Unless in communicating with it (computer) one says exactly what one means, trouble is bound to result.

– Alan Turing.

University of Alberta

RELIABLE WIRELESS SENSOR NETWORKS USING MULTIPLE SINKS AND DEGREE CONSTRAINED SHORTEST PATH TREES

by

Mohammad Saiful Islam

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

Spring 2013
Edmonton, Alberta

To my mother
Without whose help, I would not come this far.

Abstract

Wireless Sensor networks (WSN) have gained attention in both industry and academia due to their versatile fields of application. One of the main characteristics of a sensor node is its limited energy supply. The network needs to be reliable in the sense that it can deliver the data to sink with the presence of multiple link failures. Having more than one sink provides alternative paths to route packets in the presence of link failures and helps load balancing. Degree Constrained Shortest Path Trees (DCSPT) can be used as routing trees to limit the communication of sensor nodes thus conserving energy. In this thesis, we consider the problem of designing logical topologies for routing in WSNs with multiple sinks. We design reliable logical topologies most of which are based on DCSPT in grid and random graphs. We also design scheduling and routing algorithms for the logical topologies and evaluate the reliability of the designs using simulation. We demonstrate that with moderate link failures our schemes can reliably transfer data with low delay and the performance improves as the load of the network decreases.

Acknowledgements

First of all, my deepest graitude to Almighty who provided me the ability to complete the dissertation successfully.

I express my heartiest gratitude to my supervisor Dr. Janelle Harms for her priceless guidance, helpful criticisms, valuable suggestions, continuous encouragement and enthusiasm throughout the course of this work. It was a great pleasure and learning experience working with Dr. Harms. I would also like to thank my committee members for their time and attention for reviewing my work.

Last but not the least, I express my thankfulness to my family and friends for their wonderful support, cordial cooperation and inspiration during these days to complete my research successfully.

Table of Contents

1 Introduction **1**
- 1.1 Wireless Sensor Network . . . 1
- 1.2 Motivation . . . 2
- 1.3 Thesis Contribution . . . 4
- 1.4 Thesis Outline . . . 6

2 Background Study **7**
- 2.1 Reliability in Wireless Sensor Networks . . . 8
  - 2.1.1 Reliable Data Transfer in WSN . . . 8
  - 2.1.2 Retransmission-based Schemes . . . 9
  - 2.1.3 Multiple Path-based Schemes . . . 11
  - 2.1.4 Event Reliability Schemes . . . 12
- 2.2 Location of Multiple Sinks . . . 13
- 2.3 Routing with Multiple Sinks . . . 16
- 2.4 TDMA Scheduling . . . 17
- 2.5 Chapter Summary . . . 21

3 Design of Robust Topologies in Random Sensor Network Graphs **23**
- 3.1 DCSPT Algorithm . . . 24
- 3.2 Robust Topologies in Random Sensor Network Graphs . . . 25
  - 3.2.1 Experimental Results . . . 27
- 3.3 Static Analysis of the Logical Topologies . . . 31
  - 3.3.1 Experimental Setup . . . 33
  - 3.3.2 Experimental Results . . . 33
- 3.4 Chapter Summary . . . 35

4 Dynamic Analysis **38**
- 4.1 TDMA Scheduling . . . 39
  - 4.1.1 Conflict Graphs for a Single Routing Tree . . . 39
  - 4.1.2 Algorithms for Building Conflict Graphs for Multiple Trees 41
  - 4.1.3 Experiments to Compare the Scheduling Schemes . . . 44
- 4.2 Routing Protocols . . . 46
- 4.3 Experimental Setup . . . 50
  - 4.3.1 Frame Generation . . . 50
  - 4.3.2 Link Failure . . . 51
- 4.4 Simulation Metrics . . . 51
  - 4.4.1 Average time to deliver all packets . . . 51
- 4.5 Experiment Methodology . . . 52
  - 4.5.1 Assumptions . . . 53
  - 4.5.2 Design of Experiments . . . 54
- 4.6 Analysis of Results . . . 54
  - 4.6.1 Experiments with All Possible Sources . . . 54

4.7 Chapter Summary 59

5 Design of Robust Topologies in Grid Graph **62**
5.1 Problem Definition 63
5.2 Grid Graphs with Horizontal and Vertical Links 64
5.3 Grid Graphs with Multiple Sinks 65
5.4 Static Analysis of the Pattern 67
5.4.1 Experimental Setup 68
5.4.2 Experimental Results 68
5.5 Dynamic Analysis of the Pattern in Grid Graph 71
5.5.1 Experimental Setup 71
5.5.2 Experimental Results 72
5.6 Chapter Summary 75

6 Conclusion and Future Work **77**
6.1 Conclusions 77
6.2 Future Work 78

Bibliography **81**

A Sink Location from Sensor Node Position **86**
A.1 Algorithms 86
A.2 Experimental Method 88
A.3 Experimental Results 89
A.3.1 Comparsion with Edge-position and Edge-far scheme 92

