FAULT-TOLERANT TOPOLOGY CONTROL IN HETEROGENEOUS WIRELESS SENSOR NETWORKS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

HAKKI BAĞCI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
COMPUTER ENGINEERING

OCTOBER 2013

Approval of the thesis:

FAULT-TOLERANT TOPOLOGY CONTROL IN HETEROGENEOUS WIRELESS SENSOR NETWORKS

submitted by **HAKKI BAĞCI** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen _____
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Adnan Yazıcı _____
Head of Department, **Computer Engineering**

Prof. Dr. Adnan Yazıcı _____
Supervisor, **Computer Engineering Dept., METU**

Assoc. Prof. Dr. İbrahim Körpeoğlu _____
Co-supervisor, **Computer Engineering Dept., Bilkent University**

Examining Committee Members:

Prof. Dr. Özgür Ulusoy
Computer Engineering Dept., Bilkent University _____

Prof. Dr. Adnan Yazıcı
Computer Engineering Dept., METU _____

Assoc. Prof. Dr. Ahmet Coşar
Computer Engineering Dept., METU _____

Assist. Prof. Dr. Erhan Eren
Informatics Institute, METU _____

Assist. Prof. Dr. Ahmet Oğuz Akyüz
Computer Engineering Dept., METU _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: HAKKI BAĞCI

Signature :

ABSTRACT

FAULT-TOLERANT TOPOLOGY CONTROL IN HETEROGENEOUS WIRELESS SENSOR NETWORKS

Bağcı, Hakkı

Ph.D., Department of Computer Engineering

Supervisor : Prof. Dr. Adnan Yazıcı

Co-Supervisor : Assoc. Prof. Dr. İbrahim Körpeoğlu

October 2013, 83 pages

Wireless sensor networks have come into prominence for monitoring and tracking operations in many application areas including environmental monitoring, battlefield surveillance, healthcare solutions, vehicle traffic monitoring, smart home systems and many other industrial applications. A long network life time and fault-tolerant operation are two essential requirements for wireless sensor network applications. In order to satisfy these requirements, heterogeneous architectures can be employed in wireless sensor network applications, where the network consists of several layers that are composed of different types of nodes instead of a homogeneous flat topology. In order to address energy efficient and fault-tolerant topology control in heterogeneous wireless sensor networks, we propose a distributed, energy efficient and fault-tolerant topology control algorithm called the Disjoint Path Vector (DPV), which finds disjoint paths from each sensor node to set of supernodes. We prove the correctness of our approach by showing that topologies generated by DPV are guaranteed to satisfy k-vertex supernode connectivity. Our simulations show that the DPV algorithm can achieve a 4-fold reduction in total transmission power and a 2-fold reduction in maximum transmission power compared to existing solutions. Under realistic packet loss scenarios our algorithm can reach 60% decrease in total transmission power.

Keywords: Heterogeneous Wireless Sensor Networks, Topology Control, Fault Tolerance, Energy Efficiency, Disjoint Paths

ÖZ

HETEROJEN KABLOSUZ DUYARGA AĞLARINDA HATA TOLERANSLI TOPOLOJİ KONTROL

Bağcı, Hakkı

Doktora, Bilgisayar Mühendisliği Bölümü Bölümü

Tez Yöneticisi : Prof. Dr. Adnan Yazıcı

Ortak Tez Yöneticisi : Assoc. Prof. Dr. İbrahim Körpeoğlu

Ekim 2013 , 83 sayfa

Kablosuz duyarga ağları, gözlemleme ve takip operasyonlarında çevre gözlemleme, savaş alanı gözetleme, sağlık çözümleri, trafik izleme, akıllı ev sistemleri ve diğer bir çok endüstriyel uygulamalar için önem kazanmaktadır. Kablosuz duyarga ağları uygulamalarının en belirgin isterleri mümkün olduğunca uzun ömürlü olmaları ve bu tip ağlarda sıkça görülen hatalara karşı da tolerans gösterebilmeleridir. Bu isterleri sağlamak amacıyla düz mimariler yerine farklı özelliklere sahip düğümlerden oluşan heterojen mimariler kullanılabilmektedir. Bu tez çalışmasında heterojen yapılı kablosuz duyarga ağlarında enerji etkin ve hataya toleranslı topoloji kontrol problemine çözüm olarak ağda yer alan her düğümden süper düğüm kümesine ayrık yollar bulan dağıtık, enerji etkin ve hataya toleranslı bir topoloji kontrol algoritması (Disjoint Path Vector) öneriyoruz. Algoritmamızın hata toleransı için gerekli olan bağlanırlık seviyesini garanti ettiğini ispat ediyoruz. Yaptığımız simülasyonlar DPV algoritmasının var olan çözümlerden düğümlerin toplam çıkış gücü açısından 4 kat, maksimum çıkış gücü açısından 2 kat daha etkin olabileceğini göstermektedir. Olası paket kaybı senaryolarında algoritmamız diğer çözümlere göre 60% daha düşük toplam çıkış gücü elde edebilmektedir.

Anahtar Kelimeler: Heterojen Kablosuz Duyarga Ağları, Topoloji Kontrol, Hata Toleransı,

I dedicate this thesis to

my family,

my wife, Figen,

and my daughter, İpek.

ACKNOWLEDGMENTS

First and foremost I want to thank my supervisor Prof. Dr. Adnan Yazıcı for all the support and encouragement he gave me. His guidance and positive approach throughout this time period made possible to complete this work.

I would like to express my gratitude and profound respect to my co-advisor Assoc. Prof. Dr. İbrahim Körpeoğlu for his suggestions, efforts and patience during this study. He has made invaluable contributions to my academic studies since we started to work together in 2004.

I acknowledge with thanks and appreciation the members of thesis monitoring comittee, Prof. Dr. Müslim Bozyiğit, Assist. Prof. Dr. Erhan Eren and Assist. Prof. Dr. Ahmet Oğuz Akyüz. Their advice helped me to move forward in this study.

I thank to the members of the thesis jury, Prof. Dr. Özgür Ulusoy and Assoc. Prof. Dr. Ahmet Coşar for reviewing and evaluating my thesis.

I also thank to TUBİTAK / BİLGEM / İLTAREN for supporting my academic studies.

Special thanks go to my unit director at İLTAREN, Dr. Alper Yıldırım for the support and encouragement he gave me during my PhD studies.

I thank to my wife, for her understanding and support during my thesis study.

Finally, I would like to express my thanks to my parents and brother for their love, trust and every kind of supports.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

$\alpha - MOC - CDS$	α Minimum Routing Cost Connected Dominating Set
BAMP	Biconnectivity Augmentation with MinMax Power
CA	Collaborative Algorithm
CA-PCL	Collaborative Algorithm with Probable Critical Link
CBTC	Cone Based Topology Control
CDS	Connected Dominating Set
CDSA	Connected Dominating Set Augmentation
CMP	Connected MinMax Power
CPU	Central Processing Unit
DATC	Fault Tolerant Distributed Anycast Topology Control
DGFT	Distributed Geography-based Fault Tolerant Topology Control
DLSS	Directed Local Spanning Subgraph
DPV	Disjoint Path Vector
DRNG	Directed Relative Neighborhood Graph
EECDS	Energy Efficient Connected Dominating Set
FGSS	Fault Tolerant Global Spanning Subgraph
FLSS	Fault Tolerant Local Spanning Subgraph
FTTC	Fault Tolerant Topology Control
GATC	Fault Tolerant Global Anycast Topology Control
GG	Gabriel Graph
k-ATC	k-degree Anycast Topology Control
LILT	Local Information Link State Topology
LINT	Local Information No Topology
LMST	Local Minimum Spanning Tree
MAC	Medium Access Control
MCDS	Minimum Connected Dominating Set
MST	Minimum Spanning Tree
NAP	Neighbor Addition Protocol

PLR	Packet Loss Ratio
RNG	Relative Neighbor Graph
TCN	Topology Control Neighbors
TTL	Time To Live
WSAN	Wireless Sensor and Actor Network
WSN	Wireless Sensor Network

CHAPTER 1

INTRODUCTION

Wireless sensor networks (WSNs) have been studied extensively in many aspects due to their broad range of potential monitoring and tracking applications including environmental observation and habitat monitoring [2], [31], [15], fire detection [53], military applications [60], [26], [101], [80], health care solutions [49], [39], [3], [23], traffic tracking [10], [78], [83] , smart buildings [58] and many other industrial applications [96], [36] , [41], [85]. WSNs are typically composed of tens or hundreds of tiny sensors that are capable of sensing some phenomena, processing and transmitting the data via a wireless channel. Sensor nodes collaborate in a distributed and autonomous manner to accomplish a certain task usually in an environment without any infrastructure.

Sensor nodes in WSNs should be low cost and small in size in order to be used in huge amounts in a real world application. This situation restricts the sensor nodes in many aspects as they have limited power, short transmission range, relatively slow CPUs and a small memory. These limitations bring out many challenges unique to wireless sensor networks.

Power efficiency and fault tolerance are essential properties to have for WSNs in order to keep the network functioning properly in case of energy depletion, hardware failures, communication link errors, or adverse environmental conditions, events that are likely to occur quite frequently in WSNs [21], [25], [11], [24], [19]. Topology control is one of the most important techniques used for reducing energy consumption and maintaining network connectivity [69].

In this thesis, we focus on fault-tolerant topology control in heterogeneous WSNs with a two-layered architecture where the lower layer consists of low-cost ordinary sensor nodes, with limited battery power and short transmission range. The upper layer consists of supernodes, which have more power reserves and better processing and storage capabilities. Links between supernodes have longer ranges and higher data rates; however, supernodes are fewer in number due to their higher cost. Supernodes can also have some special abilities like acting against an event or a certain condition. This type of supernodes are called actors (or actuators), and sensor networks that contain actors are called wireless sensor and actor networks (WSAN). In WSANs, data gathered by sensors is forwarded to actors for performing the required actions [29]. A heterogeneous WSN with supernodes are known to be more reliable and have longer network lifetime than the homogeneous counterparts without supernodes. Heterogeneity can triple the average delivery rate and provide a 5-fold increase in the network lifetime if supernodes are deployed carefully [59].

This thesis introduces a new algorithm called the Disjoint Path Vector (DPV) algorithm for constructing a fault-tolerant topology to route data collected by sensor nodes to supernodes.

In WSNs, guaranteeing k-connectivity of the communication graph is fundamental to obtain a certain degree of fault tolerance [69]. The resulting topology is tolerant up to $k-1$ node failures in the worst case. We propose a distributed algorithm, namely the DPV algorithm, for solving this problem in an efficient way in terms of total transmission power of the resulting topologies, maximum transmission power assigned to sensor nodes, and total number of control message transmissions. Our simulations show that the DPV algorithm achieves a 4-fold reduction in total transmission power and a 2-fold reduction in maximum transmission power compared to existing solutions. Under realistic packet loss scenarios our algorithm results with 60% decrease in total transmission power. The power efficiency of our algorithm directly results from the novel approach that we apply while discovering the disjoint paths. This approach involves in storing full path information instead of just next node information on the paths and provides a large search scope for discovering the best paths throughout the network without the need of global network topology. The details of the proposed algorithm is described in Chapter 4.

The remainder of the thesis is organized as follows. In Chapter 2 we first give a brief overview of key issues in wireless sensor networks (Section 2.1), then we continue by presenting a background for our search which includes the following topics: Heterogeneous Wireless Sensor Networks (Section 2.2), Topology Control in Wireless Sensor Networks (Section 2.3) and Fault Tolerance in Wireless Sensor Networks (Section 2.4). In Chapter 3 we present related approaches related to our work in the literature. In Chapter 4 we describe our proposed algorithm in detail. Chapter 5 presents the evaluation results. We conclude the thesis with Section 6.

CHAPTER 2

BACKGROUND

In this chapter we present a general background on wireless sensor networks, heterogeneity, topology control and fault tolerance in wireless sensor networks which constitute basics for our research.

2.1 Overview of Wireless Sensor Networks

In recent years, development of tiny and cheap sensors with processing capability and a wireless communication subsystem enabled the wireless sensor networks (WSN) technology. The aim of a WSN is to accomplish a monitoring or a tracking task by combining the local data sensed by the individual sensors. WSNs do not require an infrastructure and they can be deployed in a random way on a geographical area and run autonomously without human intervention. Number of sensor nodes in a WSN can be in the order of tens, hundreds and even thousands.

In homogeneous WSNs all the sensors in the network are identical whereas in a heterogeneous WSN some nodes are different than the others in some aspects like processing capabilities, transmitting range, storage and power capacity. Sensor nodes with higher capabilities are often called supernodes. A gateway node can be considered as a supernode which directly transmits the data collected from the sensor nodes to the base station. Heterogeneous WSNs can also include actor nodes (a special type of a supernode) which can show reaction when certain events occur in the monitored region. This type of sensor networks are named as wireless sensor and actor networks(WSAN).

Sensor networks work in a data-centric manner and are usually lead by the queries started from a base station. Sensor nodes which have data that satisfy the given conditions reply the query by sending their sensed data. Usually data collected by the sensor nodes is sent to a base station or a determined sink node which can be an actor or a supernode. The sensor nodes in a sensor network typically use multihop communication instead of direct transmission in order to save energy during the data transmission.

Key features and design challenges for wireless sensor networks can be summarized as follows [13]:

- Sensor nodes are usually battery powered so WSN applications should be energy efficient in order to maximize the network lifetime.

- Sensor nodes are prone to failure and energy drain so WSN applications should be fault tolerant, that is, they should continue working in case of some node failures.
- Number of sensor nodes in a WSN application can be high, so the algorithms running on WSNs should be scalable.
- Due to high number of sensors, getting the global topology information of the network has a high energy cost. Therefore algorithms running on sensor nodes should be distributed and should be able to operate with local data.
- WSNs are usually deployed on infrastructureless geographical areas where manual intervention is not possible. For that reason sensor nodes should be self organizing and autonomously operating in order to achieve their mission.

In addition to the key features and challenges mentioned above there are also many special requirements for different types of WSN applications including quality of service (QoS), real time communication, time synchronization among the sensor nodes, localization, coverage and connectivity requirements, secure communication, in network query evaluation, data aggregation, special protocols for routing and handling mobility.

Mainly five different types of WSNs exist [36]:

1. *Terrestrial WSN*: Consists of high number low-cost sensor nodes. Reliable communication among the densely deployed sensor nodes are important. Terrestrial WSNs can both be deployed in a random way or according to a deployment plan. Random deployment can be done by throwing the sensors from an airplane on a geographical area.
2. *Underground WSN*: In order to monitor the underground conditions sensor nodes are buried into the soil or put in cave, mine or so on. Nodes above the ground are located to gather the data from the sensor nodes and relay it to the base station. Sensor nodes are more expensive in underground WSNs according to terrestrial WSNs because they should be durable to harsh underground conditions and require specialized equipments in order to communicate under the high attenuation and signal loss. In addition, a very carefully planned deployment of sensor nodes is required since the sensors are expensive and redeployment is not an option.
3. *Underwater WSN*: Number of sensors in an underwater WSN is much fewer than a terrestrial WSN both because they are more expensive and they use acoustic waves for communication which is not suitable for dense deployment. Data from the sensor nodes are collected by the autonomous underwater vehicles. Low bandwidth and high latency are challenging problems for this type of WSNs.
4. *Multi-media WSN*: Consists of sensor nodes equipped with camera and microphone. They are deployed according to a plan in order to ensure the coverage of the monitored area. Multi-media WSNs require high bandwidth, quality of service and high energy consumption.
5. *Mobile WSN*: Consists of sensor nodes with capability of moving in order to reposition themselves in the network topology. Mobile nodes can move to area of events and communicate with other nodes when they are in communication range. They require

dynamic routing protocols in contrast to static sensor networks. Localization, navigation and controlling the coverage and connectivity are the main challenges for a mobile WSN.

Here are some application areas of wireless sensor networks and examples among many others

- Military Applications: Border protection, battlefield surveillance, soldier detection and tracking, anti-submarine warfare, chemical/biological vapor detection and soldier wearable sensors [47], [60], [26], [101], [80].
- Environmental Monitoring: Forest fire detection, habitat monitoring, animal tracking, pollution detection, volcano eruption monitoring [53], [2], [31], [15].
- Health care Applications: Vital sign monitoring in hospitals, monitoring aged people at home, large-scale in-field medical studies [49], [39], [3], [23].
- Industrial Applications: Building and factory automation, industrial equipment monitoring, process monitoring and control, underground mine monitoring, offshore oil station monitoring [96], [36] , [41], [85], [56].
- Traffic Monitoring: Traffic load prediction, vehicle routing and tracking, dynamic traffic signaling, intelligent transportation systems, logistics [52], [92], [10], [78], [83] .
- Disaster Relief and Rescue: Detection of survivors after an earthquake [79], [66], [17].
- Smart Buildings: Smart home automation, energy efficient buildings [8], [30].

2.2 Heterogeneous Wireless Sensor Networks

Heterogeneous wireless sensor networks consist of nodes with different capabilities. Nodes may differ in sensing capabilities, transmitting range, processing power, energy capacity, amount of storage and so on. Nodes with superior capabilities according to normal nodes are called supernodes.

Supernodes can have different roles according to the applications. Some supernodes just overlay the data collected from the sensor nodes to a base station. In this scenario a supernode is a gateway node. Supernodes can also make additional processing on the collected data before sending it to a base station. A supernode can apply some filters or can make data aggregation to decrease the amount of data to be transmitted which also decreases the energy consumption.

Supernodes can also have some special abilities like acting against an event or a certain condition. This type of supernodes called as actors (or actuators) and sensor networks that contain actors are called wireless sensor and actor networks (WSAN). WSANs usually have a two-layer architecture where the lower level is composed of low cost sensor nodes and the upper layer consists of resource-rich actor nodes which take decisions and perform appropriate actions [29]. In WSANs, there are usually two types of wireless communication links: actor-actor and sensor-actor links [35]. The links between sensors and actors are assumed to be less reliable [40], hence there are several methods proposed for maintaining reliable

sensor-actor connectivity [35], [40], [81]. The methods of [40] and [81], however, do not employ k-connectivity between sensors and actors and thus they do not guarantee fault-tolerance in case of $k-1$ node failures. Although [35] addresses the k-actor connectivity problem, it does not consider the energy efficiency of the resulting topologies. Our approach differs from these works by maintaining k-connectivity and addressing power efficiency at the same time.

Another special type of heterogeneity arises when the sensor nodes in a WSN have the ability of tuning the transmission power. In such a network each sensor node adjusts its transmission power according to a topology control algorithm which generates a topology with unequal transmission ranges. This approach is useful for energy conservation but arising unidirectional links should be handled carefully.

A heterogeneous WSN with supernodes are known to be more reliable and have longer network lifetime than the homogeneous counterparts without supernodes. Yarvis et al. [59] reported that heterogeneity can triple the average delivery rate and provide a 5-fold increase in the network lifetime if supernodes are deployed carefully. This is apparent since ordinary sensor nodes will not be required to relay the sensed data all the way to the base station, instead data will be sent to a near supernode which has larger energy resources than the ordinary sensor nodes in the network.

In this work we mainly focus on heterogeneous wireless sensor networks with two layered architecture as defined in [51]. There are two types of nodes in our target WSN. In the lower layer we have low cost ordinary sensor nodes with restrictions like limited battery power, short transmission range, small storage, low data rate and low duty cycle. The upper layer consists of supernodes which have more power reserves, better processing and storage capabilities. Links between supernodes have longer ranges and higher data rates. Number of supernodes is fewer than the ordinary sensors due to their higher cost. We do not have special requirements on the supernodes, they can be gateways, in network query evaluators, data aggregators or actors. All supernodes in the network are assumed to be identical so delivering the sensed data to any of the supernodes is sufficient. Supernode - supernode and supernode - base station communication is out of the scope of this work.

2.3 Topology Control in Wireless Sensor Networks

In wireless sensor networks, most of the time, deployment is not done manually and so network topology cannot be known beforehand. Network topology is constructed autonomously by the sensors after the deployment to the application area. Hence a topology control mechanism is needed to build and maintain the network topology. In addition, topology in WSNs is subject to change for several reasons including communication link breaks, nodes running out of power, and mobility. In order to keep the network connected as long as possible and optimize the network lifetime and throughput, topology control mechanisms are essential for WSNs.

Topology control is defined as controlling the neighbor set of nodes in a WSN by adjusting transmission range and/or selecting specific nodes to forward the messages [98]. Topology control approaches can be divided into two main categories, namely, homogeneous and nonhomogeneous [69]. In homogeneous approaches transmission range of all sensors are the same whereas in nonhomogeneous approaches nodes can have different transmission ranges.

We can also distinguish the proposed topology control approaches based on the type of the network topology they address. Most approaches address flat topologies whereas only a few address heterogeneous topologies where the network consists of different types of nodes. Our proposed approach is in the latter category and is different from most of the approaches in this way.

There are many topology control mechanisms in the literature and they can be classified according to the techniques they use. Many topology control methods [54], [46], [62], [51], [61], [64], [77], [50], [16] are built on the transmit power adjustment technique which depends on the sensors that can control their transmit power. Some algorithms [6], [95], [76], [9], [97] use sleep scheduling which aims to decrease energy consumption while nodes are in idle state. Others use geometrical structures, location and direction information, clustering approaches and also combinations of the mentioned techniques [100], [67], [65], [93]. Santi [69] says that approaches that do not modify the transmit power of the nodes cannot be classified as topology control mechanisms but we adopt a more inclusive definition of topology control given by Wang [98] that is controlling set of neighbors by adjusting transmission range and/or selecting specific nodes to forward the messages. Selection of the (logical) neighbors depends on the topology control approach which is designed to meet the requirements for a certain scenario.

Clustering can also be considered as another way of topology control, where the aim is to organize the network into a connected hierarchy for the purpose of balancing load among the nodes and prolonging the network lifetime [68]. Hierarchical clustering techniques [74], [43] select cluster heads depending on various criteria and creates a layered architecture. However, these techniques start with a flat topology and end up with a layered one. On the other hand, we start with a layered architecture from the beginning, where the supernodes are already given. Instead of building clusters, we focus on maintaining fault-tolerant connectivity between sensor nodes and supernodes.

We give a taxonomy of existing topology control approaches below.

Santi [69] groups the topology control mechanisms as follows:

- Homogeneous: Nodes use same transmitting range. The aim is to find a minimum range value that satisfies a certain property.
- Nonhomogeneous: Nodes can have different transmitting range less than a maximum value.
  - Location based: Exact positions of the sensor nodes are known.
    - Range assignment and variants: Location information is used by a centralized authority to assign transmission ranges for each node.
    - Energy-efficient communication: Location information is used in a localized manner to establish a energy efficient topology.
  - Direction based: Exact node locations are not known but nodes can estimate relative directions of their neighbors.
  - Neighbor based: Nodes can distinguish their neighbors according to some measure like distance.

Li et al. [55] gives a taxonomy of topology control problems as follows:

- Sensor Coverage Topology: Focuses on the maximization of sensor coverage area while using less power. Approaches exist targeting the following types of sensor networks:
  - Static Networks
  - Mobile Networks
  - Hybrid Networks
- Sensor Connectivity Topology: Aims to keep the network functioning in an energy efficient manner while preserving the network connectivity.
  - Power control mechanisms: Uses power adjustment techniques to obtain a connected and energy efficient network topology.
  - Power Management Mechanisms: Uses wake/sleep schedules to conserve energy by sending the redundant nodes to sleep.

Energy efficiency is critical for wireless sensor networks because they are composed of energy-constrained sensor nodes. Topology control mechanisms can help WSNs save energy using power adjustment techniques and wake/sleep schedules. According to Zhang et al. beside power control and sleep scheduling, an energy-efficient topology control algorithm have to be aware of traffic load and run in conjunction with routing protocol [94].

Having presented general taxonomies for topology control problems and mechanisms, we now give a brief background about geometric structures frequently used in topology control algorithms.

2.3.1 Geometrical Structures

2.3.1.1 RNG and GG

The relative neighbor graph (RNG) [38] consists of all edges $uv \in E$ such that there is no vertex $w \in V$ with edges uw and $wv \in E$ satisfying $||uw|| < ||uv||$ and $||wv|| < ||uv||$. The definition of RNG is shown in Figure 2.1(a). The brute-force algorithm for creating RNG of a graph with N nodes takes $O(N^3)$ time [12]. More efficient methods also exist for computing RNG. For instance, Supowit's cone based search method finds RNG of a graph in $O(NlogN)$ [42].

Gabriel graph (GG) [44] contains an edge uv if and only if there is no other vertex $w \in V$ in the disk with diameter uv such that uw and $wv \in E$. The definition of GG is shown in Figure 2.1(b).

If the initial unidirectional graph G is connected, the constructed RNG and GG graphs are also connected which is important for wireless sensor networks. These structures can also be built in a localized manner by using only one hope neighbor information which is also a desirable feature for WSNs.

2.3.1.2 Minimum Spanning Tree

Minimum spanning trees (MST) are frequently used in topology control algorithms since they consist of the edges that keep the graph in one connected component, where the total cost of the edges is minimum. The most known algorithms for computing the MST of a graph are Kruskal's [37] and Prim's [73] algorithms. These two algorithms use greedy approach and require global network topology information.

Kruskal's algorithm grows a forest by adding a safe edge with the least weight at each step until all nodes in the graph get connected. The computational cost of the algorithm is $O(ElogE)$ where E is the number of edges in the graph [82].

Unlike Kruskal's algorithm, Prim's algorithm grows a single tree instead of a forest. It starts from an arbitrary node and at each step it adds the least cost edge that connects the current tree to an uncovered node in the graph. The computational cost of Prim's algorithm is $O(E + NlogN)$ where N is the number of nodes in the graph [82].

In wireless sensor networks, most of the time, global network topology is not known to every sensor node and it is also expensive to acquire that information. For that reason, distributed algorithms using local information are more practical for real WSN applications. For instance Local Minimum Spanning Tree (LMST) [65] protocol and LMST_k [93] algorithm are introduced to build a MST-like network topology using only local information in a distributed manner. Li and Hou [61] introduce fault tolerant topology control algorithms based on MST, namely Fault Tolerant Global Spanning Subgraph (FGSS) and Fault Tolerant Local Spanning Subgraph (FLSS) where the former is centralized and the latter is localized.

2.3.1.3 Delaunay Triangulation

Triangulation of set of vertices in V such that no triangle's circumcircle contain any other vertices, is a Delaunay Triangulation. (Assuming there are no four co-circular vertices in V). This structure is not very useful for wireless sensor networks since its construction may cause a high communication overhead because the a triangle can be much larger than the transmission ranges of the sensor nodes which leads to need for global network information. The definition of RNG is shown in Figure 2.1(c).

2.3.1.4 Bounded Degree Spanners

Since MAC-level contention and interference are reduced when node degrees are small, node degree bounded topologies are desirable for wireless sensor networks.

Suppose a node u divides the plane into k equal sized cones with rays originating from u. Yao graph [4] is constructed by finding the shortest edges from u to v in each cone and adding the directional link uv for corresponding edge. Yao graph formation is shown in Figure 2.2. If the link vu is added instead of the link uv, the resulting graph is the reverse of the Yao graph. If in each cone, the edge that has the shortest projection on the angular bisector of the cone is added, the resulting graph is called $\theta - graph$.

In a Yao graph, a node u has a bounded out-degree but it can have unbounded in-degree values

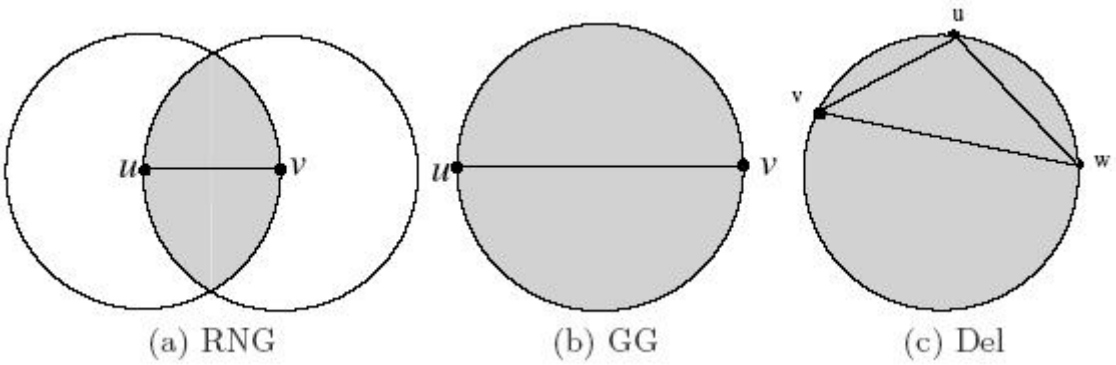


Figure 2.1: RNG, GG and Delaunay Triangulation.

