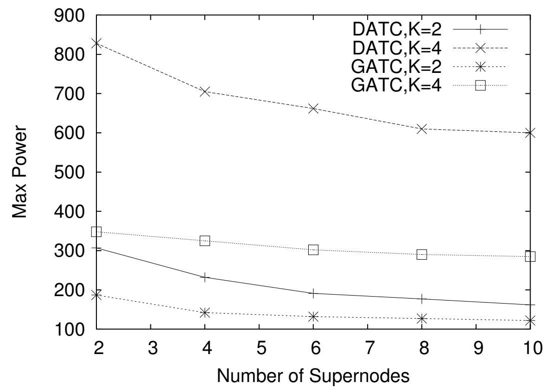
(a) Total power, $\alpha = 4$



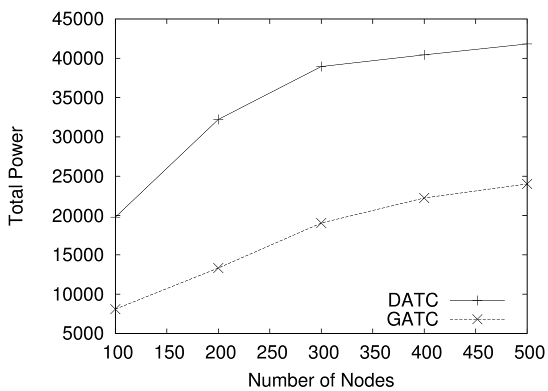
(b) Maximum power, $\alpha = 4$



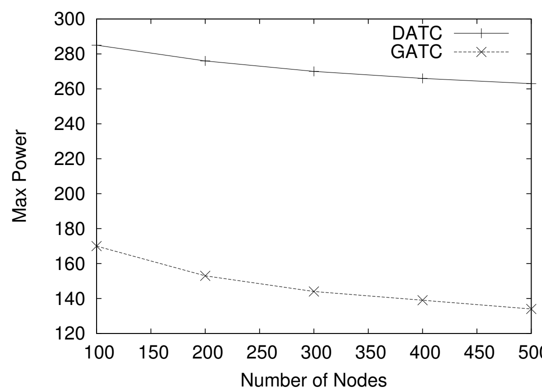
(c) Total power, different M



(d) Maximum power, different M



(e) Total power when k is 1% of N



(f) Maximum power when k is 1% of N

Figure 8: Comparison of $GATC_k$ and $DATC_k$ when $\alpha = 4$, with increasing M, and with increasing k.

more balanced energy consumption scheme. Also when the number of deployed nodes increases, the energy consumption among nodes tends to be more even.

Figure 8 is the analysis of $GATC_k$ and $DATC_k$ with different values for the parameters M, α, and k. Figures 8 (a) and (b) show the resultant power consumption when α is 4 in large scale networks. We set $M = 3$ and $k = 2, 4$. We can see that these two figures are similar to Figures 7 (a) and (b) except that the difference among all the curves is more significant.

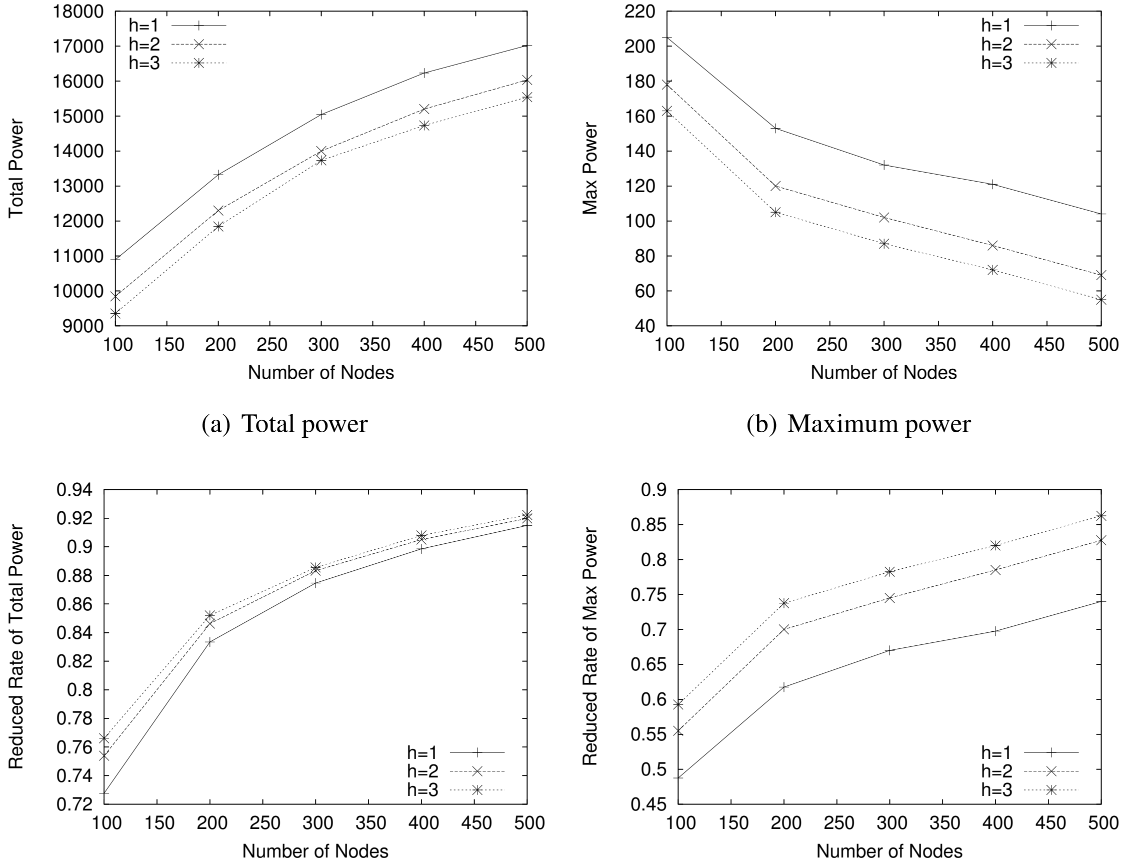
Figures 8 (c) and (d) show the variation of the total power and the maximum power with the number of supernodes when $N = 200$, $\alpha = 2$, and $k = 2, 4$. We can see that with the increase of M, the power consumption is decreased. This is consistent with the results shown in Figures 6 (a) and (b). Again, when k is 4, more power is necessary and $GATC_k$ has better performance than $DATC_k$. We also observe that the decrease of power in $DATC_k$ is more significant than that of $GATC_k$.