B Algorithms for Building and Coloring Conflict Graphs **93**

List of Tables

3.1 Number of eligible pairs (out of 50) for varying node degree (sink degree constraint = 50) 29
3.2 Comparison between different sink position schemes 31

5.1 Quality of path 69

A.1 Comparison between different sink location schemes - eligible topologies (out of 100) 89
A.2 Comparison between Edge-far and Edge-position by possibility of getting eligible pair (out of 50), sink degree constraint = 5 92

B.1 Notations Used in the Algorithms 93

List of Figures

3.1 Search space for Edge-position scheme 28
3.2 Search space for Middle-position scheme 29
3.3 Locations of eligible sink pairs 31
3.4 Path quality of the logical topology 33
3.5 Sink load 34
3.6 Degree histogram 35
3.7 Degree of robustness of union graph in multi-link failure 36
3.8 Path quality in multi-link failure 36

4.1 Full tree scheduling 40
4.2 Pruned tree scheduling 42
4.3 Two parent scheduling 43
4.4 Average slot length of TDMA schedule 45
4.5 Average slot length of TDMA schedule - impact of node degree on schedule length 46
4.6 Routing loop in LinkAck-NextHR 48
4.7 Average time to deliver all the packets to the sink(s) 56
4.8 Average hop-count 57
4.9 Network lifetime 58
4.10 Effect of MIT on time to deliver all packets 58
4.11 Effect of MIT on NLT 59
4.12 Effect of number of sources on time to deliver all packets 60

5.1 Grid based topology 63
5.2 Pattern for the grid network with the leaves and the four quadrants connected 64
5.3 Multiple sinks in grid graph with diagonal links 65
5.4 Multiple sinks in grid graph a)$M < N$ b)$M > N$ 66
5.5 Pattern with reduced number of 3-degree nodes a)$M < N$ b)$M > N$ 68
5.6 Degree of robustness in multi-link failure grid graph 70
5.7 Path quality in multi-link failure grid graph 70
5.8 Average time to deliver all the packets to the sink(s) 72
5.9 Average hop-count 73
5.10 Network lifetime 74
5.11 Effect of number of sources on average time to deliver packets . . . 75

A.1 Comparison between different sink location schemes- path quality. . 90
A.2 Comparison between different sink location schemes- reliability. . . 90

Chapter 1

Introduction

1.1 Wireless Sensor Network

A wireless sensor network (WSN) [34] is a collection of sensor nodes deployed in a specific region primarily for sensing and communicating data among the nodes to deliver it to the user of the network. Advancement of wireless networking, embedded systems, integration of micro sensors and low cost of sensor nodes have made it possible to deploy large scale WSNs which can replace wired sensor networks. WSNs enable us to monitor physical phenomena in places where human presence is extremely difficult or where wiring and maintenance of wired networks are difficult. These sensor networks are able to support many new applications, including habitat monitoring [1], agricultural monitoring [8], environmental monitoring [19] and military applications. Large scale wireless sensor networks have already been deployed in processing plants, deep inside the forests, in volcanic mountains [47] or undersea [40].

Sensor nodes have two major functionalities. First, sensor nodes can sense various physical (light, sound, temperature, *etc.*) data. The type of data sensed depends on the sensor attached to the node. Second, the nodes can communicate with other nodes wirelessly to transmit their sensed data. For this they are equipped with wireless communication devices. Normally the nodes form a data gathering tree whose root is called the sink. The sensed data is transferred to the sink usually with multihop communication. This avoids transmission over long distances which consumes much energy [2]. The user can collect the data from the sink directly or via the In-

ternet where the sink acts as a gateway to the Internet. Though the main function of sensor nodes is information gathering, they can act as repeaters to relay messages towards sinks. WSNs often operate in challenging environments and are subject to frequent disruptions and node failures. These unique settings and constraints call for a robust routing framework for WSNs that can quickly adapt to changes in traffic pattern, network conditions and environments.

There are several properties of the wireless medium that make it different from the wired medium. Wireless nodes have a limited transmission range that depends on the power of the transmission signal. In a radio wireless medium, communication is broadcasting. So, when a node transmits some data, every other node within the range of the transmitting node, can receive the data. Wireless communication is half-duplex and that means a node cannot transmit and receive at the same time. If a node receives from multiple neighbors at the same time, a collision occurs and the signals may not be recovered. The wireless medium is susceptible to various types of transmission hindrance such as path loss, fading and interference which restricts the range, data rate and the reliability of wireless transmission and the effect of these factors on transmission depends on the environment.