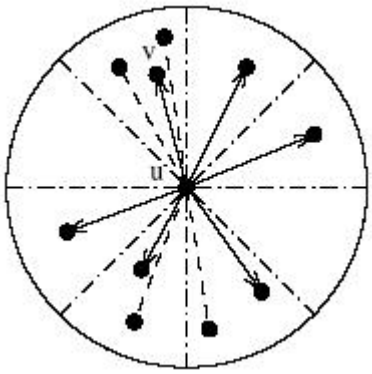


Figure 2.2: Yao Graph Formation.

which may result large communication overhead for node u.

There are also proposed structures for topology control which combine geometric structures and bounded degree spanners. For instance [88] Song et al. proposed a new structure called SYaoGG by applying the ordered Yao structures onto a Gabriel graph. According to this structure, it is guaranteed that there is at most one neighbor node in each of the k equal sized cones of the Yao structure. In a more recent work [87] Song et al. proposed an improved method to further reduce the contention by selecting fewer neighbors for communication. They called the final topology generated by this method as SθGG for power efficient unicast topology. The basic idea of this approach is to remove some links from the underlying Gabriel graph without breaking the power spanner property. In the same work, they also proposed unified power efficient topology for unicast and broadcast called LSθGG. The importance of this work is that it produces a power spanner, bounded node degree, planar topology with low average interference with a localized topology control algorithm.

2.3.1.5 Virtual Backbones

Beside the mentioned flat geometrical structures, there are also hierarchical structures that can be used to control wireless sensor network topologies. In hierarchical topologies, a subset of sensor nodes that behave as gateway or backbone nodes relay packets only. Other sensor nodes send their collected data to these backbone nodes.

Connected dominating sets are often used to construct virtual backbone structures. A dominating set for a graph G = (V, E) is a subset D of V such that every vertex not in D is adjacent to at least one vertex in D by some edge. If D induces a connected subgraph, then it is a connected dominating set. A connected dominating set is said to be minimum (MCDS) if it includes the minimum number of vertices. An MCDS is a useful structure because it provides communication by only using a small subset of nodes in the network. However, problem of finding a MCDS for an arbitrary network is NP-complete [84], so we need to use heuristics and approximation algorithms. The basic idea behind these algorithms is to construct a dominating set first and then add extra nodes in order to obtain a connected dominating set. Another approach is building a redundant CDS and then removing nodes to get a smaller sized CDS. Some of these algorithms can be found in the following works [75], [86], [14], [57], [33] and [34].

Wightman et al. [70] propose a CDS based topology control algorithm called the A3 which selects the CDS tree nodes based on the signal strength and remaining energy. This algorithm results a sub-optimal CDS with relatively longer edge lengths which causes long distance communications. Yuanyuan et. al [102] propose a topology control algorithm called the Energy Efficient CDS (EECDS) which constructs a sub-optimal CDS by first selecting cluster heads and than gateways. Cluster-heads control uniform distribution of energy resources among the network.

Ding et al. [45] introduces a special kind of CDS which is called α Minimum Routing Cost Connected Dominating Set $\alpha - MOC - CDS$ in order to achieve efficient broadcasting and routing in wireless networks. The difference between an $\alpha - MOC - CDS$ and a conventional CDS is that, in $\alpha - MOC - CDS$ for any pair of nodes, there is at least one shortest path between all pairs where the intermediate nodes on the shortest paths all included in $\alpha - MOC - CDS$. Therefore it is still possible to send packages over shortest paths as in the original network. In this work, authors also define another dominating set called $\alpha - 2hop - DS$ and they prove that they are equivalent with each other. They propose a heuristic localized algorithm to construct an $\alpha - 2hop - DS$.

As the authors of [20] and [99] reported, CDS-based topology control algorithms have relatively lower fault tolerance to random link and node failures because in a CDS, each link serves as a bridge and in case of the failure of these links, some nodes may become disconnected. In order to increase the reliability of the CDS structure, they proposed two algorithms, namely, Connected Dominating Set Augmentation (CDSA) and k-connected m-dominating set (k,mCDS) to build a k-connected CDS structure.

2.4 Fault Tolerance in Wireless Sensor Networks

Fault tolerance can be defined as the ability of a system to keep functioning at a desired level in case of faults which occur frequently in many wireless sensor networks due to error prone nature of sensor nodes. There are various factors which cause faults in sensor nodes like energy depletion, hardware failure, communication link errors, malicious attack and so on [25]. In addition, links are also failure prone which may fail permanently or temporarily when blocked by external objects or environmental conditions. Congestion may also lead to faults by causing packet loss due to simultaneous transmission of larger number of sensor nodes [48]. Multihop communication multiplies the faults that can arise in WSN applications. Therefore fault tolerance is an essential requirement for most wireless sensor applications and there are many works addressing this problem in the literature.

Fault tolerance can be addressed at different layers including hardware layer, software layer, network communication layer and application layer. Our primary focus will be on application layer solutions like finding multiple node-disjoint or braided paths, having a certain degree of connectivity of sensor nodes to the sinks, monitoring energy levels of the sensor nodes and maintaining the most reliable paths. Paradis et al [48] lists the reasons why faults in sensor networks cannot be handled in the same way as in traditional wired or wireless networks as follows:

1. Energy consumption is not a constraint for constantly powered wired or rechargeable ad hoc devices.
2. Point to point reliability is important for traditional network protocols, but reliable paths from data source to sinks are more important in WSNs.
3. In WSNs node failures occur much more frequently than the traditional wired or ad hoc networks.
4. MAC protocols in WSNs are not sophisticated as in traditional wireless networks.

For the reasons listed above, requirement for new fault tolerant methods has become clear and several techniques are developed for wireless sensor network applications.

Fault tolerant techniques can be categorized into five main groups [5]:

- Fault prevention: Includes ensuring network connectivity and coverage at the design stage, monitoring network status and taking reactive actions when necessary and maintaining redundant links and nodes.
- Fault detection: Largely depends on the type of the application and type of faults. Main indication of faults are packet loss (decrease in packet delivery rate), interruption, delay in network traffic.
- Fault isolation and identification: Diagnoses and determines the real causes for detected alarms in the network in order to take the right actions.
- Fault recovery: Faults can be recovered within the sensor network or at the sink after the data collection and analysis. Fault recovery within the network is more appropriate for WSN applications since it is costly to forward the data to the sink.

Replication and redundancy of components which are prone to failure is most commonly used method for fault prevention and recovery [25]. For instance if some nodes have problems and fail to sense the environment, redundant nodes in the vicinity can still provide data. Keeping redundant links or multiple paths also provides a fault tolerance when some of the communication links are broken due to node failures or communication problems.

In order to improve reliability and prolong the network lifetime of sensor networks, a two tiered network architecture can be used instead of a flat topology. This method introduces an upper level consisting of powerful relay nodes on top of ordinary sensor nodes. These powerful nodes gather information from sensors and send them directly to base station or via other relay nodes. Hence, sensor nodes do not consume energy for data relaying as much as in the case of flat topology because they send the data to a relay node which is most probably closer than the base station. In order to have a fault tolerant topology each sensor is assigned to more than one relay node, so a sensor node can switch to other relay nodes when communication with the current relay node is lost. Powerful nodes can also process the data gathered by the sensors before sending them to base station which decreases the size of the data transmitted and the energy consumption for transmission.

Relay node placement is a common technique to achieve a certain degree of fault tolerance in two tiered WSNs. There are many works addressing relay node placement in the literature [91], [18], [1], [32], [89] and [90]. The objective is to locate the minimum number of relay nodes to a region with a set of sensors randomly deployed, in order to ensure that each sensor node is covered by at least one relay node and the resulting network topology is k-connected.

Although relay node placement seems to be useful to achieve fault tolerance, it is not very practical in some aspects. First of all in most scenarios since the sensor nodes are deployed to remote regions it is not feasible to make a manual deployment of relay nodes. Secondly, proposed methods to determine the locations of the relay nodes to be deployed require global network information which is not suitable for real applications. Besides, WSNs have a dynamic network topology due to reasons like energy depletion and link errors so node placement needs be repeated to keep network connected and fault tolerant which is not practical in real WSN applications. Therefore a topology control approach should be used to construct and maintain the fault tolerance property of the network.

As stated by Liu et al [25], most of the existing works about fault tolerant topology control aims to obtain k-vertex connectivity between any two sensor nodes in the network. However for WSN applications, it is more important to have fault tolerant paths from sensors to sinks. There are works addressing this problem which make use of power adjustment and geometrical structures which will be discussed in Chapter 3 in detail.

Data gathering and aggregation of the sensed data is on the core of WSN applications. General approach used is to construct a spanning tree connecting all the nodes in the network. However tree structures are not fault tolerant since a subtree becomes disconnected if any node or communication failure occurs. Therefore we require extended geometric structures which have redundant links or multiple paths toward sinks in order to achieve fault tolerance.

CHAPTER 3

RELATED WORK

In this chapter we will give a brief overview of the prominent recent work addressing topology control in wireless sensor networks which is related to our research topic. We will grant more attention on the solutions that also consider the fault tolerance and heterogeneity.

Chen et al [95] proposes a fault tolerant topology control approach (FTTC) which aims to achieve a certain degree of vertex connectivity. In this approach, each node first runs SPAN [6] protocol to construct a connected dominating set (CDS). All nodes in CDS mark themselves. Then each node in the CDS adds necessary nodes and marks them until reaching to desired local vertex connectivity. All marked nodes stay in wakeup state and unmarked nodes go to sleep state. In order to balance energy consumption, a rotation process is included in this approach. Authors also prove that ensuring local vertex connectivity degree also guarantees the global vertex connectivity degree. They perform simulations with different node selection criteria including power based, connection degree based, cut bridge based and report that cut bridge based selection is better for extending the network lifetime. They do not provide a comparison with previous approaches. This work focuses on the flat topologies, whereas our work focuses on the two layered heterogeneous topologies.

Another CDS based reliable topology control protocol for wireless sensor networks called the Poly is proposed by Qureshi et al. [27]. The topology construction algorithm in this protocol is based on the idea of polygons. Subsets of the nodes in the network are arranged so that they form a closed path which provides a reliable topology because there is an alternative path to sink. It is also energy efficient because the nodes that are not in the set of nodes comprising a polygon can go to sleep mode. In addition this protocol does not need the location information of the nodes. Qureshi et al. compare their protocol with existing protocols which are also based on CDS, namely, Energy Efficient CDS (EECDS) [102], CDS-Rule K [34] and A3 [70]. They report that Poly is 120% more reliable than these protocols. However, Poly protocol does not fit to our problem, since it does not consider the k-connectivity of each node to sinks. A polygon structure can only provide 2-connectivity for the nodes in the polygon structure. The nodes that are not in a polygon structure are guaranteed to be 2-connected. In addition this protocol is not developed for layered heterogeneous network topologies.

Borbash et al [77] presents an algorithm for topology control for multihop wireless networks based on the relative neighborhood graph (RNG). Authors introduce a distributed algorithm to compute RNG by using local information and directional information of incoming signals. Each node incrementally adjusts its transmission power to obtain the desired network topology. They also perform simulations to compare RNG with minimum spanning tree MST and minimum radius graph (minR) with respect to transmission radius, hop diameter, node degree

and biconnectivity. They report that transmission radius of RNG approaches to that of MST which has the minimum transmission radius as the number of nodes increases. Simulations also show that hop diameter of RNG is closer to that of minR which achieves the minimum. In addition, RNG and MST node degrees are bounded by a small constant where node degree of minR increases linearly as the number of nodes increases. They find out that approximately %90 of the nodes are in the same biconnected component where minR is always biconnected. This algorithm is for flat topologies and also does not consider the fault tolerance.

Li et al [65] proposes a distributed topology control algorithm based on Minimum Spanning Tree (MST), called Local Minimum Spanning Tree (LMST) for wireless multi-hop networks. According to LMST algorithm each node calculates its local MST by using Prim's algorithm by using the local information gathered from the neighbors reachable with maximum transmission power. In the topology construction phase, a node u adds node v to its neighbor list if and only if node v is in u's local MST and one hop away from u. After the determination of neighbor lists each node adjusts its transmission power that will be sufficient to reach its neighbors in the resulting topology. Bidirectionality of the links is achieved either by deleting all unidirectional links or forcing the unidirectional links to become bidirectional. Authors prove that resulting topology preserves the network connectivity and degree of any node is bounded by 6. Simulation results show that LMST has a small average node degree which helps reducing MAC-level contention and small transmission radius which means a small transmission power is required to keep the network connected. LMST algorithm is developed for homogeneous flat network topologies whereas our work focuses on the two-layered heterogeneous topologies.

Wang et al [50] proposes an algorithm called Distributed Geography-based Fault Tolerant topology control algorithm (DGFT) which produces a bidirectional and strongly connected topology. In this approach deployment area is divided into a set of centric circles with the same center O resulting rings with equal width ω. Each node knows its location and the ring it belongs to. By exchanging HELLO messages each node calculates its reachable neighbor set, and its neighbors in the inner and outer rings. Topology construction is performed by selecting neighbors from the inner rings according to certain initialized weights. Nodes adjust their power to select the logical neighbors. Authors show that average node degrees of the topologies generated by DGFT are larger than that of Dist_RNG [77] and LMST [65] algorithms. They also report that DGFT is more robust against random node failures than Dist_RNG and LMST according to the network fragmentation criterion. Although this work considers the fault tolerance, it is for homogeneous flat network topologies where our solution addresses the two-layered heterogeneous topologies.

Blough et al [16] proposes a distributed, localized and asynchronous topology control mechanism called k-NEIGH which aims to keep the number of neighbors of every node equal to or slightly below a specific value k. k-NEIGH protocol makes use of power adjustment technique after the neighbor determination step where each node in the network selects the k nearest neighbors. Neighbor lists are exchanged between one hop neighbors to find out the bidirectional links. Unidirectional links are discarded in the final topology. Transmission power is set to a value which is needed to transmit to the farthest neighbor for each node. k-NEIGH also includes an optional pruning procedure which investigates if there is a more power efficient path which includes an intermediate node than an existing direct link for all neighbors of a node. If such an intermediate node exists, direct edge is replaced with the edge between the current node and the intermediate node. After this pruning stage, each node adjusts its transmission power according to the resulting pruned neighbor list. Authors

run simulations to find the preferred values for k for different number of nodes which result a connected network with probability > 0.95. They also compare energy efficiency of the topology generated by k-NEIGH with the topology generated by CBTC [72] protocol. Simulation results show that k-NEIGH performs significantly better than CBCT with respect to energy cost. This solution considers the connectivity between sensor nodes in a flat homogeneous network topology whereas our work targets k-connectivity between sensor nodes and supernodes which constitute a two-layered heterogeneous network topology.

Ramanathan et al [71] formulates the topology control of multihop wireless networks as a constrained optimization problem where the objective is to minimize the transmission power subject to keeping the network connected. Authors define the optimization problems Connected MinMax Power (CMP) and Biconnectivity Augmentation with MinMax Power (BAMP) to find a per-node minimal assignment of transmit powers for connectivity and biconnectivity respectively. They present two centralized algorithms CONNECT and BICONN giving optimal solutions for the problems CMP and BAMP for static networks. They also propose distributed heuristic methods for mobile networks namely Local Information No Topology (LINT) and Local Information Link State Topology (LILT). LINT tries to keep the number of active neighbors around specified threshold by dynamically adjusting the transmission power. When the number of neighbors becomes less than the threshold value, it increases the transmission power and when the number of neighbors becomes greater than the threshold value, it decreases the transmission power. LINT does not consider the network connectivity while adjusting the transmission power. In order to prevent the disconnectivity, LILT algorithm is proposed. LILT makes use of the link-state information to recognize and repair the network partitions. Neighbor Addition Protocol (NAP) is triggered whenever an event driven or periodic link-state update arrives. NAP sets the transmission power of the node to maximum in order to establish the connectivity when the network becomes disconnected.

In [71] authors make simulations to compare the throughput, delay, average and maximum transmit power for the topologies generated by CONNECT and BICONN for static networks, LINT and LILT for mobile networks, with topologies that have no topology control. Results show that topologies generated by BICONN provide more throughput but consume more power than CONNECT. LINT and LILT give similar results on throughput which are quite better than no topology control in general. However in dense networks, LILT performs worse than no topology control. CONNECT and BICONN have a little importance for use in real applications due to requirement of global topology information. However, topologies generated by them can be used as a reference since they give the optimal solution for per-node minimality of transmission power. Although LINT and LILT are designed for mobile networks, they can also be used in dynamic environments as in sensor networks for topology control. The algorithms in [71] focuses on the connectivity between sensor nodes and do not directly apply for two-layered heterogeneous sensor networks.

As seen in [71] trying to maintain a fixed node degree can cause partitioning of the network due to increasing or decreasing the transmission range of the nodes without collaboration. Srivastava et al [22] focuses on this problem and identifies the cases that lead to disconnected network and isolated nodes when a fixed node degree approach is applied. They propose two distributed algorithms to improve the connectivity of the fixed node topologies by maintaining critical links when necessary. The proposed algorithms use one hop local information and positions of the nodes and combine unidirectional link information in topology control decisions. We will briefly discuss the proposed algorithms, namely Collaborative Algorithm (CA) and Collaborative Algorithm with Probable Critical Link (CA-PCL).

The main objective of the CA algorithm is to identify the unidirectional links which may arise when a fixed node local topology control approach is used. Having identified the unidirectional links, CA algorithm adds extra links to the local topology graph to convert the unidirectional links to bidirectional ones. In CA algorithm each node starts by running Construct-TCN() procedure to select its topology control neighbors (TCN). First, each node transmit a HELLO message which includes the id of the transmitting node. Any node receiving a HELLO message adds the id of the node in the message to its neighbor list (N_i). Nodes in list N_i is sorted according to increasing distance from the node i. After the identification of the neighbors in the vicinity, each node adds the first k nodes to its TCN list to obtain a fixed node topology. Notice that TCN list contains the unidirectional neighbors. In order to find the bidirectional neighbors, Bidirectional-TCN() procedure is run by each node. In Bidirectional-TCN() each node transmits its TCN list at maximum power. Any node receiving a TCN list, calculates the list of bidirectional neighbors by comparing with its local unidirectional neighbor list(TCN). Finally, Convert-TCN() procedure is run by each node which iterates the TCN list of the nodes in list N_i (list of all neighbors) and adds the nodes to its bidirectional node list (BTCN) which have node i in their TCN lists.

Although the CA algorithm improves the connectivity of the network, in some cases it is still possible that the network stays disconnected. This situation occurs when all the nodes satisfy the desired node degree and they form clusters which are not connected to each other. In order to solve this problem Srivastava et al [22] propose the CA-PCL algorithm which identifies the probable critical links that may be essential for the network connectivity. In Critical-TCN() procedure, critical links are found locally by using maximum transmission power. This is achieved by first detecting the articulation points whose removal makes the network disconnected. A link connecting to an articulation point is considered as a probable critical link and added to the TCN of the link that runs the Critical-TCN() procedure. An articulation point is found by Lune method which is also used to construct the RNG graphs. If a node exists in the $Lune(i, j)$ of the nodes i and j, then the link between i and j is not considered as a critical link because there exists an alternative path between these nodes. If there is no node in $Lune(i, j)$, it means that there may not be other paths that connect these two nodes, so it is marked as a probable critical link. Since this search is made in the local neighborhood, resulting links may not be the global critical links and therefore they are called as probable critical links. For that reason we should also be aware that inclusion of these critical links can cause extra link redundancy.

The simulation results show that the connectivity that CA and CA-PCL provide is significantly higher than that of LINT. The transmission power required for the topologies generated by CA is also lower than that of LINT but CA-PCL has an average maximum transmission power that is greater than LINT. This is due to the Critical-TCN() procedure. In addition, simulation results also show that topologies generated by CA-PCL is more tolerant to node failures than the RNG topologies distributedly generated by Dist-RNG [12] algorithm. The algorithms in [22] are not developed for the heterogeneous two-layered network architectures.

Li and Hou [61],[64] propose two algorithms addressing the k-connectivity problem with the objective of minimizing the maximum transmission radius among all nodes in the network. The first algorithm they introduced is called Fault Tolerant Global Spanning Subgraph (FGSS) which is a centralized greedy algorithm. It is aimed to build k-connected spanning subgraphs based on the well known Kruskal's algorithm. Since FGSS is a centralized algorithm that requires the global knowledge of network, which is difficult to collect and distribute through the network, they also proposed a localized algorithm called Fault Tolerant Local Spanning

Subgraph (FLSS). In this algorithm, each node selects its own set of neighbors and finally adjusts its transmission power based on the local information. They proved that FLSS minimizes the maximum transmission power of nodes among all strictly localized algorithms. The difference between the problem focused in this work and our work is that we try to minimize the total transmission power of the nodes whereas Li and Hou minimize the maximum transmission power. In addition our focus is on the 2-tiered heterogeneous sensor network whereas they primarily focus on the flat homogeneous wireless ad hoc networks. Their objective is to keep k-connectivity between any two nodes but we are concerned with providing k-connectivity from each sensor to the set of supernodes.

For multi-hop wireless networks with nonuniform transmission ranges, Li et al. [63] propose two algorithms, namely, Directed Relative Neighborhood Graph (DRNG) and Directed Local Spanning Subgraph (DLSS). Both algorithms use locally collected information to construct the final network topology. These algorithms have three phases, namely, information collection, topology construction and bidirectional link construction. The last phase is optional that is used only when a bidirectional topology is required. In the information collection stage each node collects the information of its visible neighborhood. This is achieved periodically broadcasting Hello messages that include the id, the maximal transmission power and the position of the node. In the topology construction phase the DRNG algorithm constructs a directed relative neighborhood graph whereas the DLSS builds a directed local spanning subgraph. Since each node determines its neighbors independently, some links in the final topology may be uni-directional. So, if required, the optional third phase is run to obtain a bidirectional topology either by applying link addition or removal methods. The simulation results show that the DLSS performs better than the DRNG in terms of average radius, average node degree and average link length. The notion of heterogeneity in Li et al.'s work is different than ours because we focus on a two-layer topology where layers are composed of different type of nodes. For that reason, the problem they address is different than ours and the solutions they provide cannot be directly applied to our problem. In addition, they do not consider the fault tolerance of the resulting topologies.

A prominent work on fault-tolerant topology control for wireless multi-hop networks is proposed by Saha et al [28]. Their algorithm is a distributed one assigning the nodes approximately minimum power using locally collected data and it results a network topology where any two nodes in the global network is connected by k-vertex disjoint paths provided the initial graph is k-connected. The algorithm has three phases, namely, information collection and finding the vicinity topology, construction of the minimum-power vicinity topology and finally transmission power assignment. In the first phase the vicinity information is collected via Hello messages. The critical phase is the second one where the k-vertex disjoint paths to each node in the vicinity are found according to the optimality criteria including the total cost of the path, maximum edge cost in the path and number of hops. The authors use a function of these three parameters to select among the candidate disjoint paths which are found using the shortest path algorithm. However to determine the weight of these parameters for different topologies they have to carry out some experiments which decreases the practicality of the algorithm. They use a greedy approach which selects the shortest path between two nodes in the vicinity and remove the nodes that are in this path and continue finding the next shortest path in the remaining set of nodes. This approach does not result the optimum solution as expected. In the last stage of the algorithm, the transmission ranges of each node is adjusted according to the topology constructed in the second phase. The difference between this work and ours is that they focus on the k-connectivity between any two nodes in the network

whereas we focus on the k-connectivity between sensor nodes and the set of supernodes in the network. This difference dramatically changes the problem and the possible set of solutions.

Cardei et al [51] addresses fault-tolerant topology control in a heterogeneous wireless sensor network with two layer network architecture where the upper layer consists of several resource-rich supernodes which are used for data relaying on top of a layer of large number of energy constrained wireless sensor nodes. They also introduce the problem that we will also focus in this work called the k-degree Anycast Topology Control (k-ATC) problem which aims to adjust transmission ranges of the sensor nodes in order to achieve k-vertex supernode connectivity and minimize the maximum transmission power of the sensor nodes.

The main objective of the work is to maintain k vertex independent paths from the sensor nodes to the set of supernodes with minimum transmission power. Authors propose a greedy centralized algorithm (GATC) which results the optimal solution and a distributed algorithm (DATC) that provides k-vertex supernode connectivity by incrementally adjusting the sensor transmission range. They perform simulations to show the performance of the GATC and DATC algorithms with various parameters. We will compare our solutions with the solutions presented in this work. Now we will briefly discuss the algorithms the GATC and DATC introduced in [51].

The algorithm GATC starts with a graph G which is already k-vertex supernode connected which means there are at least k vertex disjoint paths from each sensor node to at least one supernode. The GATC first makes some transformations on G to obtain a reduced graph G_r where all supernodes are replaced with a single root node. In this reduced graph all edges between sensor nodes are preserved but the edges between sensor nodes and supernodes are switched to an edge between the sensor nodes and the root node. If a sensor node is connected to more than one supernodes in G, only one edge is added in G_r which has the lowest cost. Another transformation is done to obtain the directed version of the reduced graph G_r^-. In this transformation for all undirected edges between sensor nodes in G_r, two directed edges are added in G_r^-. An edge between a sensor node and a supernode is changed with one directed edge, from sensor node to supernode. After these transformations, authors show that any heterogeneous WSN is k-vertex supernode connected if and only if corresponding reduced graph is k-vertex supernode connected to the root.

After the transformations, the GATC sorts all the edges in G_r^- according to their costs which are defined in terms of length of the edges. Edges are examined in decreasing order of cost and an edge (u, v) is removed if node u remains k-vertex connected to the root. After examining all the edges and removing the edges which do not affect the k-connectivity, the GATC adjusts transmission power of each node according to the new set of neighbors in the resulting topology.

The GATC algorithm is mostly of theoretical importance since it is not practical to apply it for large scale wireless sensor networks due to the requirement of global topology knowledge. However, it is a good point of reference for comparison with other algorithms since it gives the optimal solution. We will be using the GATC algorithm for comparison with the algorithms we propose.

Cardei et al [51] proposes another algorithm called DATC which is a distributed and hence a more practical solution for k-ATC problem. This algorithm requires only 1-hop neighborhood topology information which may also be extended to h-hop as well. In the DATC algorithm, each node i starts with a transmission power p_i^{min}, which is required to reach the first k neigh-

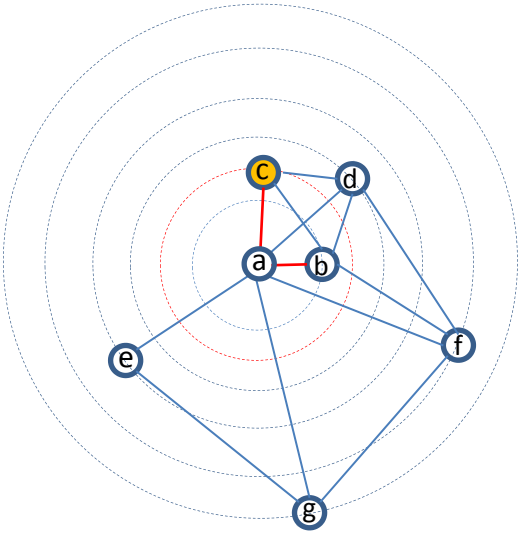


Figure 3.1: Initially node a's transmission range is set as to reach its k closest neighbors.

bors. If p_i^{min} is equal to p_i^{max} then no improvement can be done for node i, where p_i^{max} is the maximum transmission power for node i. Otherwise, transmission power is incrementally increased for node i, to reach at least one neighbor from reachable neighborhood and it is checked to see if all nodes in the reachable neighborhood are directly reachable from i or k-vertex connected to node i. If it is so, then the final transmission power is found for node i, otherwise the iterative process is repeated until the transmission power is equal to p_i^{max} in the worst case. In a 1-hop neighborhood, the chance of finding k-vertex disjoint paths between a node and any of its neighbors is relatively low, especially for sparse networks. This is due to the limited number of alternative paths in such a small neighborhood. This situation causes incrementing the transmission power to cover the nodes that are not reachable through k-vertex disjoint paths in the 1-hop neighborhood. To obtain lower transmission power values, an h-hop version of DATC is also proposed, but this causes the message complexity to increase rapidly, resulting in significantly more energy consumption than 1-hop version.