Figures 8 (e) and (f) show the variation of total power and the maximum power when $\alpha = 2$, $M = 3$, and k is 1% of the number of nodes in the network. We can see that the total power consumption increases with the number of nodes as well as with the value of k, but not significantly, especially when the number of nodes is relatively large. The maximum power decreases when the number of nodes increases since more nodes provide more chances for the connectivity. Compared with a fixed k, increasing the value of k with the number of nodes leads to larger energy consumption. However, the increase in energy consumption is insignificant.

Figure 9 shows the performance of $DATC_k^h$ with different values of h ($M = 3, k = 2$, and $\alpha = 2$). Figures 9 (a) and (b) are the comparisons in total power consumption and maximum power consumption, respectively. We can see that with the increase of h, both power consumptions decrease. This is because with more hops of neighborhood information, a node has more chances to find k-disjoint paths for its neighbors and thus does not need to increase its power to cover these neighbors. Figures 9 (c) and (d) show the power reduction rate based on Figures 9 (a) and (b). A larger value of h helps to increase the reduced rate of both total power consumption and maximum power consumption. The power reduction of h being 3 is less significant than that of 2. Therefore we know that a relatively small h, 2 or 3, is enough for a good tradeoff between performance and overhead.

The simulation results can be summarized as follows:

1. $MWATC_k$, which is a k performance ratio algorithm, has the best performance in terms of total power consumption. $GATC_k$ has the best performance in terms of maximum power consumption. This verifies our theoretical result that $GATC_k$ minimizes the maximum trans-



(a) Total power

(b) Maximum power

(c) Reduced rate of total power

(d) Reduced rate of maximum power

Figure 9: Performance of $DATC_k^h$ with different h ($M = 3, K = 2, \alpha = 2$).

mission power between all the sensors.

2. More supernodes help to reduce the power consumption of each sensor. Larger k demands larger power consumption in all approaches.

3. When the number of sensors N increases, the total power consumption increases slightly for both $GATC_k$ and $DATC_k$ if k is 2; it decreases slightly if k is 4. The maximum power consumption decreases with the growth of N.

4. The reduction rate in terms of both total power and maximum power increases with the growth of N.

5. When α increases from 2 to 4, the difference between $GATC_k$ and $DATC_k$ is more significant.

6. When h increases in $DATC_k^h$, both the total power consumption and the maximum power consumption can be reduced. A small value of h can provide a good performance.

6 Conclusions

In this paper we addressed the k-degree Anycast Topology Control problem in heterogeneous WSNs with objective of minimizing the total energy consumption while providing k vertex independent paths from each sensor node to one or more supernodes. Such a topology provides the infrastructure for fault-tolerant data gathering applications robust to the failure of up to $k-1$ sensors.

We proposed three solutions to this problem, two centralized approaches $MWATC_k$ and $GATC_k$, and one distributed and localized algorithm, $DATC_k$. $MWATC_k$ is an approximation algorithm with performance ratio k, and $GATC_k$ has the property that it minimizes the maximum power between all sensor nodes. Simulation results show that among the three proposed algorithms, $MWATC_k$ has the best performance in terms of total power consumption and $GATC_k$ has the best performance in terms of maximum power consumption. $DATC_k$ consumes the most power, sometimes as high as twice that of $GATC_k$. However, $DATC_k$ is a distributed and localized algorithm, and this is an important property in WSNs showing that this algorithm is scalable and practical for large networks.