The objective of the DATC algorithm is to make sure that any neighbor u, in the reachable neighborhood of any node v is either directly reachable from node v or there are k-vertex disjoint paths from v to u. This condition ensures the k-vertex supernode connectivity of the resulting topology. This is proved by the authors in [51].

A sample run of the DATC algorithm for $k = 2$ using a 1-hop neighborhood local information is shown in Figures 3.1 through 3.5. In Figure 3.1, node a and its 1-hop neighbors are shown. Circles centered at node a depict the distance of each neighbor to node a and thus the required transmission range for reaching a neighbor on a circle. Initially node a adjusts its transmission range as to reach its $k = 2$ closest neighbors, namely, b and c. It is because for k-vertex supernode connectivity each node in the network must have at least k neighbors. The circle that represents the initial transmission range of node a is shown in red in Figure 3.1.

The next step in the DATC algorithm is to check whether the next closest neighbor, namely d, can be reached via two disjoint paths from node a using only the nodes that are in the current transmission range of node a which are the nodes b and c. As seen in Figure 3.2 node d can

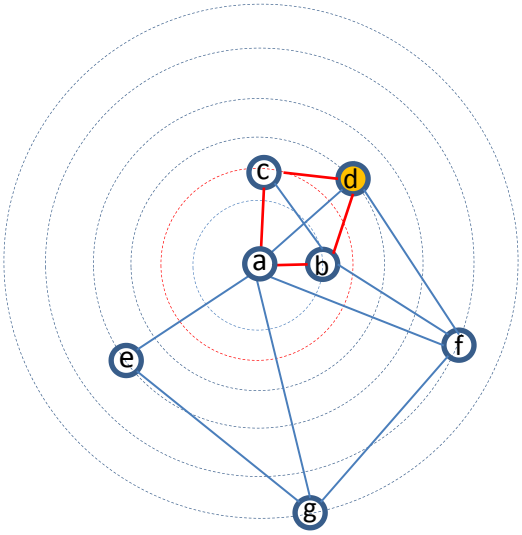


Figure 3.2: Node a has two disjoint paths to node d, so its transmission range stays the same.

be reached via two disjoint paths $a-c-d$ and $a-b-d$, so node a does not need to increase its current transmission power because it does not have to directly reach node d. There node a's transmission range stays the same in this step.

The DATC algorithm proceeds by checking the remaining 1-hop neighbors that are reachable with maximum transmission power. So the next neighbor is node e. With the current transmission range, node e is not reachable directly. With the current local 1-hop topology of node a, there does not exist any paths to node e from node a. Therefore, the only option is to increase the current transmission range of node a to reach node e directly. Figure 3.3 shows the transmission range of node a after this step. Note that, after changing the transmission range, node a have to notify its neighbors by broadcasting a message that contains the new transmission range, because it affects the local neighborhoods of the neighboring nodes. Actually, every node has to maintain transmission range and location of each neighbor that are reachable with maximum transmission power. Otherwise, a node cannot compute the connectivity between any two 1-hop neighbors.

The next closest neighbor is node f which can be reached via two disjoint paths, namely, $a-d-f$ and $a-b-f$ as shown in Figure 3.4. So there is no need to increase the transmission power in this step.

The last node to consider is node g. Node a can compute at least two disjoint paths to node g using the local topology information as shown in Figure 3.5. Again in this step the transmission range of node a stays the same. This is the final transmission range for node a. Now it is guaranteed that any 1-hop neighbor of node a can be reached directly or via at least 2 disjoint paths from node a. This condition ensures the k-vertex supernode connectivity of the resulting topology as proved in [51].

The weak point of the DATC algorithm is that it requires testing for k-connectivity in each power increment step for each neighbor. This test requires complete topological view of 1-hop neighborhood which is not very expensive to obtain but it is a very small set of vertices to check k-connectivity. For 1-hop neighborhood most of the time, there will not be k-vertex

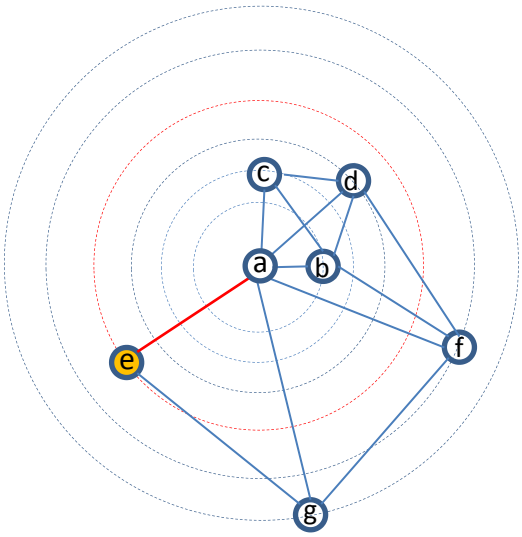


Figure 3.3: Node a's transmission range is increased to reach node e.

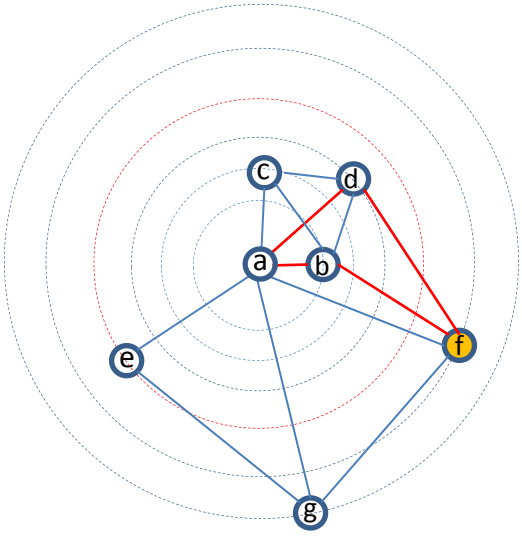


Figure 3.4: Node a has two disjoint paths to node f, so its transmission range stays the same.

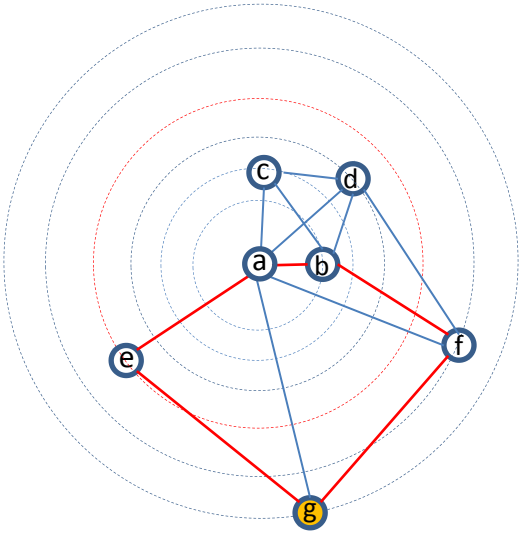


Figure 3.5: Node a has two disjoint paths to node g, so its transmission range stays the same.

disjoint paths between a node and its neighbor and the algorithm will result with p_i^{max}. In order to obtain lower transmission power values, authors also propose an h-hop version of DATC but in this case the message complexity of the algorithm will increase which will cause more energy consumption according to 1-hop version.

Our algorithm differs from DATC by the approach that we adopt for discovering vertex disjoint paths. In DATC, each node starts with a minimal set of neighbors and minimal power level. The power level is increased incrementally and only the paths from the neighborhood that are reachable with that power level can be discovered. The nodes outside of the reachable neighborhood are totally unknown to the node performing discovery and thus they are out of the search scope for discovering paths. This is an important limitation for DATC because it has a low chance to find k-vertex disjoint paths for its neighbors in its reachable neighborhood, which typically has nodes that are in 1 or 2 hops distance from the node performing the discovery. In contrary to DATC, in our algorithm, a sensor node can discover paths including nodes outside of its reachable neighborhood. This is achieved by storing full path information from supernodes to sensor nodes in local information tables. In this way, the DPV algorithm has more chance to discover better k-disjoint paths than DATC. Another difference of our algorithm from DATC is that, we decrease the power level only after deciding the final topology. During path discovery in the DPV algorithm, nodes operate with maximum power, thus, increasing the likelihood of discovering more paths than DATC. Our simulation results are in conformity with this discussion.

CHAPTER 4

PROPOSED WORK: DISJOINT PATH VECTOR ALGORITHM

4.1 Overview

In this thesis, we propose an algorithm for fault tolerant topology control in two tiered heterogeneous wireless sensor networks consisting of resource rich supernodes and simple sensor nodes which have batteries with limited capacity. In order to obtain fault tolerant topologies we will focus on k-vertex supernode connectivity which is defined in by Cardei et al in [51]. In a k-vertex supernode connected topology, each sensor is connected at least one supernode by k vertex disjoint paths. Such topologies are tolerant to k-1 node failures at worst case. In other words, sensor nodes remain connected to supernodes when up to k-1 sensor nodes are removed from the network.

We propose an algorithm named Disjoint Path Vector (DPV) algorithm which is based on the observation that we can remove the edges with the neighbors which are not on one of the k-vertex disjoint paths to supernodes. In order to achieve this, we need to determine which neighbors are on one of such paths and which are not. We will show how the DPV algorithm enables us to find a superset of the required vertices in order to guarantee the k-vertex supernode connectivity.

The objective of DPV is to minimize the total transmission power of the nodes in the network. The DPV algorithm is run by each sensor in a distributed manner with locally collected data. Using this data each node computes the set of required neighbors that guarantees k-vertex supernode connectivity. The changes in the neighbor lists are exchanged in an efficient way to keep the message overhead small. Having found the required neighbors, each node removes its edges which are not connected to a required node in coordination with its neighbors. Then to save energy, we use power adjustment technique to decrease the transmission range of the sensor nodes so as to reach to the furthest node in the new set of neighbors.

The DPV is a distributed algorithm executed by each sensor node in the network. Each node makes use of the topology information in the local 1-hop neighborhood. The paths to the supernodes are explored by using messages initiated by the supernodes that carry the traversed path information from a supernode to a sensor node. Global network topology information is not required by the DPV algorithm.

4.2 Sample Scenario

Consider a weapon system composed of artillery units installed on the country border for homeland security. This weapon system is integrated with a 2-tiered heterogeneous wireless sensor network where supernodes are located on the artillery units. Artillery units are steerable so they can head toward the targets whose locations are known. The data gathered by sensor nodes are forwarded to supernodes where this data is evaluated and a decision on whether to open fire or not is taken. The sensor nodes are deployed in the area to detect potential intrusion activities. When an activity is detected by a sensor node, it forwards this data to supernodes for evaluation and execution of proper action. In this scenario, supernodes have more power capacity and more sophisticated technical facilities (e.g. camera, IR sensors) which enables accurate target classification compared to ordinary sensor nodes. So they are more expensive than ordinary sensor nodes and number of supernodes is much lower than the ordinary ones.

In this scenario, it is very critical to deliver the data sensed by the sensors to supernodes. It is also common to lose some sensor nodes because of energy depletion, harsh environmental conditions, hostile activities of intruders or artillery strikes. Therefore a network topology which is tolerant to a certain number of node failures is required. So it is desired that a sensor node should have more than a certain number of independent paths to supernodes. It is also possible to loose supernodes due to intruder attacks. It would be better to have connections with more than one supernodes in order to remain connected to at least one super node in case of a supernode loss or failure. Since the sensor nodes are energy constrained, it is essential to have an energy efficient topology control mechanism in order to obtain a longer network lifetime.

Our work targets the topology control problem designated by the above sample scenario. We are looking for a distributed, fault tolerant and an energy efficient topology control algorithm for 2-tiered heterogeneous wireless sensor networks.

4.3 Network Model

We consider a heterogeneous WSN consisting of M supernodes and N sensor nodes, with $M << N$. N sensor nodes are randomly distributed in the 2D plane. M supernodes are deployed at known locations manually. We are interested in sensor-sensor and sensor-supernode communications only. We do not model the supernode-to-supernode communications. We assume that supernodes are not energy constrained so they can directly communicate with a base station or they can send the data collected from sensors to other supernodes if necessary. We also assume that all supernodes are identical which means delivering a message to any of the supernodes is sufficient and considered as a successful delivery. In the initial network topology each energy constrained sensor node has the transmission range R_{max}. We assume that supernodes have transmission ranges long enough to communicate with the base station and any other supernode in the network.

We represent the initial network topology with an undirected weighted graph $G = (V, E)$ where $V = \{v_1, v_2, ..., v_N, v_{N+1}, ..., v_{N+M}\}$ is the set of nodes and $E = \{v_i, v_j \mid dist(v_i, v_j) < R_{max}\}$ is the set of edges where $dist(v_i, v_j)$ depicts the distance between nodes v_i and v_j. The first N nodes in V are the energy constrained sensor nodes and the last M nodes are the resource rich

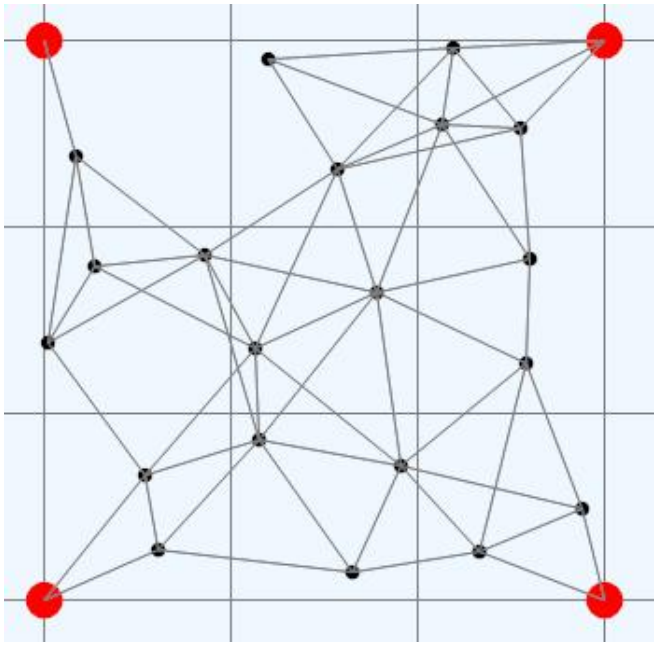


Figure 4.1: A Sample 2-Tiered WSN.

supernodes.

An example network topology is given in Figure 4.1 with 4 supernodes and 20 sensor nodes deployed in a 300 x 300 unit square area where R_{max} is set to 110 units.

4.4 Problem Definition

We aim to construct and maintain a k-vertex supernode connected network topology in order to route the data sensed by the sensor nodes to the supernodes for two tiered wireless sensor networks with network model given in section 4.3. We model topology control as a transmission range assignment problem for each sensor node in the network. The objective is to minimize the assigned transmission power for all sensors while maintaining k-vertex disjoint paths from each sensor to the set of supernodes. In this topology, each node should only keep necessary set of neighbors which satisfy k-vertex supernode connectivity. Each sensor node in the network must be connected to at least one supernode with k vertex disjoint paths.

Before formulating the problem we give some necessary definitions:

Definition 1 (Vertex disjoint paths): Set of paths with common end points that have no other vertices in common. In this work all disjoint paths are vertex disjoint paths unless otherwise stated.

Definition 2 (k-vertex supernode connectivity):A WSN is k-vertex supernode connected if, for any sensor node $n \in V$, there are k pairwise vertex disjoint paths from n to one or more supernodes. In other words, a WSN is k-vertex supernode connected if the removal of any k - 1 sensor nodes does not partition the network [51].

We can now define the problem as follows:

Given a k-vertex supernode connected WSN with M supernodes and N energy-constrained sensor nodes that can adjust their transmission range up to a predefined constant R_{max}, determine the transmission range of each sensor such that the total transmission power is minimized and the resulting topology is still k-vertex supernode connected. Next, we state the problem definition more formally.

Definition 3(Problem definition): Given an undirected graph $G = (V, E)$ where V is the set of all vertices and E is the set of all edges and given two disjoint subsets of vertices namely $M \subset V$, $N \subset V$ and $M \cap N = \emptyset$, where there exists at least k-vertex disjoint paths from each vertex $v \in N$ to the set of vertices M, find set of edges F such that $G(V, E - F)$ satisfies the following:

1. There exist at least k-vertex disjoint paths from each vertex $v \in N$ to the set of vertices M.
2. $\sum_{i=1}^{N} p_i$ is minimized, where p_i is the weight of the maximum weighted edge $\in (E - F)$ of $v_i \in N$.

4.5 Disjoint Path Vector Algorithm for k-vertex Supernode Connectivity

Our algorithm named Disjoint Path Vector algorithm (DPV) efficiently assigns transmission power levels for sensor nodes while preserving the k-vertex supernode connectivity of the network in a distributed and localized manner. It consists of five main stages as shown in 4.5.1. The first stage is path information collection, initiated by the supernodes through $INIT$ messages. An $INIT$ message contains the ID of the supernode that created the message and can only be transmitted by a supernode. These messages are received by the sensor nodes in the network and each receiver node updates its local path information according to that data. Sensor nodes transmit $PathInfo$ messages when an update occurs in their local disjoint path lists. Upon receiving a $PathInfo$ message, each sensor node extracts the disjoint paths to the supernodes by using its local data and the path information received from the $PathInfo$ message. If the incoming $PathInfo$ message improves the cost of the disjoint paths, the message is forwarded by adding the updated path information. The cost of a set of disjoint paths is defined as the maximum of the costs of the paths in the set. When improvement is not possible, the first stage of the algorithm ends and the second stage starts in which each node calculates its required neighbors using the locally found set of disjoint paths as the input.

In the third stage, for each disjoint path a $NOTIFY$ message is initiated by each sensor node and forwarded along the nodes in the path. Neighbor nodes in a disjoint path mark each other as a required neighbor. In the fourth stage, neighbors that are not selected as required are removed from the neighbor list. In the final stage, each node adjusts its transmission power level to reach its farthest required neighbor according to the new topology that resulted from the removal of the non-required nodes. We now give a detailed explanation for each stage of the DPV algorithm.

After deployment of sensor nodes and supernodes, the first stage of the algorithm is initiated by INIT messages which are broadcasted by super nodes through the network. An INIT message contains the ID of the supernode which created the message. INIT messages can only be transmitted by supernodes. Any sensor node which receives an INIT message, updates its

Algorithm 4.5.1 Five Main Stages of DPV

Input: Set of all neighbors

Output: Superset of required neighbors

1: Collect path information and calculate disjoint paths (Procedures 4.5.2 and 4.5.3)
2: Calculate the set of required neighbors (Procedure 4.5.4)
3: Notify the nodes in the disjoint paths and update the required neighbors (Procedure 4.5.5)
4: Remove the neighbors that are not marked as required in coordination with 1-hop neighbors
5: Reduce power level such that it is sufficient to reach the farthest neighbor

Table4.1: Path Information Table at node y. Node y holds 3 paths to supernodes.

Supernode	Path	Path Cost
B	y-B	230
A	y-z-x-A	250
C	y-z-C	300

local path information table by adding an entry for the path to super node which sent the INIT message. In this special case such an entry will only include the ID of the supernode and the cost of the link between the receiver node and supernode. The cost will be the length of the link. In general, an entry in the local path information table in a sensor node will include a path to a supernode and the cost of that path which is selected to be as the length of the longest link of the path. Paths are sorted according to the cost in ascending order in the local path information table. An example of the path information table is given in Table 4.1.

Details of the distributed algorithm run by each sensor node in the information collection stage is given in Algorithm 4.5.2. The DPV algorithm notations are introduced in 4.2.

Procedure 4.5.2 Path Information Collection in DPV

Input: I, L, k
Output: D
1: $T \leftarrow \emptyset$;
2: **for all** received PathInfo message I **do**
3: **if** I.Sender is a supernode **then**
4: $r \leftarrow$ new Path(I.Sender);
5: **if** $r \notin T$ **then**
6: $T \leftarrow T \cup r$;
7: Transmit PathInfo(T);
8: **end if**
9: **else**
10: $D \leftarrow$ MIN_DIS_SET(T); (Procedure 4.5.3)
11: $c \leftarrow$ Cost(D);
12: $U \leftarrow I.T \cup T$;
13: Sort(U);
14: $T' \leftarrow \{p_i \in U \mid i <= L\}$;
15: $D' \leftarrow$ MIN_DIS_SET(T'); (Procedure 4.5.3)
16: $c' \leftarrow$ Cost(D');
17: **if** $c' < c$ **then**
18: $T \leftarrow T'$;
19: Transmit PathInfo(T);
20: **end if**
21: **end if**
22: **end for**

Having updated the local path information table, a sensor node creates and transmits a $PathInfo$ message, which contains its ID and path information table. A $PathInfo$ message can only be

Table4.2: DPV Notations.

T and T'	Set of local paths
D and D'	Set of disjoint paths with the minimum cost
U	Union of two sets of paths
I	Received PathInfo message
L	Maximum number of paths to be stored
c and c'	Cost of disjoint paths
r, t, p and p_i	Variables referencing paths
k	Degree of disjoint connectivity
R	Set of required neighbors
S	Set of first k disjoint paths
B	Set of neighbors of a node
Φ	Incoming Notify message
n	Variable referencing a node
Q	Set of all subsets of T of size k
q	A subset of T of size k
q_{min}	Minimum cost subset of T of size k
d	A temporary boolean variable

sent by sensor nodes where an $INIT$ message can only be sent by super nodes. $INIT$ messages contain only ID of the sending supernode where a $PathInfo$ message includes paths to supernodes and their costs. Sensor nodes transmit $PathInfo$ messages to their reachable neighborhood using maximum power. Any sensor node that receives a $PathInfo$ message calculates the union of the existing and the received paths via $PathInfo$ message. Then, the set of disjoint paths with the minimum cost is calculated for the existing paths and for the newly calculated union. Notice that the size of calculated disjoint sets is at most k, because we need only k disjoint paths. If the new cost is lower than the current cost, the local path information table is updated. The algorithm for finding disjoint paths run by each sensor node is given in Procedure 4.5.3. The resulting list of disjoint paths is stored for use in the next stages of the DPV algorithm.

Any sensor node that updates its local path information table transmits a $PathInfo$ message that contains the updated path information table. Note that if a $PathInfo$ message does not cause an update in the receiver node's disjoint paths list, then no $PathInfo$ message is sent. This condition ensures the termination of the path information collection stage. If further improvement on the path costs is not possible, transmission of $PathInfo$ messages stops. The DPV algorithm is guaranteed to converge because there is an upper bound on the number of $PathInfo$ messages that any sensor can send during the path information collection stage. A $PathInfo$ message is only transmitted if an update occurs in the local set of disjoint paths. An update can only occur if a lower-cost disjoint path is discovered via a received $PathInfo$ message. At the worst case, there can be $|E|$ different-cost disjoint paths from a sensor node to the set of supernodes where $|E|$ is the number of edges in the network given that cost of a path is defined as the cost of the longest edge in that path. In the worst scenario, these $|E|$ different-cost disjoint paths can be discovered in the decreasing order of path cost. Suppose

Procedure 4.5.3 Finding Disjoint Paths to Supernodes (MIN_DIS_SET)

Input: T and k
Output: D

```
1:  D ← ∅;
2:  if |T| > k then
3:     Q ← { q ⊂ T | |q| = k };
4:     c ← ∞;
5:     q_min ← ∅;
6:     for all q ∈ Q do
7:        if q consists of disjoint paths then
8:           if Cost(q) < c then
9:              c ← Cost(q);
10:             q_min ← q;
11:          end if
12:       end if
13:    end for
14:    D ← q_min;
15: else
16:    for all t ∈ T do
17:       d ← true;
18:       for all q ∈ D do
19:          if t ∩ q ≠ ∅ then
20:             d ← false;
21:             break;
22:          end if
23:       end for
24:       if d = true then
25:          D ← D ∪ t;
26:       end if
27:    end for
28: end if
```

that each discovery results an update, then the node will at most send $|E|$ $PathInfo$ messages. Since a graph can have at most $O(N^2)$ edges, where N is the number of nodes, we conclude that any sensor node can transmit at most $O(N^2)$ $PathInfo$ messages.

We set an upper limit on the number of hops a $PathInfo$ message can traverse during the path information collection stage. This is the $PathInfo$ message's time-to-live (TTL) value, and it directly affects the number of hops in the resulting paths. This value is set at the supernode that sends the $INIT$ message and it is included in every $PathInfo$ message. When a sensor node receives a $PathInfo$ message that has reached the TTL value, no further $PathInfo$ message is created even if an update occurred as a result of this $PathInfo$ message. This approach ensures that paths with a length above a certain limit are not considered in the disjoint path calculation; it can be considered as a prepruning that occurs before determining the set of disjoint paths with minimum cost.

Figures 4.2 through 4.11 show a sample run of the path information collection stage of the DPV algorithm on a sample network topology with local path information tables and disjoint paths for each sensor node.

As in Figure 4.2, path information collection is initiated by the $INIT$ message sent by supernode A. Because supernode A has only one neighbor x, this message is only received by sensor node x. Upon receiving the message, node x calculates the path $x - A$ and its cost (90) and stores it in its local path information table. In this case there is only one path in node x's table and thus this path is the only possible disjoint path for node x. As the disjoint paths of node x are updated, the node will create a $PathInfo$ message containing its path information table and send it to its 1-hop neighbors. Node x has only one neighboring sensor node, namely z, hence node x's $PathInfo$ message will only be received by node z.

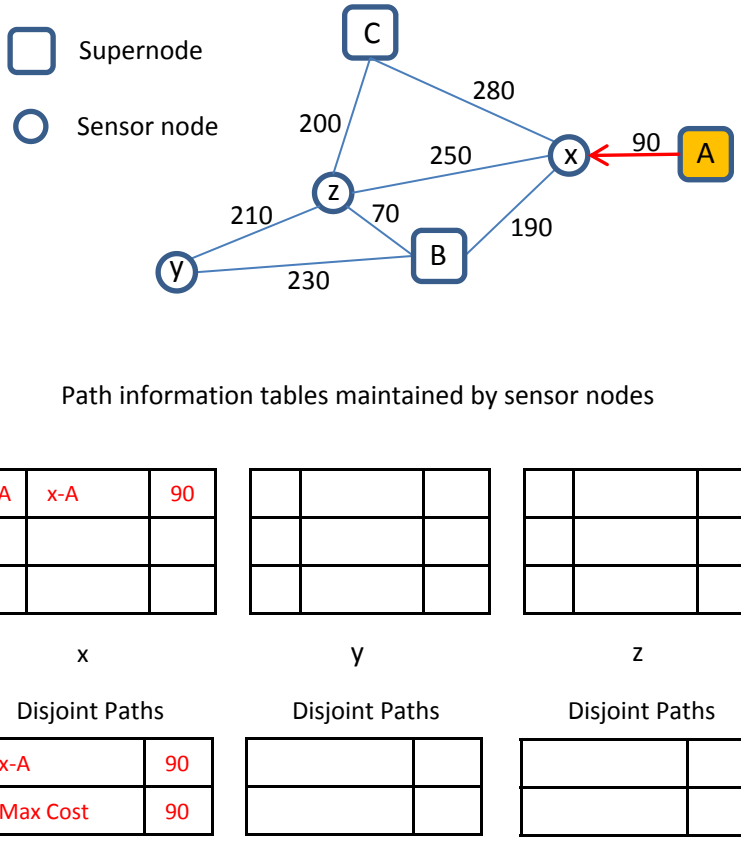
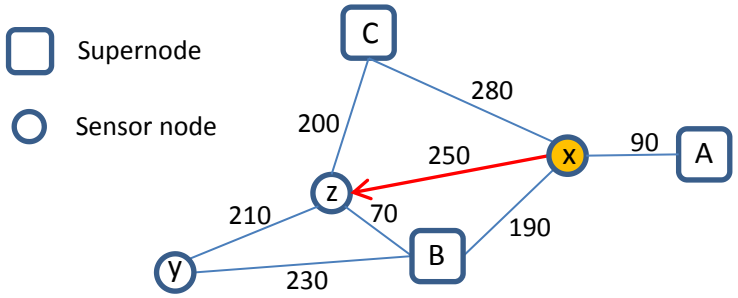


Figure 4.2: Path information tables after supernode *A* transmits an *INIT* message.

Figure 4.3 shows the transmission of node x to node z. The path information tables and disjoint paths present the updated and final situation after node z receives the $PathInfo$ message sent by node x and corresponding path calculations. Because there was only one path $(x - A)$ in node x's path information table, the $PathInfo$ sent by node x contains only this path and its cost. After receiving this information, node z calculates the path from itself to supernode A and its cost. The resulting path and its cost are stored in node z's path information table. The only path in this table is also the only disjoint path that node z currently has. Since an update occurred in its disjoint paths, node z will transmit a $PathInfo$ message to its 1-hop neighbors to notify them of this update. Node z has two neighbors, namely x and y, but node z will transmit its $PathInfo$ message only to node y, because the previous message that caused the update was sent by node x, so there is no need to send this message to node x. In general, a sensor node does not send a $PathInfo$ message to the neighbor from which it had received the last $PathInfo$ message.



Path information tables maintained by sensor nodes

A	x-A	90

x

y

A	z-x-A	250

z

Disjoint Paths

x-A	90
Max Cost	90

Disjoint Paths

Disjoint Paths

z-x-A	250
Max Cost	250

Figure 4.3: Path information tables after node x transmits a $PathInfo$ message to node z.

Figure 4.4 shows the transmission of the $PathInfo$ message from node z to node y. This message contains path $z - x - A$ and its cost 250. Before this message was received, there was no path in node y's path information table, therefore node y updates its table by calculating the path for itself. Note that path costs are determined by the most costly link in the path. Here, the most costly link in path $y - z - x - A$ is $z - x$, with a cost of 250, and thus the path cost is also 250. This is again the only disjoint path for node y. Although node y has an updated

disjoint path list now, it does not transmit a *PathInfo* message because it was node y's only neighbor (node z) that caused this update. Therefore, node y does not make any transmission in this case.

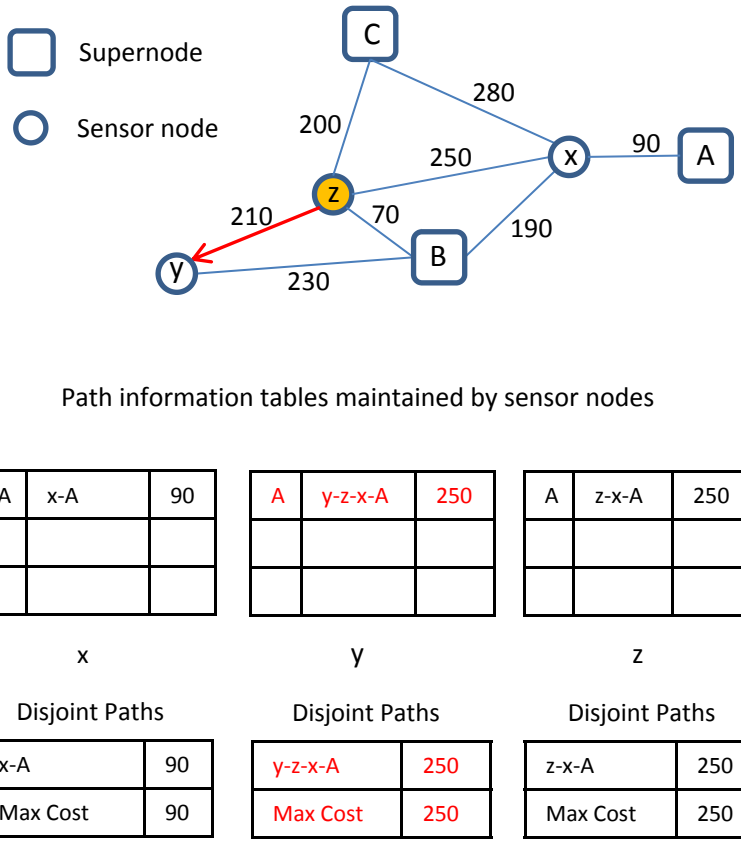
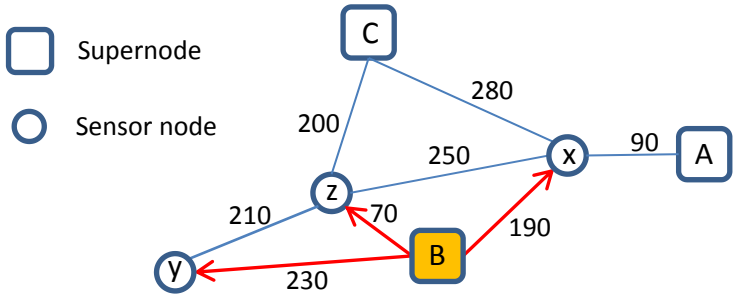


Figure 4.4: Path information tables after node z transmits a *PathInfo* message to node y.

Transmission of INIT message from supernode B is shown in Figure 4.5. Supernode B has 3 neighbors (x, y, z) and all of them receive this INIT message and update their path information tables accordingly. All updated paths and disjoint paths are shown in red after the update. All 3 sensor nodes, insert a new path into their path information tables which includes a direct link to supernode B. Calculation of disjoint paths results 2 disjoint paths for each node as shown in Figure 4.5. Since all 3 nodes update their list of disjoint paths, all of them need to transmit PathInfo messages to their one hop neighbors. In this sample run, we neglect the order of the PathInfo messages sent, we simply follow the alphabetical order of the node names. Note that the intermediate steps of the path information collection stage are dependent on the order of the messages transmitted. We will continue with the node z's transmission in the next step.

In Figure 4.6 we see the results of PathInfo transmission of node x to node z. This transmission brings a new path for node z that is $z - x - B$. However this path is has the same cost with an already existing path and thus it does not make an improvement on the cost of the disjoint paths. Therefore node z does not need to make a PathInfo message transmission for this reception. However node z had already updated its disjoint paths after the supernode B's



Path information tables maintained by sensor nodes

A	x-A	90
B	x-B	190

x

Disjoint Paths

x-A	90
x-B	190
Max Cost	190

B	y-B	230
A	y-z-x-A	250

y

Disjoint Paths

y-B	230
y-z-x-A	250
Max Cost	250

B	z-B	70
A	z-x-A	250

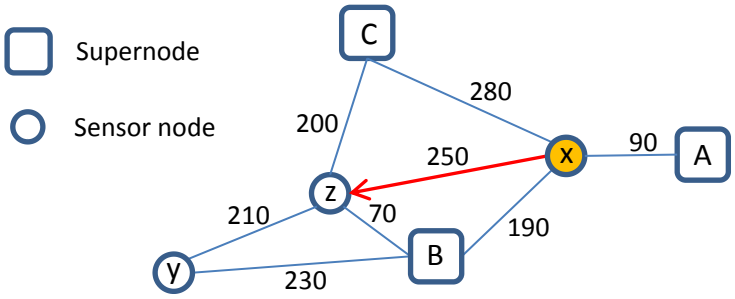
z

Disjoint Paths

z-B	70
z-x-A	250
Max Cost	250

Figure 4.5: Path information tables after supernode *B* transmits an *INIT* message.

transmission in the previous step (Figure 4.5). So the next step continues with the transmission of node z.



Path information tables maintained by sensor nodes

A	x-A	90
B	x-B	190

x

Disjoint Paths

x-A	90
x-B	190
Max Cost	190

B	y-B	230
A	y-z-x-A	250

y

Disjoint Paths

y-B	230
y-z-x-A	250
Max Cost	250

B	z-B	70
A	z-x-A	250
B	z-x-B	250

z

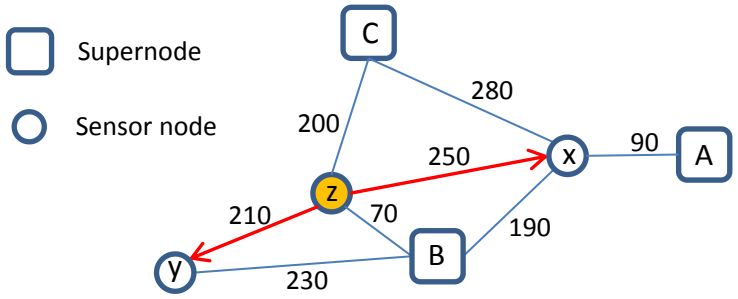
Disjoint Paths

z-B	70
z-x-A	250
Max Cost	250

Figure 4.6: Path information tables after node x transmits a $PathInfo$ message to node z.

Node z transmits its PathInfo message containing its path information table to its neighbors x and y. Node x inserts a new path namely $x - z - B$ to its path information table and node y adds 2 new paths $y - z - B$ and $y - z - x - B$. The update that occurred in node x's path information table does not cause an update in the disjoint path list because node x has already 2 disjoint paths both of which are cheaper than the last added path. Note that in this sample run, k value is 2 so we only keep 2 disjoint paths where the total cost is minimum. However we see a different picture for node y. The path $y - z - B$ has the cost 210 and it is the cheapest path among all paths in the path information table of node y. Therefore it causes an update in the disjoint paths of node y. The path $y - z - x - A$ with cost 250 is removed from disjoint path list and $y - z - B$ with cost 210 is added instead. Node y does not need to make a PathInfo transmission to node z regarding to this update because it is already caused by node z but still it has to transmit a PathInfo message associated with the update caused by the supernode B's transmission depicted in Figure 4.5.

Figure 4.8 illustrates the case where node y makes a PathInfo transmission after the update caused by the supernode B's INIT message transmission. As a result of this PathInfo message node z finds out the path $z - y - B$ with cost 230 and it updates its path information table and



Path information tables maintained by sensor nodes

A	x-A	90
B	x-B	190
B	x-z-B	250

x
Disjoint Paths

x-A	90
x-B	190
Max Cost	190

B	y-z-B	210
B	y-B	230
A	y-z-x-A	250
B	y-z-x-B	250

y
Disjoint Paths

y-z-B	210
y-B	230
Max Cost	230

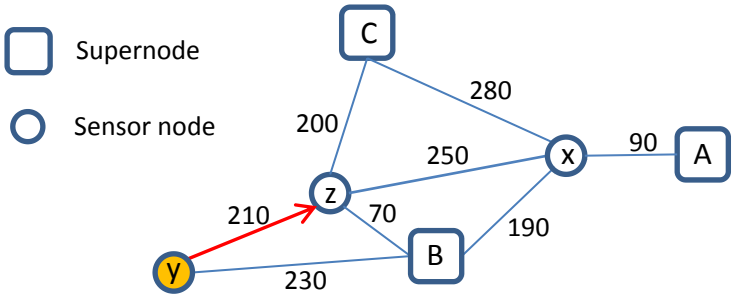
B	z-B	70
A	z-x-A	250
B	z-x-B	250

z
Disjoint Paths

z-B	70
z-x-A	250
Max Cost	250

Figure 4.7: Path information tables after node z transmits $PathInfo$ message to nodes x and y.

disjoint paths. The more costly path $z - x - A$ is replaced with the cheaper path $z - y - B$ as seen in Figure 4.8. Since node z's disjoint paths have been updated, now node z will transmit a PathInfo message to node x. Again, PathInfo message will not be sent to node y because it was the node that sent the last PathInfo message which caused this update.



Path information tables maintained by sensor nodes

A	x-A	90
B	x-B	190
B	x-z-B	250

x
Disjoint Paths

x-A	90
x-B	190
Max Cost	190

B	y-z-B	210
B	y-B	230
A	y-z-x-A	250
B	y-z-x-B	250

y
Disjoint Paths

y-z-B	210
y-B	230
Max Cost	230

B	z-B	70
B	z-y-B	230
A	z-x-A	250
B	z-x-B	250

z
Disjoint Paths

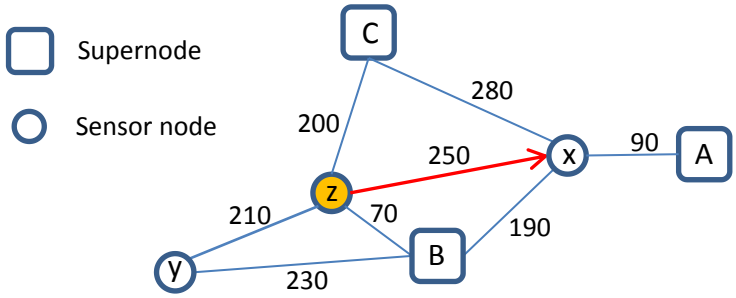
z-B	70
z-y-B	230
Max Cost	230

Figure 4.8: Path information tables after node y transmits a $PathInfo$ message to node z.

As shown in Figure 4.9 node z's transmission to node x causes an update in the path information table but this change does not affect the list of disjoint paths, because the new path $x-y-z-B$ is more costly than the existing disjoint paths. Therefore node x does not transmit a PathInfo message in this case.

The path information collection stage of the sample run of the DPV algorithm continues with the INIT message transmitted by the supernode C in Figure 4.10. Supernode C has two neighbors namely x and z. The INIT message received by node x does not make a change on the list of disjoint paths as the link $x - C$ has a high cost (280). On the other hand, the INIT message received by the node z does make a difference. The new path $z - C$ has a cost of 200 which is the second cheap path in the path information table of node z. Calculation of disjoint paths for node z results the paths $z - B$ and $z - C$. As a consequence of this update node z will again need to transmit a PathInfo message.

In Figure 4.11 we see the final PathInfo transmission made by node z. As seen from the path information tables there is one new path for node x and one for node y. However none of



Path information tables maintained by sensor nodes

A	x-A	90
B	x-B	190
B	x-z-y-B	250
B	x-z-B	250

x
Disjoint Paths

x-A	90
x-B	190
Max Cost	190

B	y-z-B	210
B	y-B	230
A	y-z-x-A	250
B	y-z-x-B	250

y
Disjoint Paths

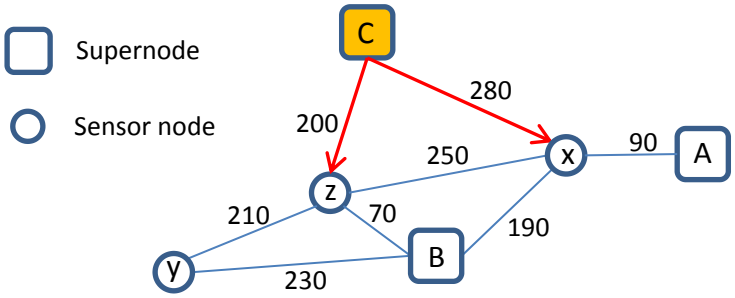
y-z-B	210
y-B	230
Max Cost	230

B	z-B	70
B	z-y-B	230
A	z-x-A	250
B	z-x-B	250

z
Disjoint Paths

z-B	70
z-y-B	230
Max Cost	230

Figure 4.9: Path information tables after node z transmits a $PathInfo$ message to node x.



Path information tables maintained by sensor nodes

A	x-A	90
B	x-B	190
B	x-z-y-B	250
B	x-z-B	250
C	x-C	280

B	y-z-B	210
B	y-B	230
A	y-z-x-A	250
B	y-z-x-B	250

B	z-B	70
C	z-C	200
B	z-y-B	230
A	z-x-A	250
B	z-x-B	250

x
Disjoint Paths

x-A	90
x-B	190
Max Cost	190

y
Disjoint Paths

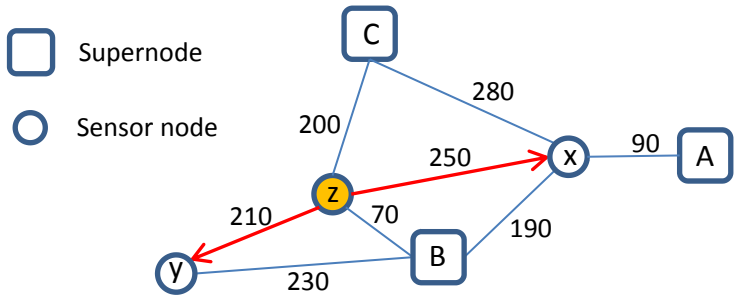
y-z-B	210
y-B	230
Max Cost	230

z
Disjoint Paths

z-B	70
z-C	200
Max Cost	200

Figure 4.10: Path information tables after supernode *C* transmits an *INIT* message.

these paths makes a difference on the list of disjoint paths as demonstrated in Figure 4.11. So there will not be any PathInfo transmissions regarding to this step. Since all 3 supernodes have transmitted their INIT messages and all corresponding PathInfo transmission are over, the path information collection stage of the DPV algorithm ends. The disjoint paths shown in Figure 4.11 are the final disjoint paths for the nodes x, y and z.



Path information tables maintained by sensor nodes

A	x-A	90
B	x-B	190
B	x-z-y-B	250
B	x-z-B	250
C	x-z-C	250

B	y-z-B	210
C	y-z-C	210
B	y-B	230
A	y-z-x-A	250
B	y-z-x-B	250

B	z-B	70
C	z-C	200
B	z-y-B	230
A	z-x-A	250
B	z-x-B	250

x Disjoint Paths

x-A	90
x-B	190
Max Cost	190

y Disjoint Paths

y-z-B	210
y-B	230
Max Cost	230

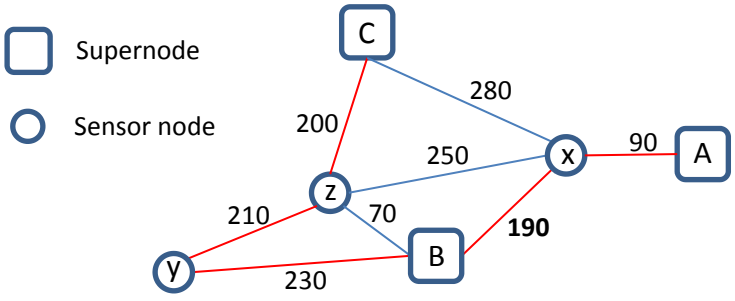
z Disjoint Paths

z-B	70
z-C	200
Max Cost	200

Figure 4.11: Path information tables after node z transmits $PathInfo$ message to nodes x and y.

Having found the disjoint paths, each sensor node now creates its required neighbor list. These are the neighbors where a disjoint path starts from. A node v holds path information to a supernode A, as a sequence of node ID's that constitute the path as seen in Disjoint Paths tables in Figure 4.12. So the first ID in the sequence represents the current node and the next ID represents the first node u of the path to the supernode A. In other words, when node v wants to send a message to the supernode A, the message will be forwarded through the node u. If such a path is one of k best disjoint paths of node v, then node u is labeled as a required neighbor of node v. Since we aim to come up with a bidirectional network, node v is also labeled as required neighbor for node u as well. For instance, node y has two disjoint paths namely $y - z - B$ and $y - B$. For the disjoint path $y - z - B$ node z is the starting point of the path because it is the next node after y. Therefore node z is a required neighbor for node y. In the same manner, supernode B is also a required neighbor for node y. Required neighbors for all sensor nodes are shown in in Figure 4.12. A link with a required neighbor is shown

in red. Procedure for finding the required nodes which is run by each sensor node is given in Algorithm 4.5.4.



x

Disjoint Paths

x-A	90
x-B	190
Max Cost	190

Required Neighbors

{A,B}

y

Disjoint Paths

y-z-B	210
y-B	230
Max Cost	230

Required Neighbors

{z,B}

z

Disjoint Paths

z-B	70
z-C	200
Max Cost	200

Required Neighbors

{B,C}

Figure 4.12: Disjoint paths and required neighbors for all sensor nodes after the path information collection stage.

To guarantee that all nodes in a selected disjoint path are labeled as required neighbors, we need to notify all the nodes on that path. To achieve this, each node sends a $NOTIFY$ message for each of its selected disjoint paths. A $NOTIFY$ message is forwarded along the disjoint path for which it was created. Each neighboring node in the disjoint path marks each other as required neighbors. This stage ensures that any node on a selected disjoint path will be marked as a required neighbor of its neighbors that are also on the same disjoint path. If any two neighbor nodes do not mark each other as required neighbors, it means that the link between these two nodes is not necessary and can be removed. Procedure 4.5.5 shows the steps taken by each sensor node upon receiving a $NOTIFY$ message.

For a node which do not have the path information of at least k vertex disjoint paths to supernodes, all neighbors are defined to be required because in that case it is not known which neighbors can be given away. Such a node cannot decide which neighbors are required and which are not because it has not enough information locally. In that case all neighbors are kept because all of them can potentially be part of a disjoint path connecting some sensor nodes to supernodes. This approach ensures the k-vertex supernode connectivity of the resulting network topology in case of insufficient amount of local information.

Procedure 4.5.4 Finding Required Neighbors

Input: D and k
Output: R
1: $R \leftarrow \emptyset$;
2: $S \leftarrow \emptyset$;
3: **if** $|D| >= k$ **then**
4: Sort(D);
5: $S \leftarrow \{p_i \in D \mid i <= k\}$;
6: **for all** $p \in S$ **do**
7: $R \leftarrow R \cup p$.First;
8: **end for**
9: **end if**
10: **for all** $p \in S$ **do**
11: Transmit Notify(p);
12: **end for**

Procedure 4.5.5 Updating Required Neighbors By *Notify* Messages

Input: R, B and Φ: Incoming Notify message
Output: R: Updated set of required neighbors
1: **for all** received Notify message Φ **do**
2: **for all** Node $n \in \Phi$.Path **do**
3: **if** $n \in B$ **then**
4: $R \leftarrow R \cup n$;
5: **end if**
6: **end for**
7: **end for**

Having determined the final set of neighbors, each node adjusts its transmission power as to reach its furthest neighbor according to the new topology which came out after the removal of neighbors that are not required. This is the last stage of the DPV algorithm.

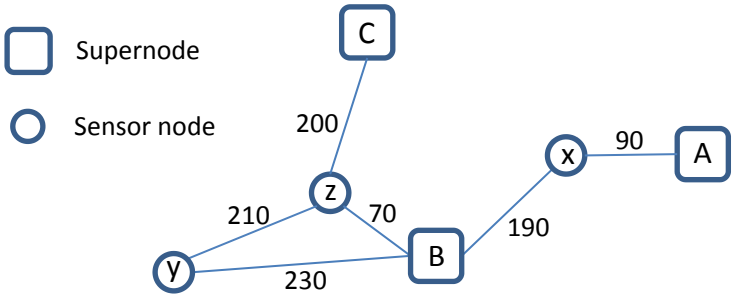
The resulting topology generated by applying the last stage of the DPV algorithm to the sample network topology is shown in Figure 4.13. The link between nodes z and x is removed. Also the link between the supernode C and node x is removed. All the sensor nodes in the network are connected to at least one supernode with 2-vertex disjoint paths.

Theorem 1 (Correctness): If G is k-vertex supernode connected then the final topology generated by the DPV algorithm is a k-vertex supernode connected topology.

Proof: Let us assume that the current graph is k-vertex supernode connected and that an edge (u, v) is removed. Consider any sensor node n. It is k-vertex supernode connected before the removal of edge (u, v), which can only happen in the DPV algorithm when neither u nor v mark each other as required. Note that both u and v have k-vertex disjoint paths to at least one supernode that does not contain node v and u, respectively. After the removal of (u, v) it is thus guaranteed that both u and v will remain k-vertex connected to at least one supernode; otherwise the DPV algorithm would not remove the edge (u, v). If removal of an edge (u, v) does not break the k-vertex supernode connectivity of node u and v, it is guaranteed that the resulting topology will be k-vertex supernode connected as proved in [51]. Therefore, we conclude that the DPV algorithm results in a topology where all sensor nodes are k-vertex supernode connected.

Theorem 2 (Bidirectionality): If the links in the initial network topology is bidirectional then the final topology generated by the DPV algorithm is also bidirectional.

Proof: In the initial network topology, all the links are bidirectional. The algorithm DPV removes a link between a pair of nodes only when both nodes agree that they are redundant neighbors to each other. So before removing an edge (u, v) the DPV algorithm makes sure



Nodes x, y and z have at least 2 node-disjoint paths
to the set of supernodes

x

Disjoint Paths

x-A	90
x-B	190
Max Cost	190

y

Disjoint Paths

y-z-B	210
y-B	230
Max Cost	230

z

Disjoint Paths

z-B	70
z-C	200
Max Cost	200

Figure 4.13: The final topology resulted after the application of the DPV algorithm.

that u is in the redundant list of v and v is in the redundant neighbor list of u. This means an edge can only be removed in agreement of both nodes. If this condition is not met none of the nodes remove its neighbor. So it is not possible to have any unidirectional links in the topology generated by the DPV algorithm. Therefore we conclude that the DPV algorithm always generates bidirectional topologies.

4.6 Time Complexity Analysis of the DPV Algorithm

The execution of the DPV algorithm is initiated by the *INIT* messages transmitted by supernodes. *INIT* messages can only be transmitted by supernodes and they contain the ID of the transmitting supernode. Any sensor node that receives an *INIT* message simply calculates the path cost between the transmitting supernode and the sensor node itself and adds this path into its local path information table. Since an update occurred in the local path information table of the receiving sensor node, this node now creates an updated $PathInfo$ message and transmits it to its 1-hop neighbors. These steps takes constant time (1).

Sensor nodes receiving a $PathInfo$ message calculate the union of the local path information and the received paths in the incoming message. The running time complexity of this step depends on the number of paths in the local path information table of the receiving node and in the incoming $PathInfo$ message. In the DPV the maximum number of paths that can be

stored in a sensor node's path information table is set to a constant value (p) because sensor nodes have limited memory and limited processing power. Calculating the union of the two path information tables takes $O(p)$ time. Paths in the union are sorted in ascending order; the first p of the paths are kept and the others eliminated. Sorting takes $O(plogp)$ time and getting the first p paths takes $O(p)$ time (2).

Having formed the union of the path information tables, we now find the set of disjoint paths of size k with the minimum cost. We achieve this by enumerating all subsets of paths of size k and find the set with the minimum cost. Enumerating all these subsets takes $O(p^k)$ time and (p^k) subsets are formed. For each subset we perform a disjointness test and find the disjoint set with the minimum cost. This step takes $O(k\lambda p^k)$ time, where λ is the maximum length for a path in terms of number of hops. This parameter is also set to be a constant in the DPV and is a result of setting a TTL value for a $PathInfo$ message forwarded in the network. In our experiments we set λ to 7 but it can be thought as a small constant compared to number of sensor nodes in the network. Therefore, we conclude that upon receiving a $PathInfo$ message a sensor node requires $O(k\lambda p^k)$ time, where k, λ and p are constants (3). The step by step running time analysis of the procedure for finding set of disjoint paths of size k with minimum total cost is given in Figure 4.14.

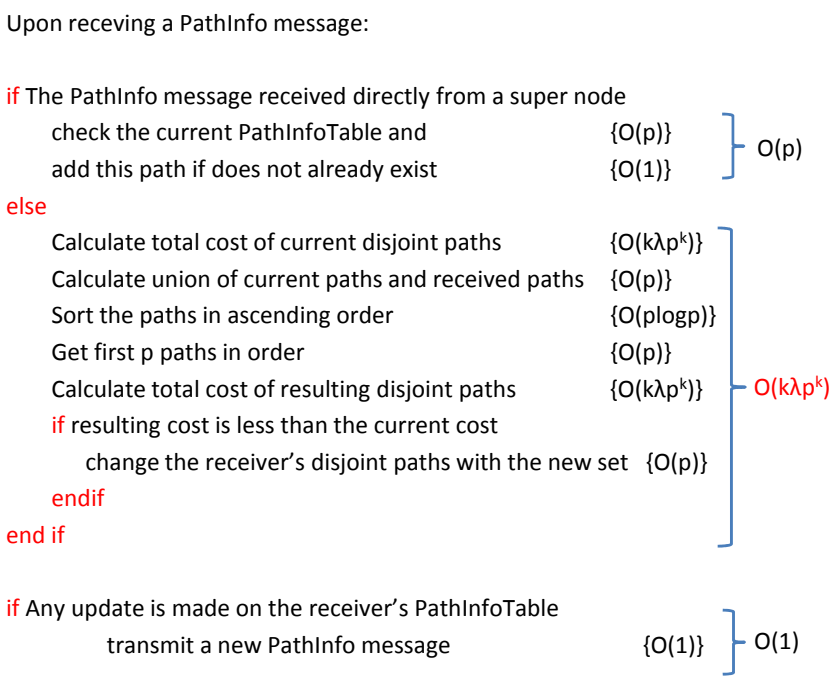


Figure 4.14: Time Complexity Analysis of Finding Disjoint Paths.