For future work, we plan to extend our work for applications that require a fault-tolerant bidirectional topology that provides communication paths both from sensors-to-supernodes and from supernodes-to-sensors. Another related problem that we will address is deriving the value of k when we know the network topology.

References

[1] M. Bahramgiri, M. T. Hajiaghayi, and V. S. Mirrokni, Fault-Tolerant and 3-Dimensional Distributed Topology Control Algorithms in Wireless Multi-hop Networks, *IEEE Int'l. Conf. on Computer Communications and Networks (ICCCN'02)*, 2002.

[2] G. Calinescu and P. -J. Wan, Range Assignment for High Connectivity in Wireless Ad Hoc Networks, *2nd Int'l. Conf. on Ad-Hoc Networks and Wireless*, 2003.

[3] M. Cardei and Y. Yang, Heterogeneous Wireless Sensor Networks, to appear in the *Encyclopedia of Wireless and Mobile Communications*, B. Furht (ed.), CRC Press, Taylor & Francis Group.

[4] X. Cheng, A. Thaeler, G. Xue, and D. Chen, TPS: A Time-Based Positioning Scheme for Outdoor Wireless Sensor Networks, *IEEE INFOCOM*, 2004.

[5] CPLEX solver, http://www.cplex.com.

[6] Crossbow Mica2 motes and Stargate-Xscale, http://www.xbow.com.

[7] S. Even and R. E. Tarjan, Network Flow and Testing Graph Connectivity, *SIAM Journal on Computing*, Vol. 4, pp. 507-518, 1975.

[8] A. Frank and E. Tardos, An Application of Submodular Flows, *Linear Algebra and its Applications*, Vol. 114-115, pp. 329-348, 1989.

[9] H. N. Gabow, A Representation for Crossing Set Families with Applications to Submodular Flow Problems, *4th ACM-SIAM Symposium on Discrete Algorithms*, pp. 202-211, 1993.

[10] O. Gnawali, B. Greenstein, K.-Y. Jang, A. Joki, J. Paek, M. Vieira, D. Estrin, R. Govindan, and E. Kohler, The Tenet Architecture for Tiered Sensor Networks, *ACM SenSys*, Nov. 1-3, 2006.

[11] M. Hajiaghayi, N. Immorlica, and V. S. Mirrokni, Power Optimization in Fault-Tolerant Topology Control Algorithms for Wireless Multi-Hop Networks, *9th Annual Int'l. Conf. on Mobile Computing and Networking*, 2003.

[12] L. Hu, Topology Control for Multihop Packet Radio Networks, *IEEE Transactions on Communications*, 41(10), 1993.

[13] F. Ingelrest, D. Simplot-Ryl, and I. Stojmenovic, Optimal Transmission Radius for Energy Efficient Broadcasting Protocol in Ad Hoc and Sensor Networks, *IEEE Transactions on Parallel and Distributed Systems*, 17(6), 2006.

[14] Heterogeneous Networks with Intel XScale,
http://www.intel.com/research/exploratory/heterogeneous.htm.

[15] X. Jia, D. Kim, P. Wan, and C. Yi, Power Assignment for k-Connectivity in Wireless Ad Hoc Networks, *IEEE INFOCOM*, 2005.

[16] N. Li and J. C. Hou, FLSS: A Fault-Tolerant Topology Control Algorithm for Wireless Networks, *10th Annual Int'l. Conf. on Mobile Computing and Networking*, 2004.

[17] E. L. Lloyd, R. Liu, M. V. Marathe, R. Ramanathan, and S. S. Ravi, Algorithmic Aspects of Topology Control Problems for Ad Hoc Networks, *MobiHoc'02*, 2002.

[18] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, Wireless Sensor Networks for Habitat Monitoring, *Int'l. Workshop on Wireless Sensor Networks and Applications (WSNA'02)*, 2002.

[19] T. Melodia, D. Pompili, V. C. Gungor, and I. Akyildiz, A Distributed Coordination Framework for Wireless Sensor and Actor Networks, *MobiHoc'05*, 2005.

[20] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, Energy-Aware Wireless Microsensor Networks, *IEEE Signal Processing Magazine*, Vol. 19, pp. 40-50, 2002.

[21] R. Ramanathan and R. Rosales-Hain, Topology Control of Multihop Wireless Networks using Transmit Power Adjustment, *IEEE INFOCOM*, 2000.

[22] T. Schoellhammer, B. Greenstein, and D. Estrin, Hyper: A Routing Protocol to Support Mobile Users of Sensor Networks, *CENS Technical Report #63*, Jan. 2006.

[23] H. Wang, D. Estrin, and L. Girod, Preprocessing in a Tiered Sensor Network for Habitat Monitoring, *EURASIP Journal on Applied Signal Processing*, Vol. 4, pp. 392-401, 2003.

[24] F. Wang, M. T. Thai, Y. Li, and D. -Z. Du, Fault Tolerant Topology Control for All-to-One and One-to-All Communication in Wireless Networks, University of Minnesota *Technical Report TR 05-038*, Nov. 2005.

[25] M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu, and S. Singh, Exploiting Heterogeneity in Sensor Networks, *IEEE INFOCOM*, 2005.