The calculated cost for the union is compared with the original disjoint path cost. If the new cost is less than the previous cost, the node updates its local path information table and transmits a new $PathInfo$ message that contains the updated path information. Calculating the cost for the original path information table is not necessary because it was calculated in the last update. If a better set of disjoint paths is found then the path information table would need to be updated. This step involves clearing the table and inserting the new paths into it, which takes $O(p)$ time (4). The step by step running time analysis of the path information collection stage of the DPV algorithm is given in Figure 4.15.

```
Create an empty list DisjointPaths to hold disjoint paths                {O(1)}
if Number of paths in PathInfoTable is greater than k
    Find all subsets of size k for PathInfoTable                         {O(p^k)}
    Find the disjoint subset with minimum total cost                     {O(kλp^k)}
    Fill the DisjointPaths with the paths in min. cost subset            {O(k)}
else
    for each PathInfoRow in PathInfoTable do                             { max. k times}
        IsDisjoint = TRUE
        for each Path in DisjointPaths do                                { max. k-1 times}
            if DP contains any node in PathInfoRow.Path                  {O(λ)}
                IsDisjoint = FALSE
                Break the loop
            end if
        end for
        if IsDisjoint == TRUE
            Add path PathInfoRow.Path to DisjointPaths                   {O(1)}
        end if
    end for
end if
```

O(kλp^k)

O(k²λ)

p : Maximum number of paths that can be stored by a node
k : Desired number of disjoint paths to super nodes per node
λ : Maximum number of hops for a path

Figure 4.15: Time Complexity Analysis of Path Information Collection Stage.

After the path information collection stage, required neighbors are found. In this step each sensor node traverses its disjoint path set and marks the first nodes of each path as required. In addition, it sends a notification message along each disjoint path to notify the nodes that are on that path. Since there can be at most k such paths, this step takes $O(k)$ time (5).

Having found the required neighbors, each sensor adjusts its transmission range accordingly. This step can be completed in $O(k)$ time (6).

Above, we analyzed the running time complexity of the main steps of the DPV algorithm. The dominating step is (3), which finds the minimum-cost disjoint paths. Because this step

takes $O(k\lambda p^k)$ time, we conclude that any sensor that receives a $PathInfo$ message requires $O(k\lambda p^k)$ processing time at the worst case.

To determine the total running time for the algorithm to reach the final topology, we must determine the maximum number of $PathInfo$ messages that a sensor node can receive during the execution of the DPV algorithm. As explained in Section 4.5, the upper bound on the number of $PathInfo$ messages that a node can transmit is $O(n\Delta)$, where n is the number of nodes in the network and Δ is the maximum degree of a sensor node. Thus, a node can receive at most $O(n\Delta^2)$ $PathInfo$ messages. Hence, total asymptotic running time of the DPV algorithm is $O(n\Delta^2 k\lambda p^k)$ per node. Since the parameters k, λ and p are constant, DPV's running time complexity is $O(n\Delta^2)$.

In [51], the time complexity of the DATC algorithm is given as $O(\Delta^5)$. So, in dense graphs where Δ is high compared to n, DATC will experience much higher computation overhead than the DPV algorithm. In the worst case, where there is an edge between every node, complexity of DATC will be $O(n^5)$ whereas our algorithm will be $O(n^3)$.

4.7 Message Complexity Analysis of the DPV Algorithm

During the execution of the DPV algorithm, messages are transmitted in four situations. The first one occurs in the initiation phase, where supernodes send $INIT$ messages. Each supernode sends one $INIT$ message, therefore m messages are transmitted in this phase, where m is the number of supernodes. However, we do not consider these messages because they are sent by supernodes, which have no power restrictions in our scenario. In addition, since $m << n$, where n is the number of sensor nodes, the asymptotic message complexity is not affected, as we show below.

Table4.3: Time and message complexities of DPV and DATC per node.

Algorithm	Time Complexity	Message Complexity
DPV	$O(n\Delta^2)$	$O(n\Delta)$
DATC	$O(\Delta^5)$	$O(\Delta^h)$

The second case occurs when a path information update is made to the local path information table maintained by each sensor node. As discussed in Section 4.5, the upper bound on the number of $PathInfo$ messages that a node can transmit is $O(n\Delta)$, where n is the number of nodes in the network and Δ is the maximum degree of a sensor node. Then, considering that a node has at most Δ neighbors to receive from, the number of $PathInfo$ messages that a sensor node can receive is bounded by $O(n\Delta^2)$.

The third situation is node notification, which occurs on a selected disjoint path. A $Notify$ message is transmitted for each selected disjoint path, and because a node can select at most k paths, a sensor node initiates k $Notify$ messages at most. Each $Notify$ message causes $\lambda - 1$ message transmissions, where λ is the length of a disjoint path in terms of number of hops. Therefore, at most a sensor node transmits $k(\lambda - 1)$ $Notify$ messages during this step.

From these calculations, we have $O(n\Delta)$ messages and $k(\lambda-1)$ $Notify$ messages per node and the total number of messages that a sensor node transmits in the worst case is thus $O(n\Delta)$ +

$k(\lambda - 1)$. Eliminating low-order terms, we conclude that the asymptotic message complexity of the DPV algorithm is $O(n\Delta)$.

The message complexity of DATC is reported as $O(\Delta)$ per node in [51]. However this is valid for 1-hop neighborhood only. For h-hop neighborhood the complexity will be $O(\Delta^h)$, because each sensor needs to forward the messages that it receives from its h-hop neighborhood. On the other hand, one can note that the worst case for the DPV algorithm, where paths are discovered in the order of decreasing cost, is unlikely, since it is more probable to receive messages first from shorter paths than the longer ones. Therefore, we expect a lower message complexity in the average case. Our simulation results are in conformity with this analysis. Time and message complexity of the algorithms are summarized in Table 4.3.

CHAPTER 5

EVALUATION

In this chapter we report the results of the experiments performed to evaluate our proposed algorithm. We evaluate our algorithm by comparing it with GATC and DATC. algorithms. In order to achieve this, we implemented DPV, GATC and DATC algorithms by using a custom evaluation framework which provides generating random network topologies, executing the algorithms on the generated topologies, calculating the desired metrics and visualizing the outputs of the experiments.

5.1 Evaluation Approach

The general approach that we follow for evaluating our algorithm is summarized in Figure 5.1.1.

Algorithm 5.1.1 Evaluation Approach.

```
1:  for all Evaluation Scenarios S do
2:    for seed = 1 to n do
3:      Generate random network topology T using seed
4:      for all Topology control algorithms A do
5:        Run experiment with A for S on T
6:        Store the output in a table
7:      end for
8:      Calculate the average values of the stored result
9:      Write the average values into the output file
10:   end for
11: end for
```

This approach ensures that each algorithm is run on the same random topologies, so it prevents any biased results that may occur due to the specific topology of the network generated.

As summarized in Figure 5.1.1, for each scenario we run DPV, GATC and DATC algorithms with different seeds, and for each experiment we output the results, namely total transmission power and total number of links in the resulting topologies generated by each algorithm, also the number of total transmissions and receptions required for execution of the algorithms.

Table5.1: Simulation Parameters.

Deployment Area	$600m \times 600m$
Path Loss Exponent: α	2
Initial Transmission Range of Nodes: R_{max}	$100m$
Number of Sensors: N	[100...500]
Number of Supernodes: M	5% and 10% of N
Degree of Disjoint Connectivity: k	2 and 3
Number of Hops for Neighborhood in DATC: h	1 and 2
Confidence Level	95%

5.2 Experimental Setup

In our experiments, the sensors are randomly deployed in a 600 m × 600 m area. The supernodes are also randomly and uniformly deployed in this area. The path loss exponent for the wireless channel α is chosen to be 2 and the initial maximum transmission range R_{max} of the nodes is set to be 100 m. Since it is required to have a network topology where each sensor is at least k-vertex connected to the set of supernodes, topologies that do not satisfy this condition are discarded. For the parameters k and h we use similar values with [51], which are typical values seen in k-connectivity studies. The experiments are repeated at least hundred times for each parameter set and average values are reported at 95% confidence level with a maximum 5% confidence interval. Confidence intervals for reported values are depicted on the corresponding charts. Our simulation parameters are summarized in Table 5.1.

We assume that each node in the network has a unique ID and no collisions or retransmissions occur during wireless communications. We extend our simulations with packet loss scenarios in Section 5.3.6. In addition, we assume that nodes can predict the length of the links using received signal strength. We do not target a specific sensor node technology in the simulations, but it can be considered as any sensor node platform capable of adjusting transmit power with a maximum attainable range R_{max} of 100 m. As supernodes we consider actor devices with dual radios, one for communicating with sensors and one with other supernodes. The maximum transmission range for supernodes to communicate with sensor nodes is set to be 100 m as well. Transmission range for supernodes to communicate with other super nodes is irrelevant for our purpose, but we assume that range to be sufficiently large to communicate with any other supernode in the network. For sensor nodes, the transmit power needed to talk to a receiver in range is calculated by assuming a simple path loss model and using the simple formula $p_i = r_i^{\alpha}$, as done in [51]. Here, p_i is the transmit power node i is using, r_i is the transmission range of the node, and α is the path loss exponent. Hence, the power values are simply expressed as a function of the transmission range (i.e., distance) for the purpose of comparing the algorithms. Therefore, we do not give power values in a specific energy unit such as Joule. We focus mainly on the following metrics to evaluate our algorithms:

- Total transmission power: This is the sum of final transmission power values of each single node in the resulting topology after the execution of the given topology control algorithm. This metric is directly related to transmission power cost for communication in the resulting network and hence an important performance metric for the evaluation

of the algorithms.

- Maximum transmission power: This is the maximum of final transmission power assigned among all sensor nodes in the resulting topology after the execution of the given topology control algorithm. This metric is an indication of how the power consumption is balanced in the resulting network topology. If an algorithm produces a topology with a high maximum transmission power, that means some nodes will die earlier than the other nodes and hence the network may get disconnected or partitioned in early stages of the network lifetime.
- Total number of links: This is the sum of the number of the links between the sensor nodes in the resulting topology. This metric is related with the total reception power cost of the resulting network topology. Although this metric is not considered in [51], we believe that it is important because number of receptions is also important with respect to power efficiency of the resulting topology.
- Number of Total Transmissions: This is the sum of number of messages transmitted during the execution of the algorithms. This metric depicts the transmission power cost of generating the k-vertex supernode connected topology. In [51], power consumption during the execution of the DATC algorithm is not considered. They only consider the power consumption of the final topology but it is clear that power consumption of the algorithms should also be taken into account, since it directly affects the remaining energy levels of the sensor nodes in the network.
- Number of Total Receptions: This is the total number of received messages by the sensor nodes in the network during the execution of the algorithms. This metric depicts the power used during the reception of the messages while generating the final topology. Although number of total transmissions constitutes the essential part of the total power consumption, number of receptions may have an important impact on the total power consumption. Hence we also monitored this metric in our experiments.

Before going into detailed analysis of the experiment results, we present resulting topologies after the execution of the algorithms DATC, DPV and GATC on a sample initial topology for 2-vertex disjoint connectivity. The initial network topology shown in Figure 5.1 has 200 sensor nodes and 10 supernodes deployed on a 600m x 600m area with an average node degree of 8.75.

Figure 5.2 shows the result of execution of DATC algorithm with h=1. It achieves a 30% improvement on the total transmission power where DATC algorithm with h=2 yields a 65% improvement compared to initial topology. The resulting topology after the execution of DATC algorithm with h=2 is given in Figure 5.3.

The topology generated by our algorithm DPV is illustrated in Figure 5.4. DPV performs better than both of the DATC algorithms and achieves a 73% decrease on the total transmission power according to the initial topology.

Finally centralized GATC algorithm which produces the optimal result achieves 84% improvement on the total transmission power. The topology generated by GATC algorithm is shown in Figure 5.5. Note that our distributed algorithm performs fairly close to the optimum result that is achieved by the centralized GATC algorithm in this case. During the detailed analysis of the experiment results,we will see that this is not just a coincidence and it is actually the case in general.

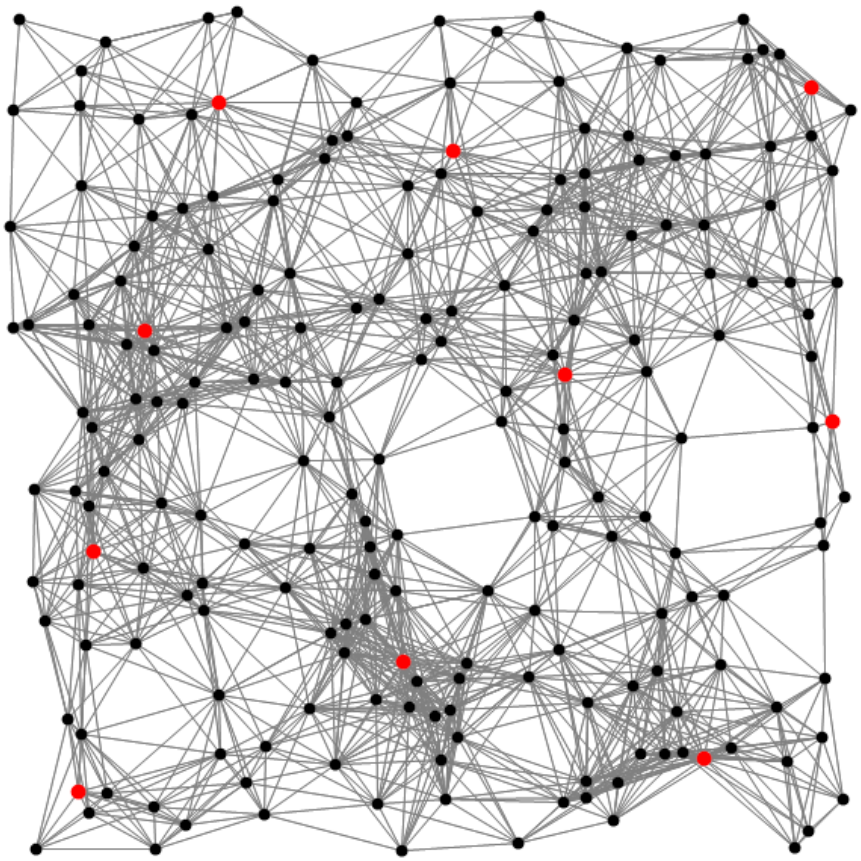


Figure 5.1: Initial sample topology with 200 sensor and 10 supernodes.

5.3 Results

5.3.1 Total Transmission Power

In this section we report experiment results which show the total transmission power of the sensor nodes in the resulting topologies after the execution of GATC, DATC with 1-hop local neighborhood, DATC with 2-hop local neighborhood and our algorithm DPV. Results of GATC algorithm is presented as a reference point, since it assigns the optimal transmission power to sensor nodes. However note that GATC is a centralized algorithm which requires global knowledge of network topology.

In Figure 5.6(a) and Figure 5.6(b), we present the resulting total transmission powers of the algorithms with k=2. In Figure 5.6(a) we see that number of sensor nodes goes from 100 to 500, where number of sensor nodes is set to 5% of number of sensor nodes. We see that total transmission power of the topologies that are generated by DPV is much lower than that of DATC(h=1) and DATC(h=2). This shows that the link elimination approach utilized by DATC has difficulties in discovering alternative disjoint paths to a long direct edge using local data gathered from 1 or 2-hop neighborhood, as we discussed in Section 3. Since the search scope is limited, long, hence, costly edges can not be substituted with cheaper disjoint paths, resulting long transmission ranges and thus high total transmission power. Another important observation related to limited search scope is that DATC(h=1) performs significantly worse than DATC(h=2) which has a broader search scope. However, as will be seen in later

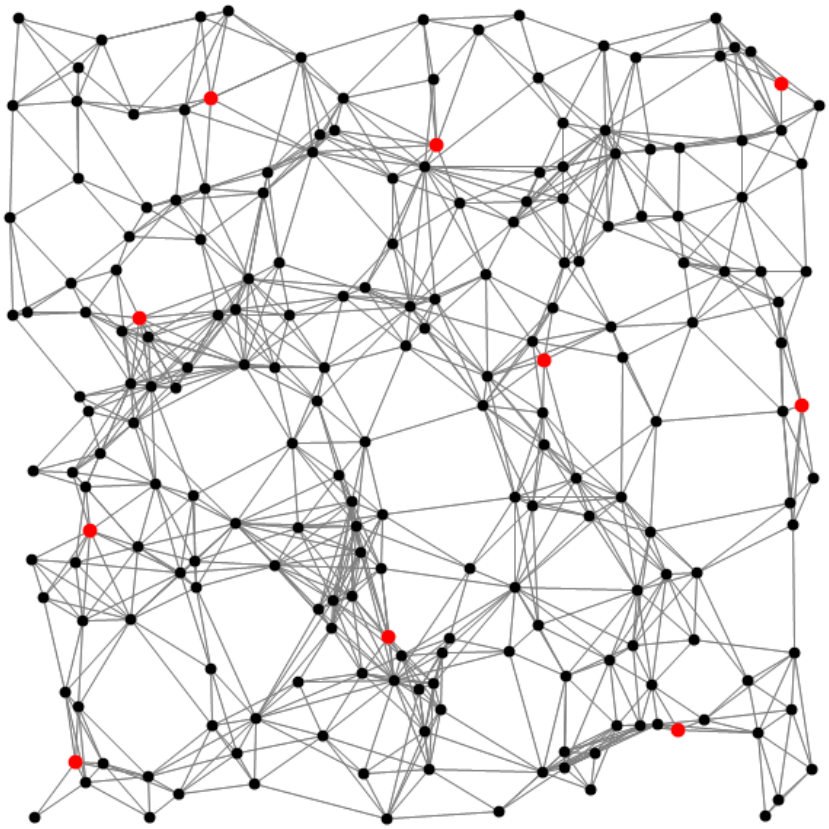


Figure 5.2: Topology generated by DATC algorithm with h = 1.

figures, DATC(h=2) requires significantly more control message transmissions compared to DATC(h=1) for maintaining 2-hop neighborhood.

Figure 5.6(b) shows the results of our experiments where the number of supernodes is 10% of number of sensor nodes. In Figure 5.6(b), we see almost the same picture as in Figure 5.6(a), except that the total power values are little lower because, due to the high number of supernodes in the network, it is more likely that cheaper paths will be found. Our DPV algorithm performs parallel to GATC and the difference between DPV and DATC(h=2) becomes clearer as the network gets denser.

Figure 5.6(b) shows the results of our experiments where the number of supernodes is 10% of number of sensor nodes. In Figure 5.6(b), we see almost the same picture as in Figure 5.6(a), except that the total power values are little lower because, due to the high number of supernodes in the network, it is more likely that cheaper paths will be found. Our DPV algorithm performs parallel to GATC and the difference between DPV and DATC(h=2) becomes clearer as the network gets denser.

Figures 5.7(a) and 5.7(b) depict the total transmission power of the topologies generated by the algorithms for $k = 3$. We see that the total transmission power of the topologies generated by DPV is significantly lower than that of DATC(h=1) and DATC(h=2). For $k = 3$, it is harder to substitute a high cost edge with a low cost one because three disjoint paths are needed to be able to do that. DATC needs to find these paths in its reachable neighborhood where our algorithm can directly search for paths using information in the path vectors. Therefore, our algorithm can achieve lower total transmission power compared to DATC. We also observe

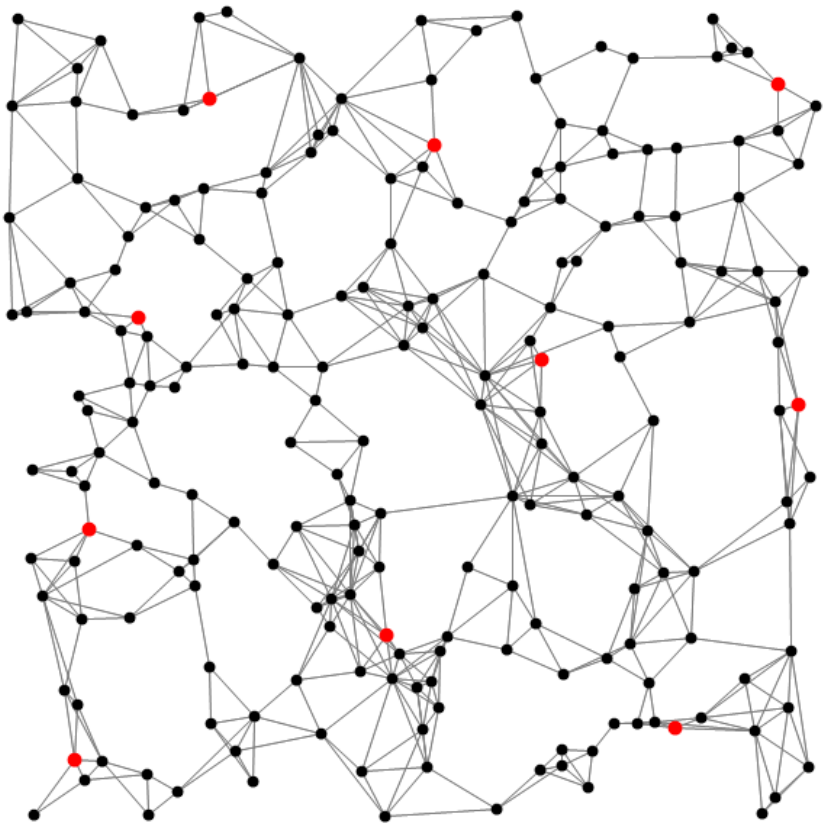


Figure 5.3: Topology generated by DATC algorithm with h = 2.

that satisfying 3-vertex disjoint supernode connectivity requires more transmission power than satisfying 2-vertex disjoint supernode connectivity in general, as expected.

5.3.2 Maximum transmission power

In this section, we report our experiment results on maximum transmission power among all sensors in the resulting topologies generated by the DATC, DPV and GATC algorithms. Maximum transmission power is an important metric for the resulting topologies because it is an indication of the balance of energy consumption among all sensors. Even if total transmission power is low, the resulting topology may become disconnected or even partitioned if the maximum transmission power is high because sensor nodes with a high transmission range will use more energy than other nodes and deplete their batteries sooner.

Figures 5.8(a) and 5.8(b) show the maximum transmission powers of the topologies generated by the algorithms for $k = 2$, where the number of supernodes is 5% and 10% of the sensor nodes, respectively. We observe that GATC produces topologies with the lowest maximum transmission power among all algorithms. This is due to the fact that GATC is a centralized algorithm and has global network topology information. Thus it directly knows where the longest edges reside in the network and eliminates edges in the order of decreasing cost when possible. The DPV and DATC algorithms, on the other hand, are localized algorithms and have a partial knowledge of the network topology. For that reason, they generate topologies with higher maximum transmission power compared to GATC. Our proposed DPV algorithm

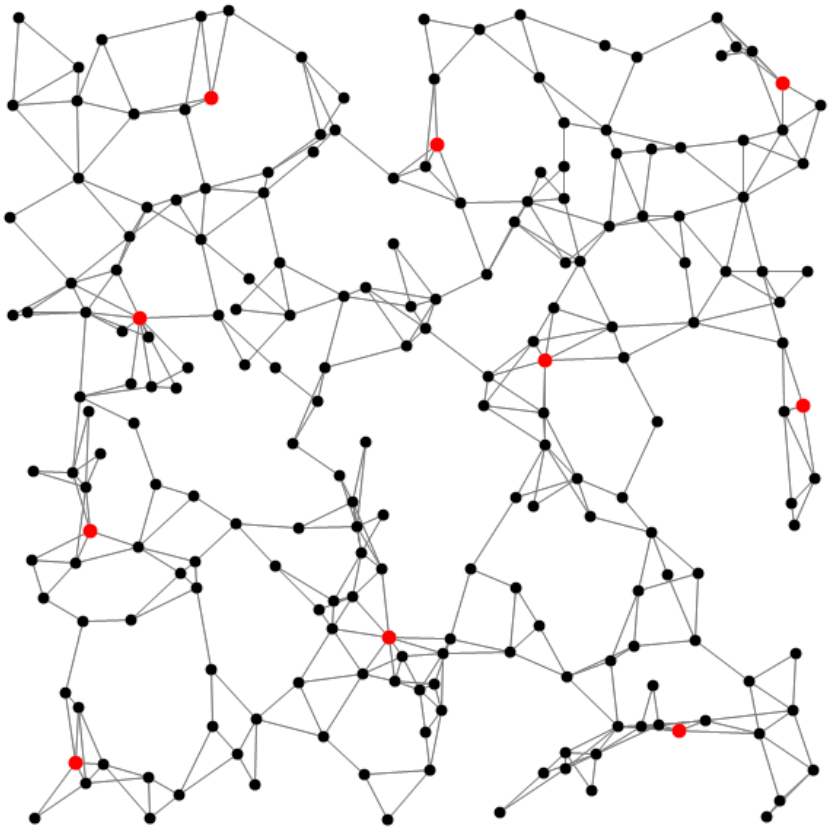


Figure 5.4: Topology generated by our algorithm DPV.

has the next best performance, which is significantly better than the DATC algorithms. This result is again related to the available information for discovering disjoint paths. DATC can obtain limited information compared to DPV, therefore it cannot discover as many disjoint path as our DPV algorithm. This results higher maximum transmission ranges for DATC. DATC(h=1) seems to make almost no improvement on maximum transmission power and DATC(h=2) performs only slightly better.

For $k = 3$, Figures 5.9(a) and 5.9(b) show the comparison of maximum transmission powers for the resulting topologies. The results are similar to when $k = 2$ except that maximum transmission power values are greater. This is an expected result because keeping the network 3-connected requires keeping more neighbors connected, which leads to higher transmission ranges for the sensor nodes.

5.3.3 Total number of links

In this section we present the experiment results regarding the total number links after the execution of each topology control algorithm being compared. Smaller number of links means lower number of receptions due to a transmission. Beside the transmission, message reception also consumes energy so it is important to decrease the number of receptions and hence the number of total links.

Figures 5.10(a) and Figure 5.10(b), show the resulting number of links after the execution of

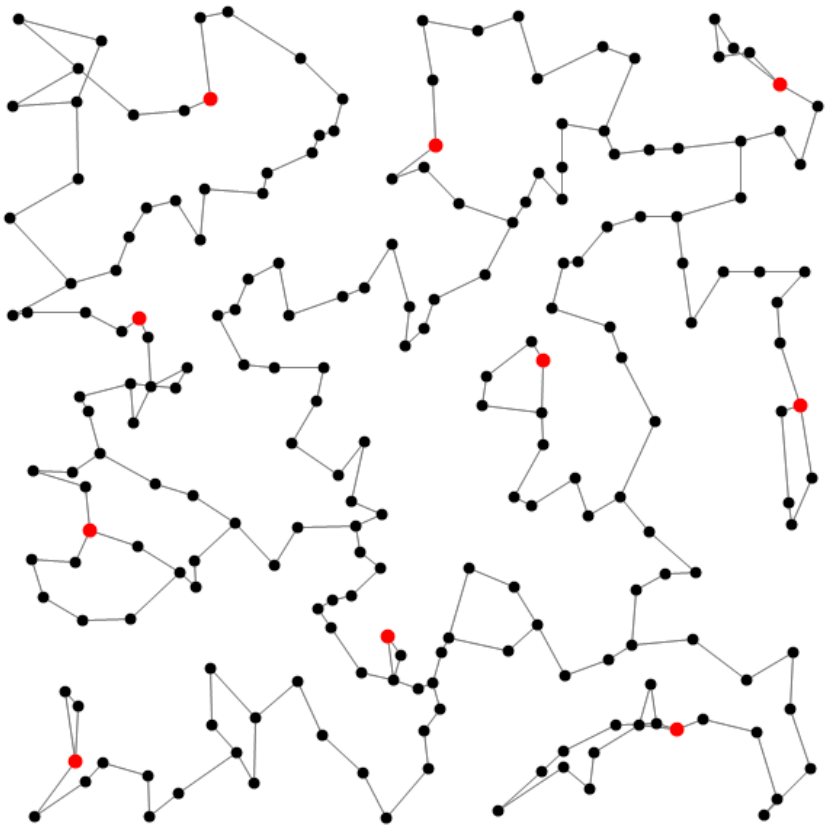


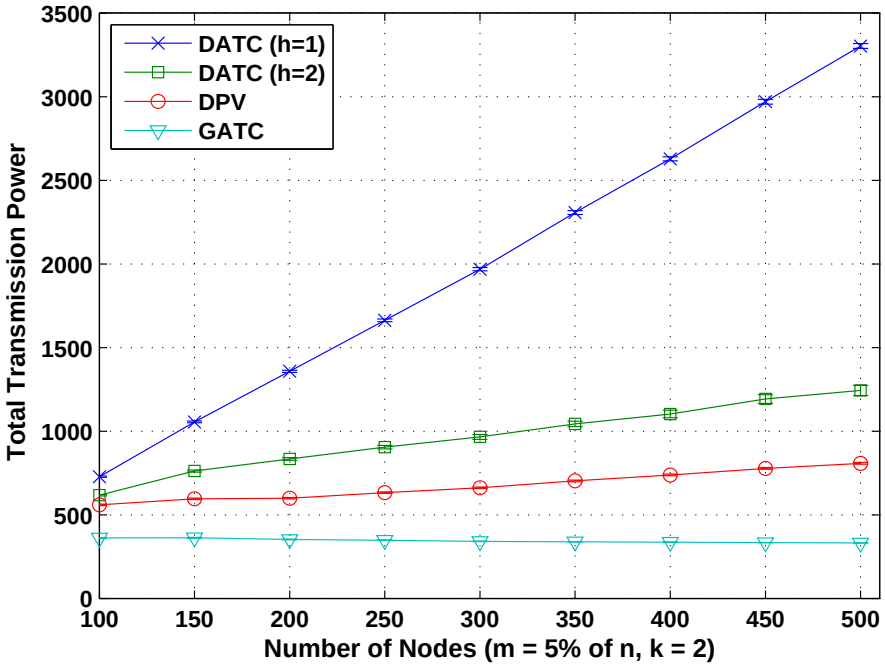
Figure 5.5: Optimum topology generated by GATC algorithm.

each algorithms with k=2.

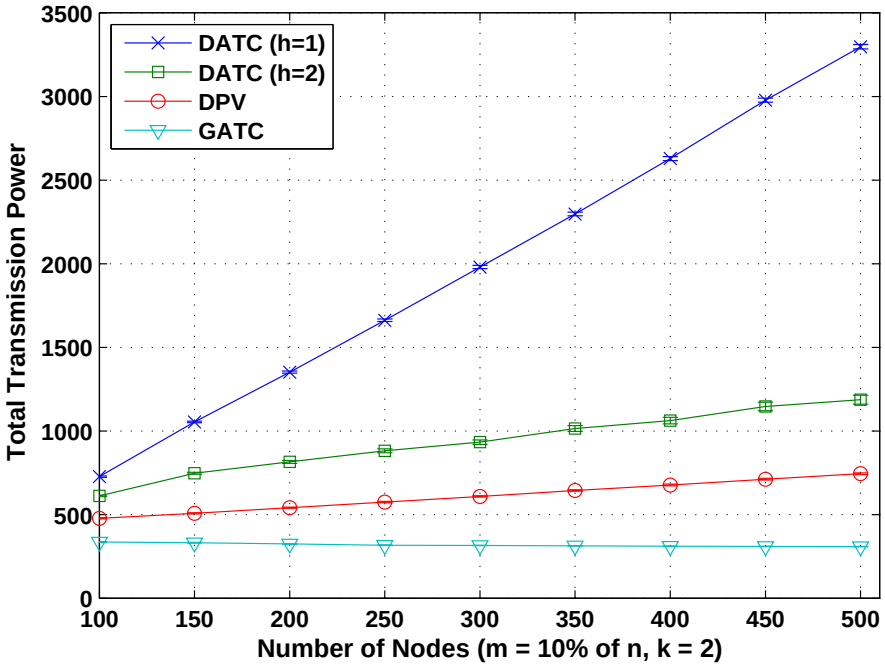
In Figure 5.10(a) number of sensor nodes goes from 100 to 500, where number of sensor nodes is set to 5% of number of sensor nodes. We see that total number of links in the topologies that are generated by DATC(h=1) are increasing very fast while the network size is increasing. DPV results less number of links than both of the DATC algorithms. Number of links for both DPV and GATC increase slowly while the number of nodes is increasing. GATC performs slightly better than DPV, and DPV performs slightly better than DATC(h=2). So we can say that topologies generated by DATC algorithm will consume more power for reception than that of DPV. Another observation is that DATC(h=2) produces less number of links than DATC(h=1) as expected.

In Figure 5.10(b) when supernodes is 10%, we see almost the same picture as in Figure 5.10(a) in terms of relative behaviors of the algorithms.

Figures 5.10(a) and Figure 5.10(b), show the resulting number of links after the execution of each algorithm with k=3. We observe the same pattern with the case where k=2, but here we see that we need more links to keep the sensors 3-vertex disjoint path connected to set of supernodes. We can infer this by comparing the behavior of DPV in cases where k=2 and k=3. When k=2, and number of sensor nodes is 500, approximately 1000 links are enough to maintain the 2-vertex connectivity. On the other hand when k=3, we need approximately 1400 links to keep the network 3-vertex connected to supernodes. Same behavior is also observed for the other algorithms.

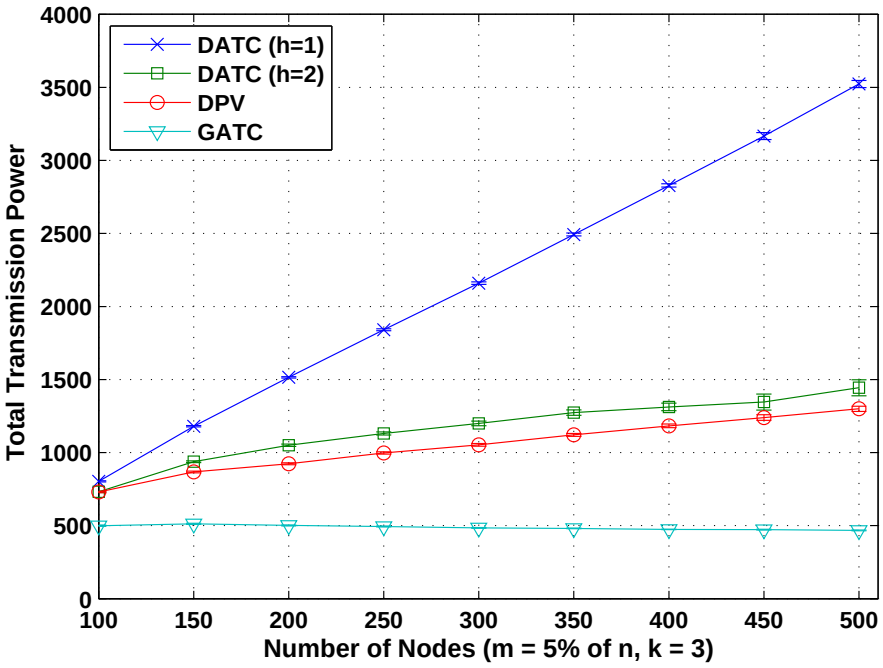


(a)

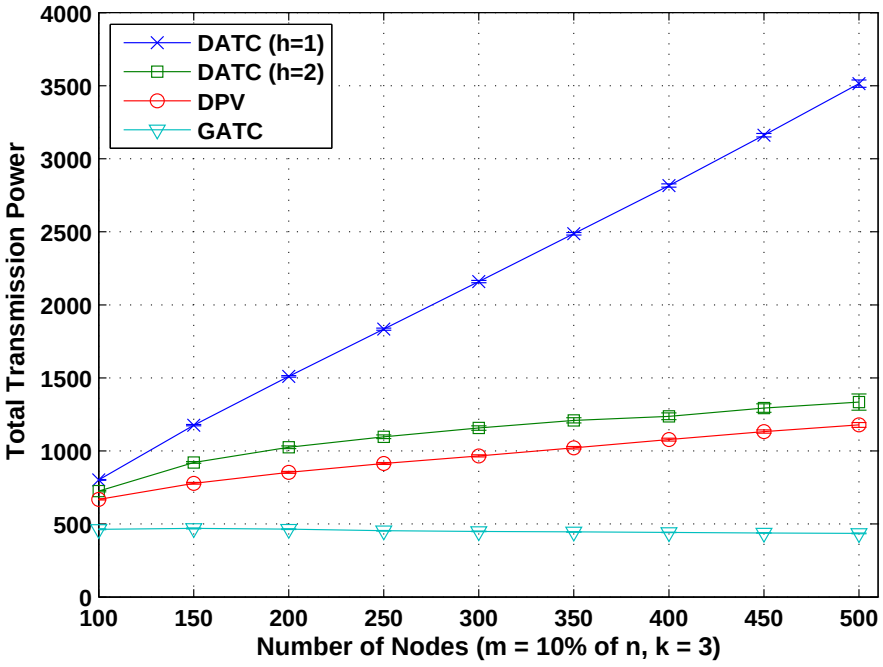


(b)

Figure 5.6: Total Transmission Power for k = 2.

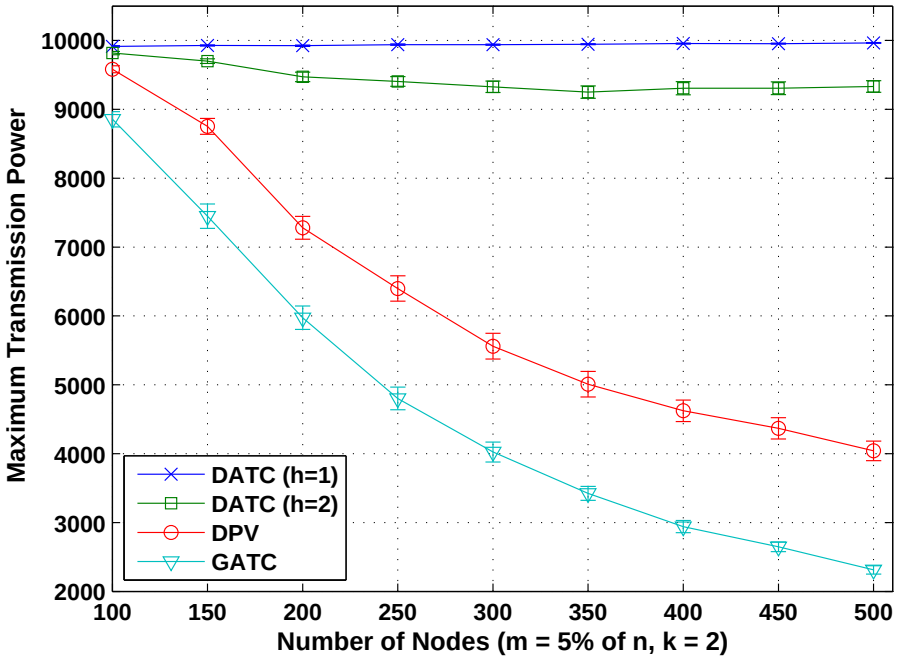


(a)

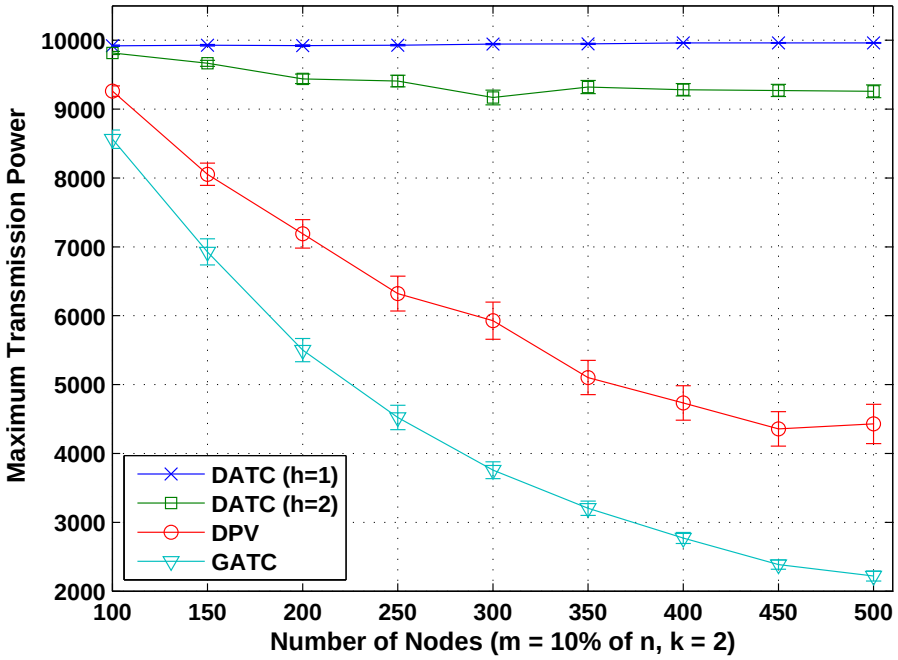


(b)

Figure 5.7: Total Transmission Power for k = 3.

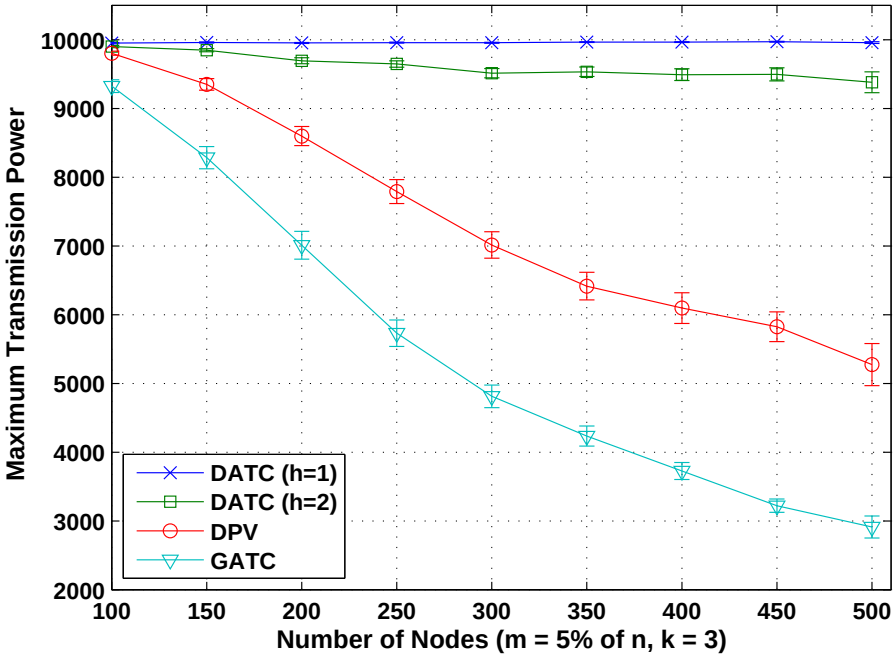


(a)

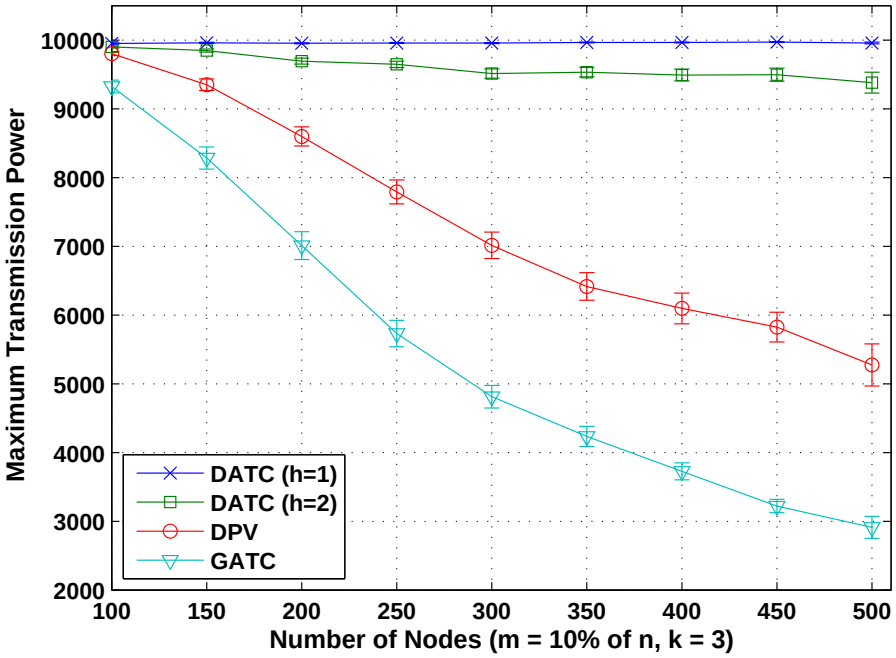


(b)

Figure 5.8: Maximum Transmission Power for k = 2.

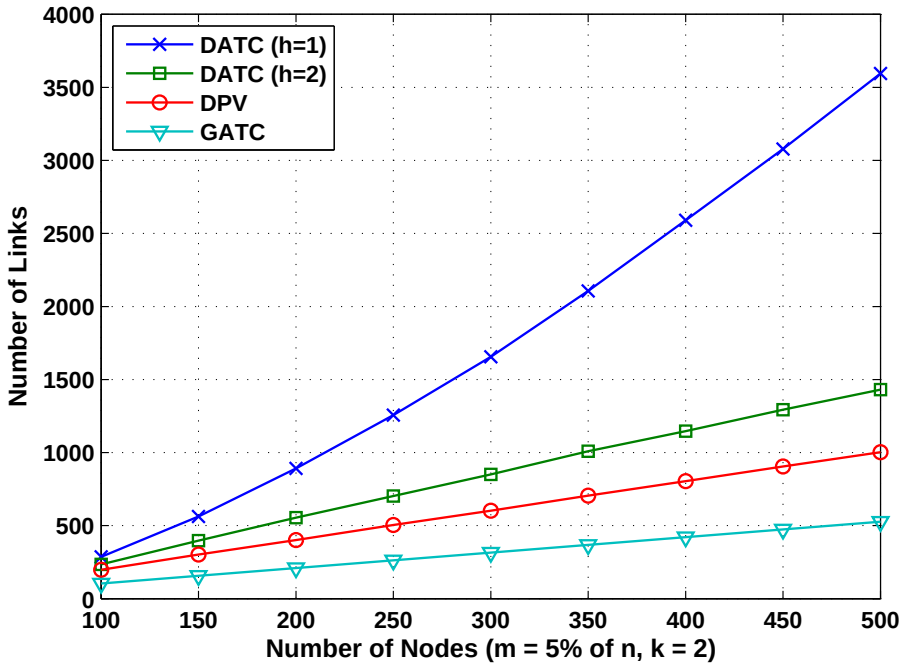


(a)

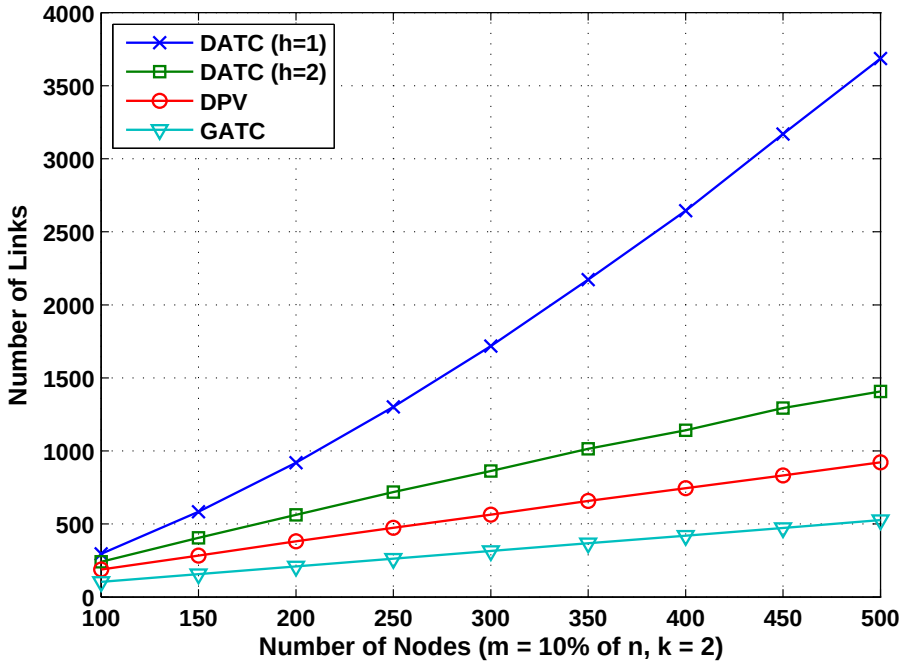


(b)

Figure 5.9: Maximum Transmission Power for k = 3.

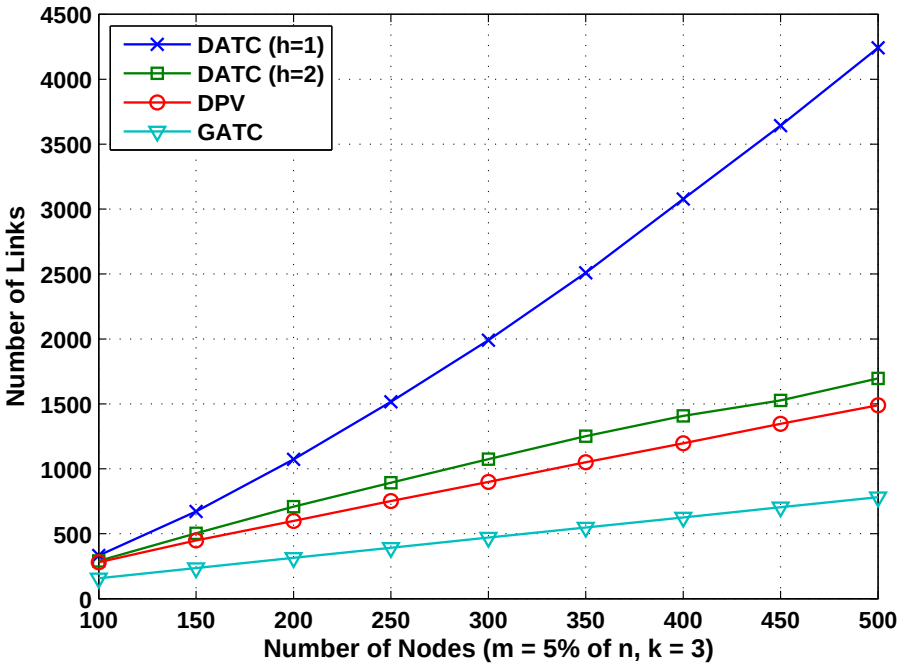


(a)

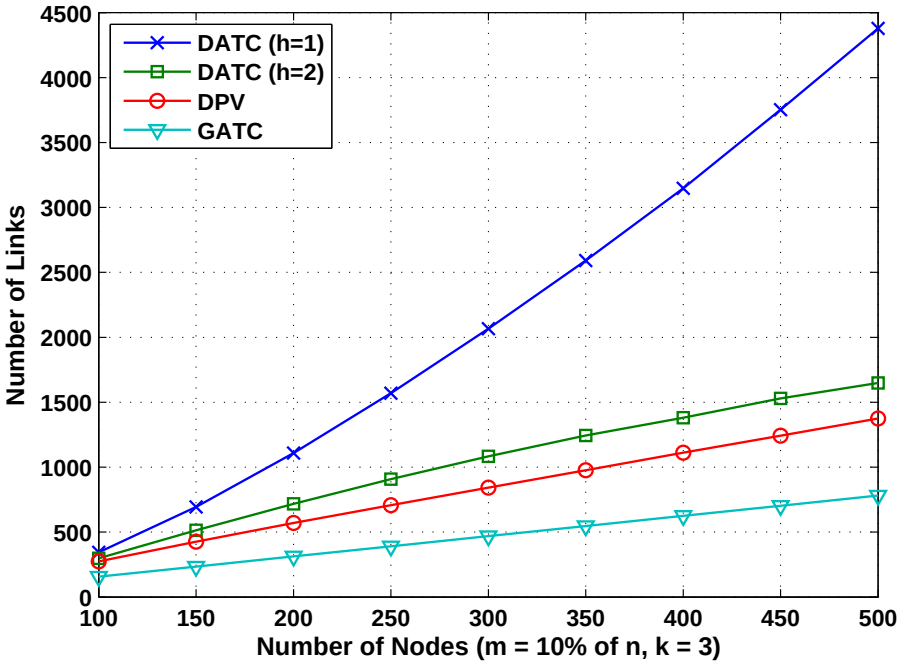


(b)

Figure 5.10: Total Number of Links for k = 2.



(a)



(b)

Figure 5.11: Total Number of Links for k = 3.

5.3.4 Total Number of Control Message Transmissions

In this section we give and discuss our experiment results regarding the total number of control message transmissions during the execution of the DPV and DATC algorithms. GATC does not require control message transmissions while computing the topology, since the computation is done centrally. Control message transmission metric is important because it is necessary not only to consider power consumption in the resulting topologies but also to consider the power consumption required to generate those topologies, which can be viewed as a fixed cost of obtaining the final topologies. If this cost is high, then the power efficiency of the resulting topology might become meaningless.

In Figures 5.12(a) and 5.12(b), we present the total number of required control message transmissions to execute the algorithms with $k = 2$. The most important observation here is that DATC(h=2) requires an asymptotically larger number of transmissions than DATC(h=1) or DPV, because in DATC(h=2) each node needs to notify all nodes in its 2-hop neighborhood. While DATC(h=1) and DPV require only one message transmission for an update, DATC(h=2) requires Δ transmissions, where Δ is the degree of a given sensor node. This situation shows that DATC(h=2) may not be feasible in practice, because transmitting so many messages during algorithm execution will cause rapid battery drain. Therefore, extending the DATC algorithm for more than 1-hop neighborhood seems not to be scalable due to the exponentially increasing number of update messages.

As another benefit, DPV requires fewer transmissions than DATC(h=1) to obtain a k-vertex disjoint connected network topology. This is expected because DPV transmits an update message only if a set of disjoint paths with a lower cost is discovered whereas DATC sends update messages in all power increments. Due to smaller search scope, DATC is often unable to find disjoint paths and thus the number of power increments tends to be high.

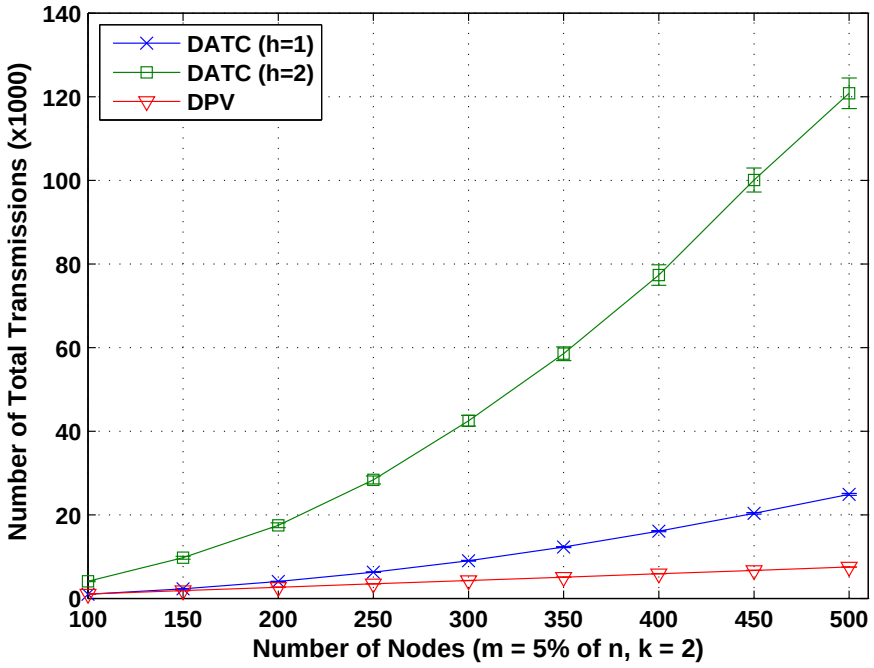
In Figures 5.13(a) and 5.13(b), we present the total number of required control message transmissions to execute the algorithms with $k = 3$. The behaviors of all three algorithms are similar with $k = 2$, except when $k = 3$ all algorithms require slightly more message transmissions.

5.3.5 Total Number of Control Message Receptions

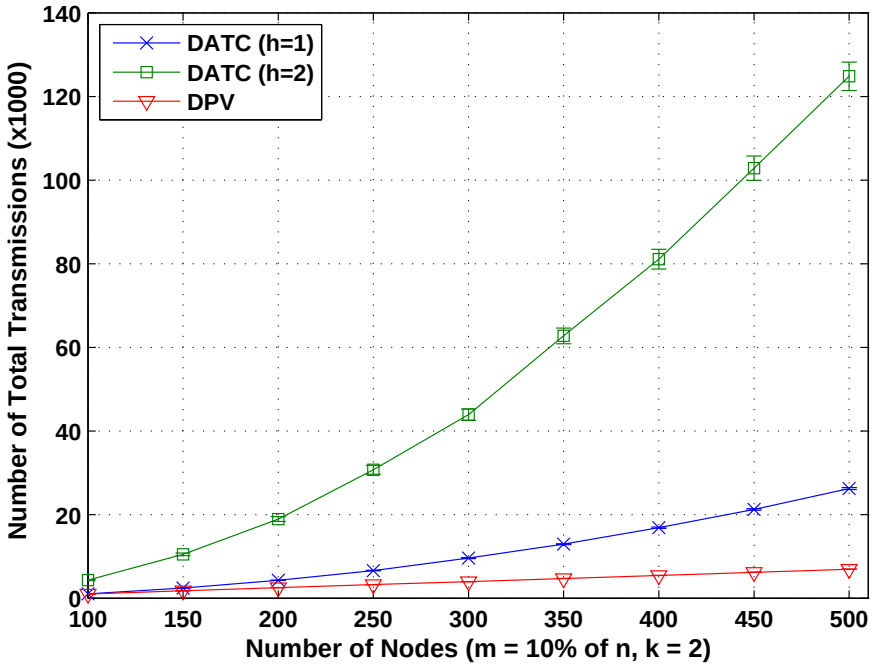
Another factor that has an impact upon power efficiency is the total number of received messages during the execution of the topology control algorithms. This is again a fixed cost as in the case of total transmissions required.

Figures 5.14(a) and 5.14(b) show the total number of received control messages during the execution of the algorithms with $k = 2$, and Figures 5.15(a) and 5.15(b) show the same for $k = 3$. Here again we observe that during the execution of DATC(h=2) many more receptions occur compared to DATC(h=1) and DPV in all cases. This is a direct result of high number of transmissions required by DATC as discussed in Section 5.3.4.

We observe that when the number of supernodes is 5% of the sensor nodes and $k = 2$, DPV requires fewer receptions than DATC(h=1). In all other cases the total number of received messages for DPV and DATC(h=1) is almost the same. For all algorithms, number of receptions tend to increase when k and the number of supernodes increase, but the increase for DATC(h=2) is significantly faster than that of DPV and DATC(h=1). Another important

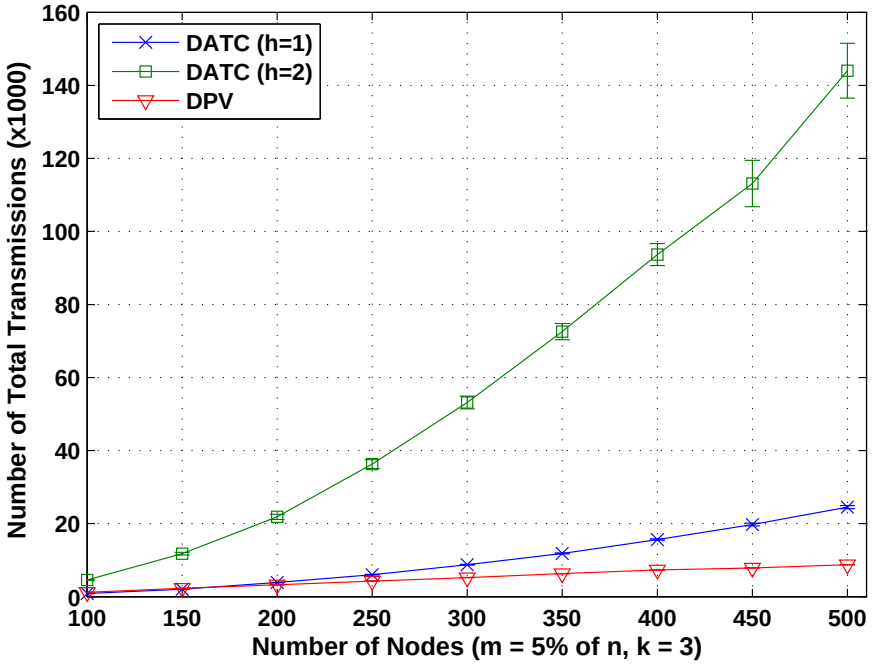


(a)

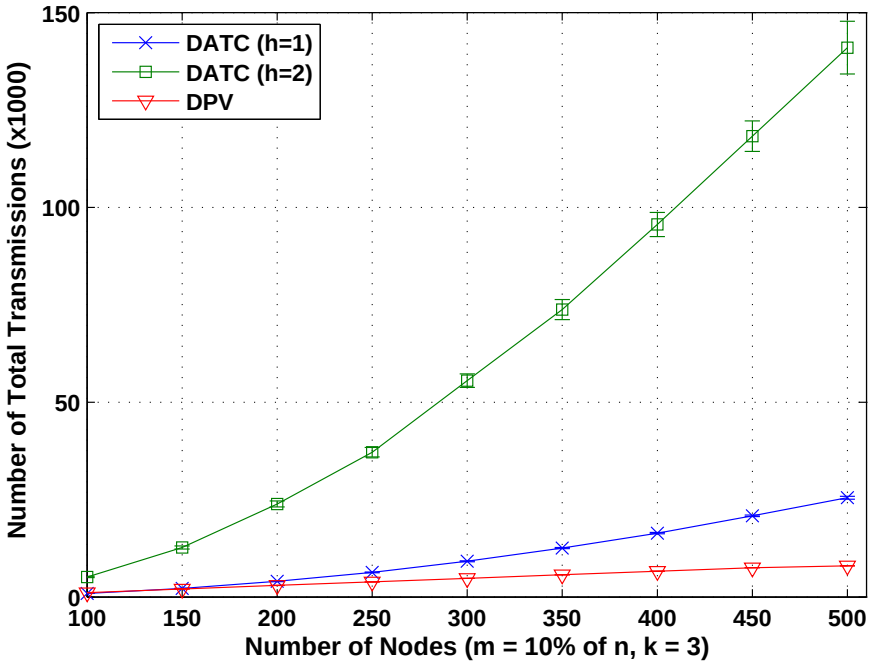


(b)

Figure 5.12: Number of Total Transmissions for k = 2.

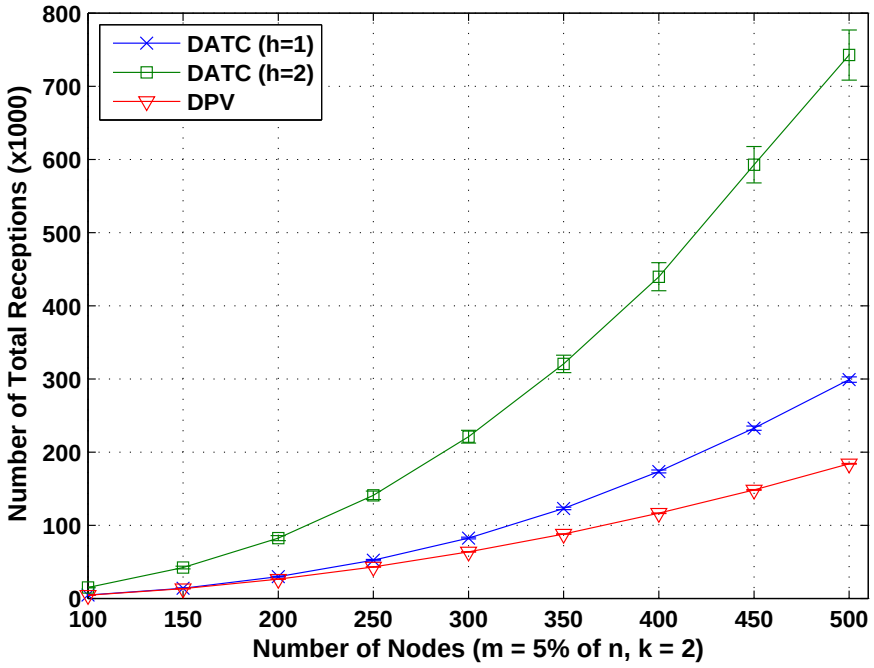


(a)

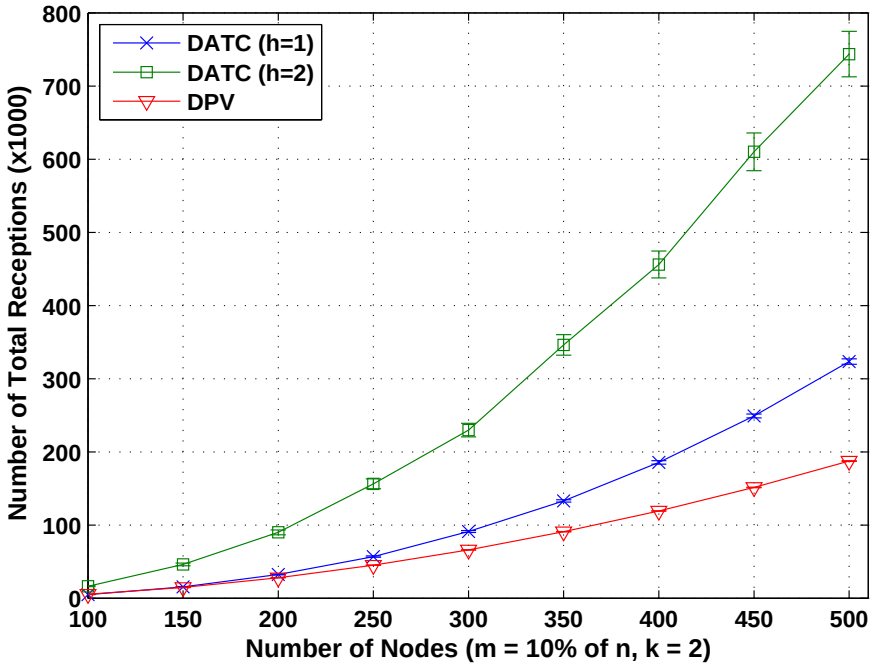


(b)

Figure 5.13: Number of Total Transmissions for k = 3.

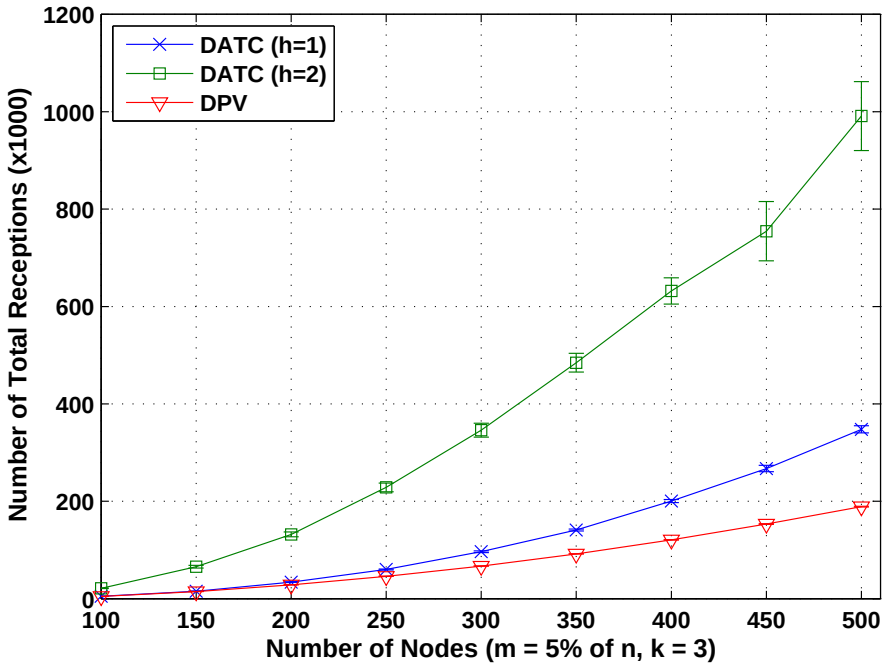


(a)

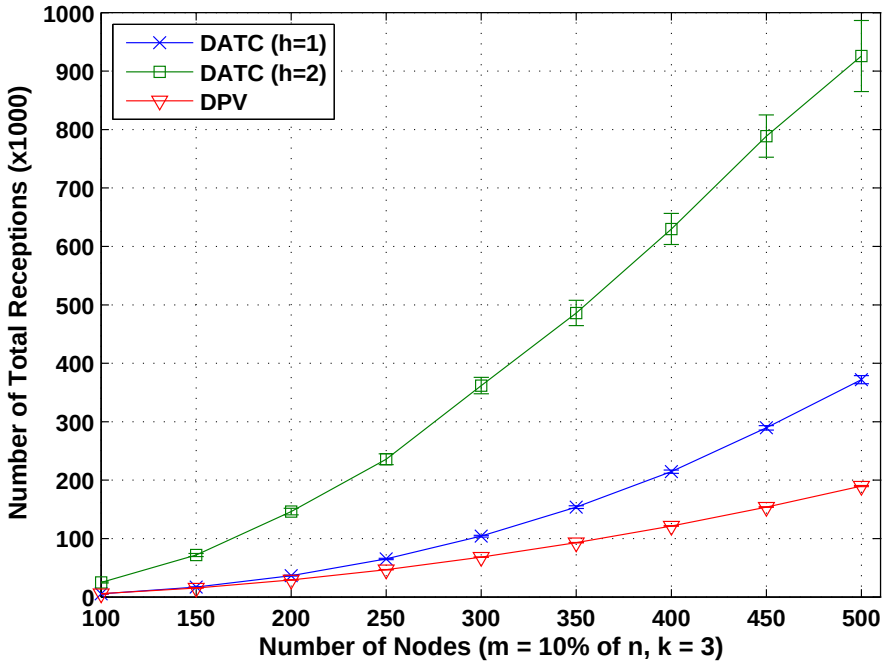


(b)

Figure 5.14: Number of Total Receptions for k = 2.



(a)



(b)

Figure 5.15: Number of Total Receptions for k = 3.

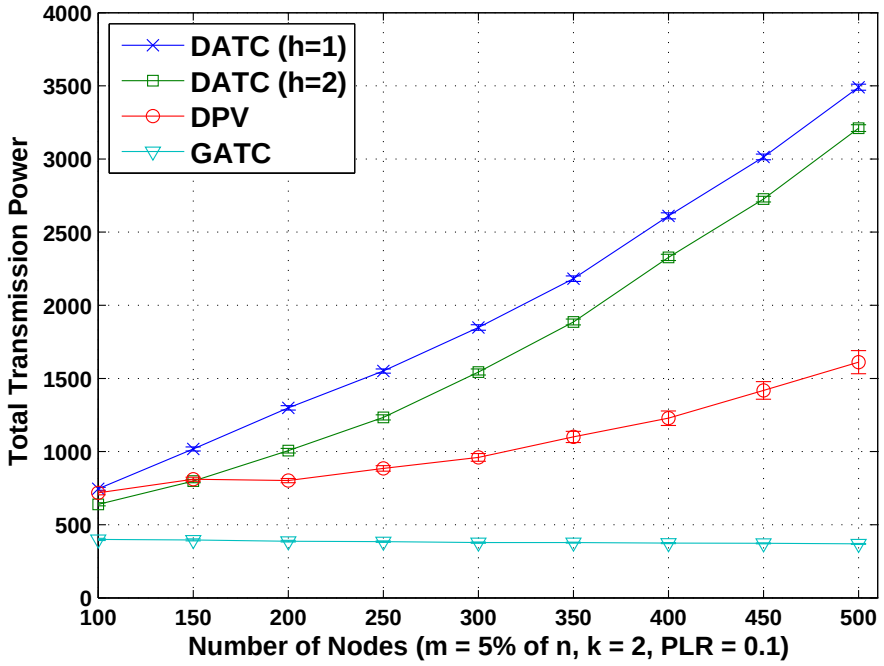
observation is that the number of receptions is about five times greater than the number of transmissions. This result is expected because when a message is transmitted it is received by multiple nodes in the 1-hop neighborhood.

5.3.6 Effect of Packet Losses

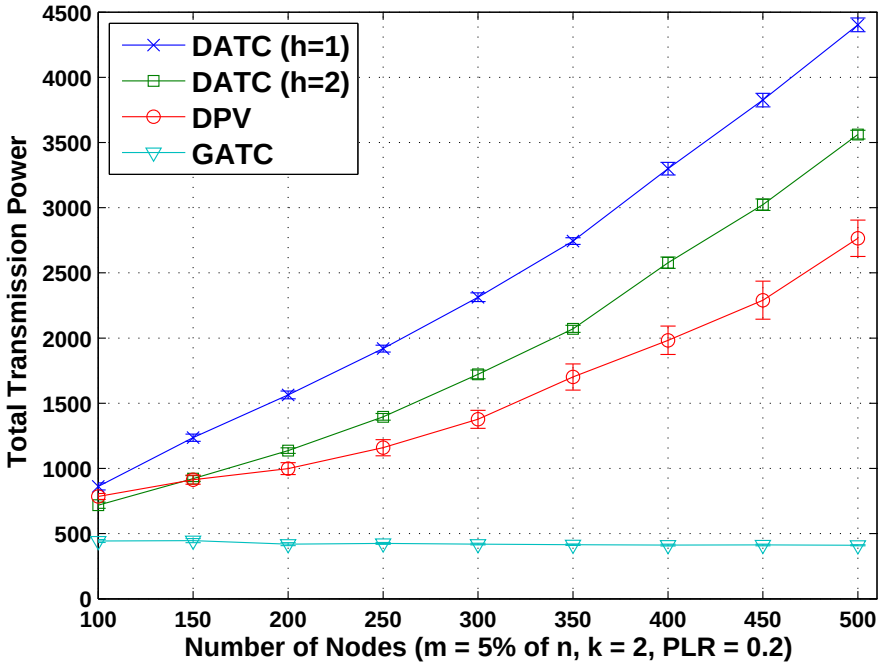
In wireless transmission medium packet losses are frequent due to collisions, link errors and environmental conditions. In order to justify our results, we present an additional scenario where packet losses can occur randomly in any link in the network with a given probability. In this scenario, retransmissions are also considered and maximum number of retransmissions is set to be 3. We change the packet loss ratio (PLR) between 0.1 and 0.4 with 0.1 steps and present the effect of packet losses on total transmission power in Figures 5.16(a), 5.16(b), 5.17(a) and 5.17(b).

We observe that when PLR increases total transmission power also increases. This is an expected result because under a higher PLR, a higher number of information exchange messages are lost and thus nodes can obtain an incomplete topology information from their neighbors. Using this partial information all three algorithms miss some of the more efficient paths because messages carrying that information were lost due to communication faults.

As seen in Figures 5.16(a) and 5.16(b), our algorithm DPV performs better than DATC(h=1) and DATC(h=2) as in the previous scenarios without packet loss consideration. However we see that the reduction in total power is not as high as in the previous scenario, it drops to 30% when PLR is 0.2. When PLR is 0.3 and higher DATC(h=2) starts to perform better than our algorithm in terms of total transmission power of the final topologies. This is due to the fact that DATC algorithms are more localized than DPV because they only use 1-hop or 2-hop neighborhood information which means their messages go only 1 or 2-hops. On the other hand DPV algorithm gathers path information from a wider area where messages can traverse up to the predefined TTL value which is typically 6 or 7. Since messages traverse longer paths in the DPV algorithm, these messages are more prone to packet losses. To elaborate, succesful delivery chance of a message over a 2-hop path is 3 times higher compared to a 5-hop path when PLR is 0.3. However a PLR ratio as high as 0.3 per link is not realistic. In [7] it is reported that between 0 and 70 meters the average PLR is under 0.1 for Crossbow IRIS mote. In addition, total number of control message transmissions for DATC(h=2) is much higher than DPV due to the need of power update messages in 2-hop neighborhood as seen in Figures 5.18(a), 5.18(b), 5.19(a) and 5.19(b). DATC(h=1) still results with the highest total transmission power due to its limited search scope as discussed in 5.3.1. Therefore we can conclude that under realistic PLR values, our algorithm DPV still outperforms DATC algorithms in terms of total transmission power and total number of transmissions.

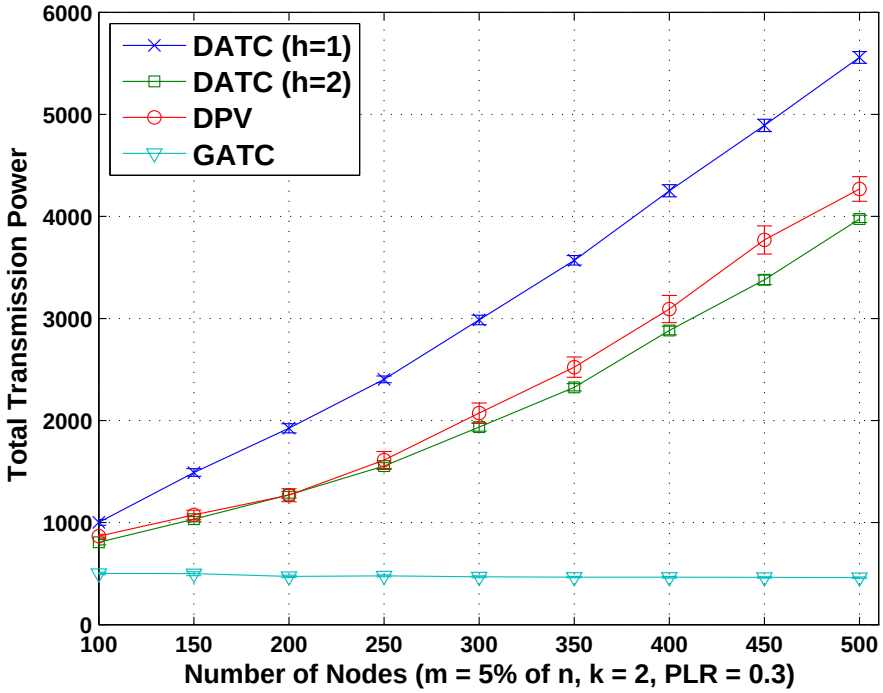


(a)

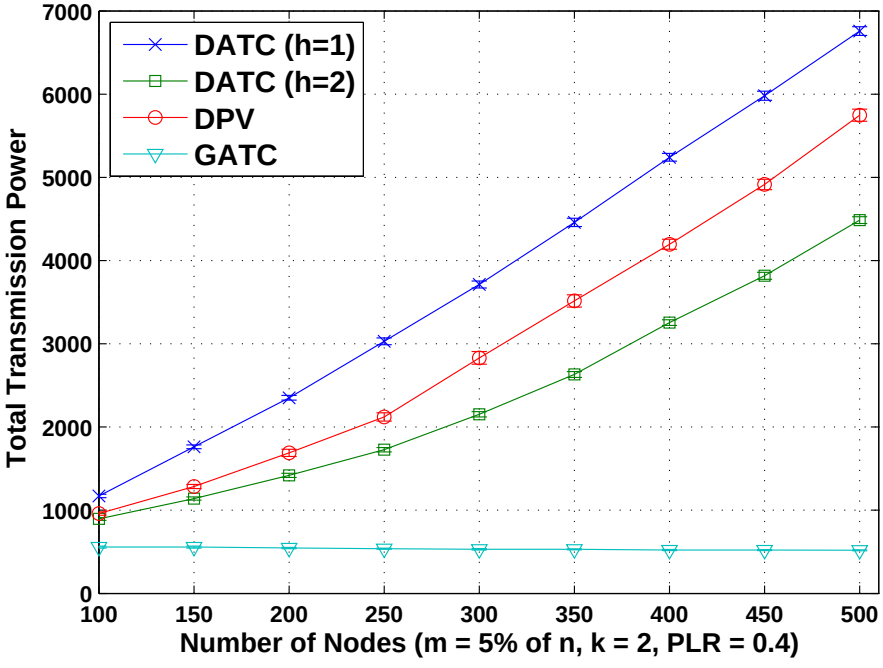


(b)

Figure 5.16: Total Transmission Power for $k = 2$ with PLR 0.1 and 0.2

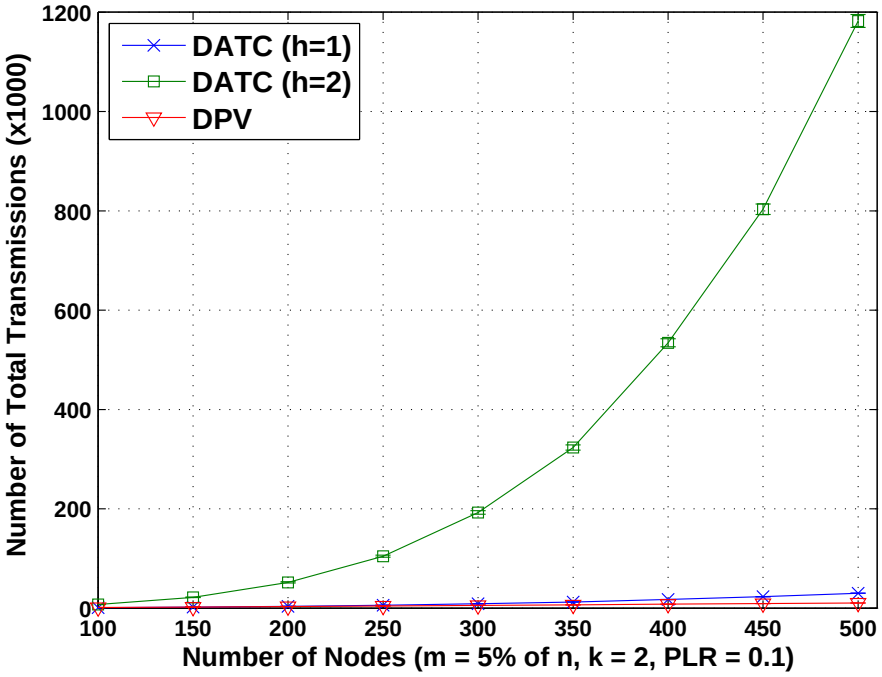


(a)

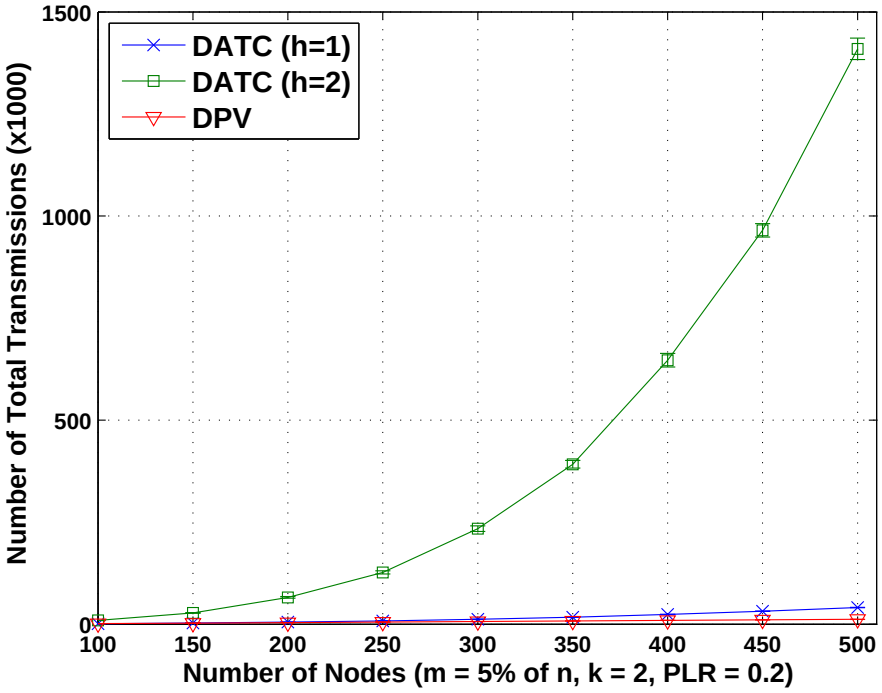


(b)

Figure 5.17: Total Transmission Power for $k = 2$ with PLR 0.3 and 0.4

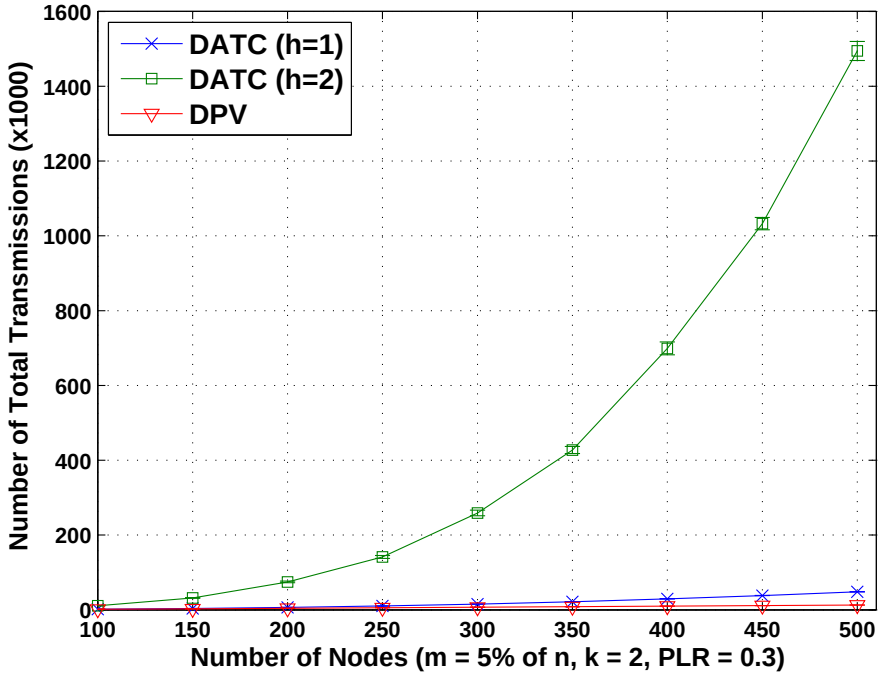


(a)

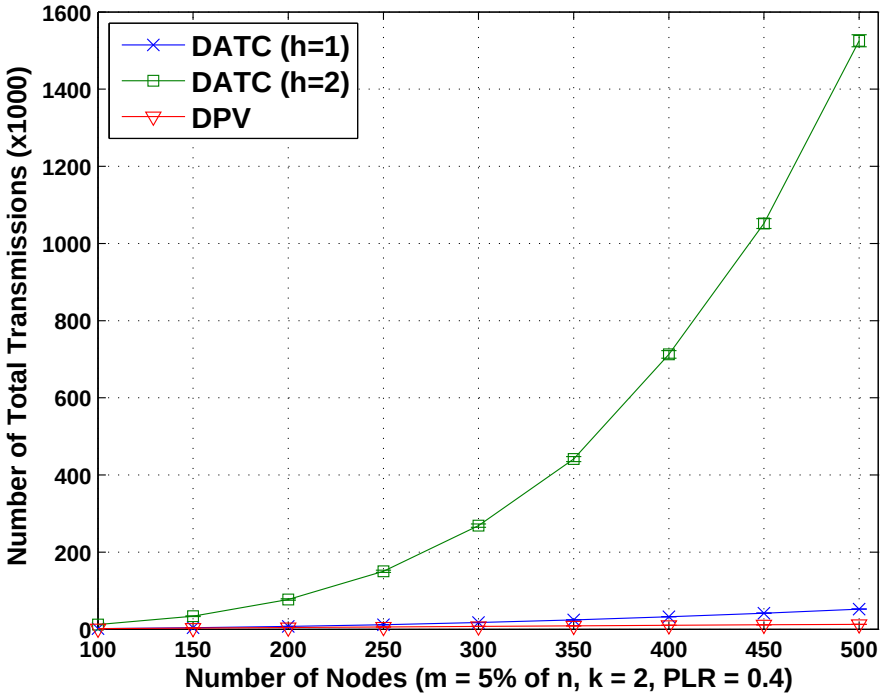


(b)

Figure 5.18: Total Number of Transmissions for $k = 2$ with PLR 0.1 and 0.2



(a)



(b)

Figure 5.19: Total Number of Transmissions for $k = 2$ with PLR 0.3 and 0.4

CHAPTER 6

CONCLUSION

Wireless sensor networks (WSNs) have been widely recognized as a promising technology and studied extensively for their advantages over traditional communication technologies including their low cost and easy deployment without any infrastructure. However, WSNs have their unique challenges and requirements such as energy efficiency and fault-tolerant operation. Recent studies have revealed that heterogeneous WSNs, where the network consists of different kind of sensors can result better energy efficiency and a higher fault tolerance compared to homogeneous wireless sensor networks [59]. In this work, we focus on heterogeneous two-layered network architectures where the lower layer consists of ordinary sensor nodes and the upper layer consists of resource-rich supernodes. In such topologies, sensor nodes forward their data towards supernodes using multi-hop paths. Supernodes collect and process the incoming data and can make critical decisions based on the application. For some scenarios, data delivery can be critical, so a fault-tolerant topology would be essential where each node has a certain degree of connectivity to the set of supernodes.

In this thesis, we introduce a new distributed algorithm, called the DPV (Disjoint Path Vector Algorithm), for constructing fault-tolerant topologies for heterogeneous wireless sensor networks consisting of supernodes and ordinary sensor nodes. There are many topology control algorithms in literature for ad hoc and wireless sensor networks as discussed in Section 2.3 and Chapter 3. However, not all of these algorithms consider fault tolerance which is a critical requirement for wireless sensor network applications. In addition, most of the existing topology control algorithms target connectivity between any two nodes in the network, but typically the connectivity between sensor nodes and the sink nodes is important for most of the WSN applications. Also, we noticed that there are a few topology control algorithms for layered network architectures where each layer consists of different type of sensor nodes. With this thesis we contribute with the DPV algorithm, which fulfills all the requirements mentioned above.

The DPV algorithm results in topologies where each sensor node in the network has at least k-vertex disjoint paths to the supernodes provided that the given network topology is k-vertex supernode connected. The objective of the algorithm is to minimize the total transmission power of the nodes in the network. The DPV algorithm is run by each sensor in a distributed manner with locally collected data. Using this data each node computes the set of required neighbors that guarantees k-vertex supernode connectivity. The changes in the neighbor lists are exchanged in an efficient way to keep the message overhead small. We showed that the generated topologies are k-vertex supernode connected and bidirectional with formal proofs in Section 4.5.

We evaluate our algorithm using a custom simulator that we developed, which enables generating random network topologies, executing the algorithms on generated topologies, calculating the desired metrics and visualizing the outputs of the experiments. We have implemented the DPV, GATC and DATC algorithms using this simulator and we compare the results, namely, total transmission power and maximum transmission power among all the sensors in the resulting topologies generated by each algorithm, as well as the number of total transmissions and receptions required for executing the algorithms. The simulation results show that our approach outperforms the existing distributed algorithms in all aspects. Compared to existing solutions, our DPV algorithm achieves a 4-fold reduction in total transmission power and a 2-fold reduction in maximum transmission power. Under realistic packet loss scenarios our algorithm results with 60% decrease in total transmission power. In other words, topologies generated by our algorithm use up to four times less power compared to topologies generated by the existing algorithms. In addition, our algorithm achieves these results by requiring fewer message transmissions and receptions. The most important contribution of the DPV algorithm is to generate topologies that have total transmission powers close to optimum, i.e., that is achieved by the centralized GATC algorithm. As mentioned before, the GATC is a centralized algorithm and requires global network information. Therefore it is less practical for large-scale networks. The solution we propose is, however, distributed and localized, thus is scalable to large networks and therefore suitable for use in real applications.

Future work could construct power-balanced topologies, which could be achieved by considering the remaining energy levels of sensor nodes while selecting the required neighbors. Such a topology would be more stable because it would prevent early node failures due to energy depletion.

REFERENCES

[1] A. Kashyap, S. Khuller, and M. Shayman. Relay Placement for Higher Order Connectivity in Wireless Sensor Networks. pages 1–12, April 2006.

[2] A. Mainwaring, J. Polastre, R. Szewczyk, , D. Culler, and J. Anderson. Wireless sensor networks for habitat monitoring. In *In Proceedings of ACM Wireless Sensor Networks and Applications (WSNA)*, pages 88–97, 2002.

[3] A. Milenkovic, C. Otto and E. Jovanov. Wireless sensor networks for personal health monitoring: Issues and an implementation. *Computer Communications*, 29(13):2521 – 2533, 2006.

[4] A.C.C. Yao. On constructing minimum spanning trees in k-dimensional spaces and related problems. *SIAM J. Comput.*, 11(4):721–736, 1982.

[5] A.S. Tanenbaum, and M. Van Steen. *Distributed systems*. Citeseer, 2002.

[6] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *Wireless Networks*, 8(5):481–494, 2002.

[7] C. Cirstea, M. Cernaianu, and A. Gontean. Packet Loss Analysis in Wireless Sensor Networks Routing Protocols. In *Telecommunications and Signal Processing 35th International Conference*, 2012.

[8] C. Gomez, and J. Paradells. Wireless home automation networks: A survey of architectures and technologies. *Communications Magazine, IEEE*, 48(6):92 –101, june 2010.

[9] C. Hua, and T.-S. P. Yum. Asynchronous random sleeping for sensor networks. *ACM Transactions on Sensor Networks*, 3(3):15, 2007.

[10] C. Wenjie, C. Lifeng, C. Zhanglong, and T. Shiliang. A realtime dynamic traffic control system based on wireless sensor network. In *Parallel Processing, 2005. ICPP 2005 Workshops. International Conference Workshops on*, pages 258 – 264, june 2005.

[11] C.E. Jones, K.M. Sivalingam, P. Agrawal, and J.C. Chen. A Survey of Energy Efficient Network Protocols for Wireless Networks. *Wireless Networks*, 7(4):343–358, 2001.

[12] C.F. Huang, Y.C. Tseng, S.L. Wu, and J.P Sheu. Distributed topology control algorithm for multihop wireless networks. In *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on*, volume 1, pages 355–360, 2002.

[13] C.S.R Murthy, and B.S. Manoj. Wireless sensor networks. In *Ad Hoc Wireless Networks*, pages 647–696. Prentice Hall, 2004.

[14] D. Chen, D.-Z. Du, X.-D. Hu, G.-H. Lin, L. Wang, and G. Xue. Approximations for steiner trees with minimum number of steiner points. *Journal of Global Optimization*, 18(1-3):17–33, 2000.

[15] D. Steere, A. Baptista, D. McNamee, C. Pu, and J. Walpole. Research challenges in environmental observation and forecasting systems. In *In Proceedings of ACM Mobicom*, pages 292–299, 2000.

[16] D.M. Blough, M. Leoncini, G. Resta, and P. Santi. The k-neigh protocol for symmetric topology control in ad hoc networks. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, page 152. ACM, 2003.

[17] E. Cayirci, and T. Coplu. Sendrom: sensor networks for disaster relief operations management. *Wireless Networks (Springer Journal)*, 13(3):409–423, 2007.

[18] E.L. Lloyd, and G. Xue. Relay Node Placement in Wireless Sensor Networks. *IEEE Trans. Computers*, 56(1):134–138, 2007.

[19] F. Koushanfar, M. Potkonjak, and A. Sangiovanni-Vincentell. Fault tolerance techniques for wireless ad hoc sensor networks. In *Sensors, 2002. Proceedings of IEEE*, volume 2, pages 1491 – 1496 vol.2, 2002.

[20] F. Wang, M.T. Thai, and D.Z. Du. On the construction of 2-connected virtual backbone in wireless network. *IEEE Transactions on Wireless Communications*, 8(3):1230–137, 2009.

[21] G. Anastasi, M. Conti, M. Francesco, and A. Passarella. Energy conservation in wireless sensor networks: A survey. *Ad Hoc Networks*, 7(3):537–568, 2009.

[22] G. Srivastava, P. Boustead, and J. Chicharo. Connected fixed node degree based topologies in ad hoc networks. *Computer Communications*, 29(9):1330–1340, 2006.

[23] H. Alemdar, and C. Ersoy. Wireless sensor networks for healthcare: A survey. *Computer Networks*, 54(15):2688 – 2710, 2010.

[24] H. Alwan, and A. Agarwal. A survey on fault tolerant routing techniques in wireless sensor networks. In *Sensor Technologies and Applications, 2009. SENSORCOMM '09. Third International Conference on*, pages 366 –371, june 2009.

[25] H. Liu, A. Nayak, and I. Stojmenovic. Fault-Tolerant Algorithms/Protocols in Wireless Sensor Networks. *Guide to Wireless Sensor Networks*, pages 265–295, 2009.

[26] H.B. Lim, D. Ma, B. Wang, Z. Kalbarczyk, R.K. Iyer, and K.L. Watkin. A soldier health monitoring system for military applications. In *Body Sensor Networks (BSN), 2010 International Conference on*, pages 246 –249, june 2010.

[27] H.K. Qureshi, S. Rizvi, M. Saleem, S.A. Khayam, V. Rakocevic, and M. Rajarajan. Extended dominating set and its applications in ad hoc networks using cooperative communication. *Computer Communications*, 34:1235–1242, 2001.

[28] I. Saha, L.K. Sambasivan, S.K. Ghosh, and R.K. Patro. Distributed fault-tolerant topology control in wireless multi-hop networks. *Wireless Networks*, 16:1511–1524, 2010.

[29] I.F. Akyildiz, and I.H. Kasimoglu. Wireless sensor and actor networks: research challenges. *Ad Hoc Networks*, 2(4):351–367, 2004.

[30] J. Duan, Y. Qin, S. Zhang, and Z. Tao. Improved deployment of wireless sensor networks in the intelligent building. In *Advanced Intelligence and Awareness Internet (AIAI 2011), 2011 International Conference on*, pages 88 –92, oct. 2011.

[31] J. Polastre, R. Szewczyk, A. Mainwaring, D. Culler, and J. Anderson. Analysis of wireless sensor networks for habitat monitoring. In *Wireless Sensor Networks*, pages 399–423. Springer US, 2004.

[32] J. Tang, B. Hao, and A. Sen. Relay Node Placement in Large Scale Wireless Sensor Networks. *Computer Communications*, 29(4):490–501, 2006.

[33] J. Wu, F. Dai, M. Gao, and I. Stojmenovic. On calculating power-aware connected dominating sets for efficient routing in ad hoc wireless networks. *IEEE/KICS Journal of Communications and Networks*, 4:59–70, 2002.

[34] J. Wu, M. Cardei, F. Dai, and S. Yang. Extended dominating set and its applications in ad hoc networks using cooperative communication. *IEEE Transactions on Parallel and Distributed Systems*, 17(8):851–864, 2006.

[35] J. Wu, S. Yang, and M. Cardei. On Maintaining Sensor-Actor Connectivity in Wireless Sensor and Actor Networks. In *IEEE INFOCOM*, pages 888–896, 2008.

[36] J. Yick, B. Mukherjee, and D. Ghosal. Wireless sensor network survey. *Computer Networks*, 52(12):2292–2330, 2008.

[37] J.B. Kruskal. On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. *Proceedings of the American Mathematical Society*, 7(1):48–50, 1956.

[38] J.W. Jaromczyk, and G.T. Toussaint. Relative neighborhood graphs and their relatives. *Proceedings of the IEEE*, 80(9):1502–1517, 1992.

[39] K. JeongGil, L. Chenyang, M.B. Srivastava, J.A. Stankovic, A. Terzis, and M. Welsh. Wireless sensor networks for healthcare. *Proceedings of the IEEE*, 98(11):1947 –1960, nov. 2010.

[40] K. Ozaki, K. Watanabe, S. Itaya, N. Hayashibara, T. Enokido, and M. Takizawa. A Fault-Tolerant Model for Wireless Sensor-Actor System. In *Proceedings of the 20th International Conference on Advanced Information Networking and Applications (AINA 06)*, 2006.

[41] K. Srinivasan, M. Ndoh, H. Nie, H. Xia, K. Kaluri, and D. Ingraham. Wireless technologies for condition-based maintenance (cbm) in petroleum plants. In *Proc. of DCOSS'05 (Poster Session)*, 2005.

[42] K.J. Supowit. The relative neighbourhood graph with application to minimum spanning trees. *Journal of the ACM*, 30(3):428–448, 1983.

[43] K.K. Mamidisetty, M.J. Ferrara, S. Sastry. Systematic selection of cluster heads for data collection. *Journal of Network and Computer Applications*, 35:1548–1558, 2012.

[44] K.R. Gabriel, and R.R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Biology*, 18(3):259, 1969.

[45] L. Ding, W. Wu, J. Willson, H. Du, W. Lee, and D-Z. Du. Efficient Algorithms for Topology Control Problem with Routing Cost Constraints in Wireless Networks. *IEEE Transactions on Parallel and Distributed Systems*, 22(10):1601–1609, 2011.

[46] L. Li, J. Halpern, P. Bahl, Y. Wang, and R. Wattenhofer. A cone-based distributed topology control algorithm for wireless multi-hop networks. *IEEE/ACM Transactions on Networking*, 13(1):147–159, 2005.

[47] L. M. L. Oliveira and J. J. P. C. Rodrigues. Wireless sensor networks: a survey on environmental monitoring. *Journal of Communications*, 6(2):143–151, april 2011.

[48] L. Paradis, and Q. Han. A survey of fault management in wireless sensor networks. *Journal of Network and Systems Management*, 15(2):171–190, 2007.

[49] L. Schwiebert, S. Gupta, and J. Weinmann. Research challenges in wireless networks of biomedical sensors. In *In Proceedings of ACM Mobicom*, pages 151–165, 2001.

[50] L. Wang, H. Jin, J. Dang, and Y. Jin. A fault tolerant topology control algorithm for large-scale sensor networks. In *Parallel and Distributed Computing, Applications and Technologies, 2007. PDCAT'07. Eighth International Conference on*, pages 407–412, 2007.

[51] M. Cardei, S. Yang, and J. Wu. Algorithms for Fault-Tolerant Topology in Heterogeneous Wireless Sensor Networks. *IEEE Transactions on Parallel and Distributed Systems*, 19(4):545–558, 2008.

[52] M. Franceschinis, L. Gioanola, M. Messere, R. Tomasi, M.A. Spirito, and P. Civera. Wireless sensor networks for intelligent transportation systems. In *Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th*, pages 1 –5, april 2009.

[53] M. Hefeeda, and M. Bagheri. Wireless sensor networks for early detection of forest fires. In *Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE Internatonal Conference on*, pages 1 –6, oct. 2007.

[54] M. Kubisch, H. Karl, A. Wolisz, L. Zhong, and J. Rabaey. Distributed algorithms for transmission power control in wireless sensor networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, pages 16–20, 2003.

[55] M. Li, and B. Yang. A survey on topology issues in wireless sensor network. In *Proceedings of the International Conference on Wireless Networks*. Citeseer, 2006.

[56] M. Li, and Y. Liu. Underground coal mine monitoring with wireless sensor networks. *ACM Trans. Sen. Netw.*, 5(2):10:1–10:29, Apr. 2009.

[57] M. Min, H. Du, X. Jia, C. X. Huang, S.C.-H. Huang, and W. Wu. Improving construction for connected dominating set with steiner tree in wireless sensor networks. *Journal of Global Optimization*, 35(1):111–119, 2006.

[58] M. Srivastava, R. Muntz, and M. Potkonjak. Smart kindergarten: Sensor-based wireless networks for smart developmental problem-solving environments. In *In Proceedings of ACM Mobicom*, pages 132–138, 2001.

[59] M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu, and S. Singh. Exploiting heterogeneity in sensor networks. In *Proceedings of the IEEE International Conference on Computer Communication Volume 2*, pages 878–890, Miami, FL, USA, 2005.

[60] M.P. Durisic, Z. Tafa, G. Dimic, and V. Milutinovic. A survey of military applications of wireless sensor networks. In *Embedded Computing (MECO), 2012 Mediterranean Conference on*, pages 196 –199, june 2012.

[61] N. Li, and J.C. Hou. Flss: A fault-tolerant topology control algorithm for wireless networks. In *Proc. 10th ACM Int'l Conf. Mobile Computing and Networking (MOBICOM)*, pages 275–286, 2004.

[62] N. Li, and J.C. Hou. Topology control in heterogeneous wireless networks: problems and solutions. In *Proceedings of the IEEE Conference on Computer Commnications (INFOCOM)*, pages 232–243, 2004.

[63] N. Li, and J.C. Hou. Localized topology control algorithms for heterogeneous wireless networks. *IEEE/ACM Transactions on Networking*, 13(5):1313–1324, 2005.

[64] N. Li, and J.C. Hou. Localized fault-tolerant topology control in wireless ad hoc networks. *IEEE Transactions on Parallel and Distributed Systems*, 17(4):307–320, 2006.

[65] N. Li, J.C. Hou, and L. Sha. Design and analysis of an MST-based topology control algorithm. *IEEE Transactions on Wireless Communications*, 4(3):1195–1206, 2005.

[66] N. Pogkas, G. Karastergios, C. Antonopoulos, S. Koubias, and G. Papadopoulos. An ad-hoc sensor network for disaster relief operations. In *Proceedings of the 10th IEEE conference on Emerging Technologies and Factory Automation*, 2005.

[67] O. Younis, and S. Fahmy. Heed: A hybrid, energy efficient, distributed clustering approach for ad hoc sensor networks. *IEEE Transactions on Mobile Computing*, 3(4):660–669, 2004.

[68] O. Younis, M. Krunz, and S. Ramasubramanian. Node Clustering in Wireless Sensor Networks: Recent Developments and Deployment Challenges. In *IEEE Network*, pages 20–25, 2006.

[69] P. Santi. Topology control in wireless ad hoc and sensor networks. *ACM Computing Surveys (CSUR)*, 37(2):164–194, 2005.

[70] P.M. Wightman, and M.A. Labrador. A3: A topology construction algorithm for wireless sensor networks. In *In Proceedings of IEEE Globecom*, 2008.

[71] R. Ramanathan, and R. Kosales-Hain. Topology control of multihop wireless networks using transmit power adjustment. In *IEEE INFOCOM*, volume 2, pages 404–413. Citeseer, 2000.

[72] R. Wattenhofer, L. Li, P. Bahl, and Y.M. Wang. Distributed topology control for power efficient operation in multihop wireless ad hoc networks. In *IEEE INFOCOM*, volume 3, pages 1388–1397. Citeseer, 2001.

[73] R.C. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36:1389–1401, 1957.

[74] S. Banerjee, and S. Khuller. A Clustering Scheme for Hierarchical Control in Multihop Wireless Networks. In *IEEE INFOCOM*, pages 1028–1037, 2001.

[75] S. Guha, and S. Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20(4):374–387, 1996.

[76] S. Kumar, T. H. Lai, and J. Balogh. On k-coverage in a mostly sleeping sensor network. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 144–158, 2004.

[77] S.A. Borbash, and E.H. Jennings. Distributed topology control algorithm for multihop wireless networks. In *Proc. 2002 World Congress on Computational Intelligence (WCCI 2002)*, pages 355–360. Citeseer, 2002.

[78] S.Coleri, S.Y. Cheung, and P. Varaiya. Sensor networks for monitoring traffic. In *In Allerton Conference on Communication, Control and Computing*, 2004.

[79] S.M. George, W. Zhou, H. Chenji, M. Won, Y.O. Lee, A. Pazarloglou, R. Stoleru, and P. Barooah. Distressnet: a wireless ad hoc and sensor network architecture for situation management in disaster response. *Communications Magazine, IEEE*, 48(3):128 –136, march 2010.

[80] T. Bokareva, W. Hu, S. Kanhere, B. Ristic, N. Gordon, T. Bessell, M. Rutten, and S. Jha. Wireless sensor networks for battlefield surveillance. In *In Proc. of LWC*, 2006.

[81] T. Melodia, D. Pompili, V.C. Gungor and I.F. Akyildiz. A Distributed Coordination Framework for Wireless Sensor and Actor Networks. In *In Proceedings of the 6th ACM international*, pages 99–110, 2005.

[82] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. The algorithms of kruskal and prim. In *Introduction to Algorithms, Third Edition*, pages 631–638. MIT Press, 2009.

[83] T.T. Hsieh. Using sensor networks for highway and traffic applications. *Potentials, IEEE*, 23(2):13 – 16, 2004.

[84] V. Chvatal. A greedy heuristic for the set-covering problem. *Mathematics of operations research*, 4(3):233–235, 1979.

[85] V.C. Gungor, and G.P. Hancke. Industrial wireless sensor networks: Challenges, design principles, and technical approaches. *Industrial Electronics, IEEE Transactions on*, 56(10):4258 –4265, oct. 2009.

[86] W. Wu, H. Du, X. Jia, Y. Li, and S.C.-H. Huang. Minimum connected dominating sets and maximal independent sets in unit disk graphs. *Theoretical Computer Science*, 352(1):1–7, 2006.

[87] W.-Z. Song, X.-Y. Li, O. Frieder, and W. Wang. Localized Topology Control for Unicast and Broadcast in Wireless Ad Hoc Networks. *IEEE Transactions on Parallel and Distributed Systems*, 17(4):321–334, 2006.

[88] W.-Z. Song, Y. Wang, X.-Y. Li, and O. Frieder. Localized algorithms for energy efficient topology in wireless ad hoc networks. In *Proc. ACM International Symp. Mobile Ad-Hoc Networking and Computing (MobiHoc)*, 2004.

[89] W. Zhang, G. Xue, and S. Misra. Fault-Tolerant Relay Node Placement in Wireless Sensor Networks: Problems and Algorithms. pages 1649–1657, 2007.

[90] X. Cheng, D. Du, L. Wang, and B. Xu. Relay Sensor Placement in Wireless Sensor Networks. *Wireless Networks*, 14(3):347–355, 2008.

[91] X. Han, X. Cao, E.L. Lloyd, and C.-C. Shen. Fault-Tolerant Relay Node Placement in Heterogeneous Wireless Sensor Networks. *IEEE Transactions on Mobile Computing*, 9(5):643–656, 2010.

[92] X. Laisheng and P. Xiaohong and W. Zhengxia and X. Bing and H. Pengzhi. Research on traffic monitoring network and its traffic flow forecast and congestion control model based on wireless sensor networks. In *Measuring Technology and Mechatronics Automation, 2009. ICMTMA '09. International Conference on*, volume 1, pages 142 –147, april 2009.

[93] X. Li, Y. Wang, and W. Song. Applications of k-local mst for topology control and broadcasting in wireless ad hoc networks. *IEEE Transactions on Parallel and Distributed Systems*, 15(12):1057–1069, 2004.

[94] X. Zhang, X. Ding, S. Lu, and G. Chen. Principles for energy-efficient topology control in wireless sensor networks. In *Proceedings of the 5th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM'09)*, pages 1–3, 2009.

[95] Y. Chen, and S.H. Son. A fault tolerant topology control in wireless sensor networks. In *Computer Systems and Applications, 2005. The 3rd ACS/IEEE International Conference on*, 2005.

[96] Y. Huang, M. Hsieh, and F. Sandnes. Wireless sensor networks and applications. In *Sensors*, volume 21 of *Lecture Notes Electrical Engineering*, pages 199–219. Springer Berlin Heidelberg, 2008.

[97] Y. M. Baryshnikov, E. G. Coffman, and K. J. Kwak. High performance sleep-wake sensor systems based on cyclic cellular automata. In *Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN)*, pages 517–526, 2008.

[98] Y. Wang. Topology control for wireless sensor networks. *Wireless Sensor Networks and Applications*, pages 113–147, 2008.

[99] Y. Wu, F. Wang, M.T. Thai, and Y. Li. Constructing k-connected m-dominating sets in wireless sensor networks. In *Military Communications Conference*, October 2007.

[100] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad hoc routing. In *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, pages 70–84, 2001.

[101] Z. Sun, P. Wanga, M.C. Vuran, M.A. Al-Rodhaan, A.M. Al-Dhelaan, and I. F. Akyildiz. Bordersense: Border patrol through advanced wireless sensor networks. *Ad Hoc Networks*, 9(3):468 – 477, 2011.

[102] Z. Yuanyuan, X. Jia, and H. Yanxiang. Energy efficient distributed connected dominating sets construction in wireless sensor networks. In *Proceedings of the ACM International Conference on Communications and Mobile Computing*, pages 797–802, 2006.

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Bağcı, Hakkı
Nationality: Turkish (TC)
Date and Place of Birth: 23-Feb-1982, Beypazarı
Marital Status: Married
Phone: +90 506 541 65 57
E-Mail: hakkibagci@gmail.com

EDUCATION

Degree	Institution	Year of Graduation
M.S.	Bilkent University, Computer Engineering	2007
B.S.	Bilkent University, Computer Engineering	2004

PROFESSIONAL EXPERIENCE

Year	Place	Enrollment
2013 - ...	TUBİTAK / BİLGEM / İLTAREN	Chief Researcher
2009 - 2013	TUBİTAK / BİLGEM / İLTAREN	Senior Researcher
2004 - 2009	TUBİTAK / BİLGEM / İLTAREN	Researcher

PUBLICATIONS

International Journal Publications

Hakki Bagci and Ibrahim Korpeoglu, Distributed and Location Based Multicast Routing Algorithms for Wireless Sensor Networks. EURASIP Journal on Wireless Communications and Networking, vol. 2009, Article ID 697373, 14 Pages, 2009.