1.2 Motivation

Despite the wide use of WSN, there are some limitations of the sensor nodes used in such networks: limited energy, limited processing power and limited memory. Having limited energy seems to be the highest constraint of using WSN. Because of the application domain, it is often very difficult to replace the power source. Also integrating a recharging system in the nodes will increase the cost of the network that may make the network infeasible to deploy. Even if recharging is added, it is likely to harvest small amounts of energy and conservation may still be needed. Thus the main focus of researchers in WSN is to extend the lifetime of the network by extending the lifetime of each sensor node. Because the energy needed for transmission and reception of data is considered very high compared to the energy of processing data [25], the main goal for saving battery life is to limit the commu-

nication between nodes, thus limiting the number of transmissions and receptions done by each sensor node.

Data reliability is also an important aspect of the sensor networks. In the first place, the wireless medium is far less reliable compared to a wired medium. Again the environment where the sensors are deployed, is often very unreliable so node or link failures become very common. These failures result in loss in the data received by the sink node. For some WSNs, the tolerable loss rate may be very low [4]. For example, in a process control system, where a WSN is used to supply control data to the system, even small loss can cause a disastrous situation. In these situations often we need high reliability in the network. Having multiple paths for the data to reach the sink is considered an important way to ensure reliability against node or link failure. If the nodes in the sensor network have multiple paths to the sink, which are node/link disjoint, then nodes can still deliver their data to the sink using the alternative paths if node/link failure causes one path to fail. Another strategy for reliable transmission is the retransmission of the same data. If data is lost, it can be retransmitted successfully, ensuring reliable data transfer.

Wireless sensor nodes have limited communication range, so the network uses a multi-hop system to send data to the sink where each path from a source node to the sink may have several intermediate nodes forwarding the data. Some practical issues should be handled when the path between the source and the sink increases as the network size grows. The transmission of data from a node to the sink consumes much energy. The nodes closer to the sink have the burden of routing data of many down-link nodes which depletes their energy quickly [53]. This also increases the queuing and processing delay of the intermediate nodes because they have to handle many descendants. The delay to reach the sink also increases which is not preferable for real-time control systems. The increase in path length also increases the probability of conflict among sending nodes, thus there is a higher probability of failure. So the network becomes not scalable. Deploying multiple sinks in the network can be a novel way to solve these problems to make the network more scalable. This has an added advantage that we can improve the reliability of the network by having more than one path to the sinks.

The hot-spot problem is well known in the field of wireless communication especially in WSN [39, 52, 33]. In WSNs, nodes form a data gathering tree where sensor nodes are the nodes of the tree and the sink is the root. The communication load of a node is measured by its number of transmissions and receptions. Nodes with a high communication load will use more energy. Such a node is said to be in a hot-spot. Hot-spot nodes deplete energy fast and are subject to extinction making data stored at them inaccessible and removing their role as forwarding nodes. Generally the nodes closer to the sink have the burden of transferring data of many nodes down the line. This occurs when a node in the data gathering tree has too many children whose data it has to deliver to its parent. This creates an unbalanced energy consumption problem. One way to help balance load is to have a bound on the number of children a node can have. This brings the possibility of using Degree-Constrained Shortest Path Tree (DCSPT) [7] as the routing tree. A DCSPT of a general graph is a tree where the path from each node to the root is shortest, but it also ensures that the degree of each internal node is constrained so that no internal node can have a number of children higher than a given value.

1.3 Thesis Contribution

In this thesis, we have designed logical topologies for wireless sensor networks with multiple sinks. In the network, separate routing trees rooted at the different sinks that span all the nodes are designed. We define the logical topology as the union of these trees. So in the network, for each sink, there is one path from each of the nodes in the logical topology. In this thesis we consider the case of two sinks in the network.

The contribution of the thesis can be summarized as follows:

- A process of creating a logical topology from a wireless sensor network with two sinks. A degree-constrained shortest path tree (DCSPT) rooted at each of the sinks is found using the algorithm proposed in [7]. The DCSPTs are used as the routing trees for the network. The DCSPT ensures that for each node we have a shortest path to the sink which supports lowest energy cost

communication between the node and the sink in a failure free environment. It also ensures that there is a constraint on the number of children each node in the tree can have. In this thesis, we extend the use of the algorithm to the case of multiple (in this case two) sinks. We have used the algorithm to create logical topologies for the network that ensure robustness, path quality and limited number of children for each node. In the logical topology, each node has two edge-disjoint paths, one to each of the two sinks.

- A process of creating a logical topology on a grid network with two sinks. We place the sinks at the opposite corners of a grid of nodes and design a pattern for a logical topology that ensures robustness and path quality. We assume each of the nodes in the grid can communicate with nodes placed in the next grid points horizontally, vertically and diagonally. The logical topology is designed in such a way that all the nodes can communicate with both the sinks using the shortest path. This ensures lowest cost communication between the sinks and the nodes in an error free environment. The number of logical neighbors a node can have in the topology is limited to three and the number of 3-degree nodes is kept small. The topology also supports robustness, because for each of the nodes, the two paths to each of the sinks are link disjoint.

- A static simulation-based evaluation of the proposed logical topologies evaluates the performance with respect to probabilistic robustness and path qualities in the presence of multiple link failures. In the static evaluation, we consider the properties of the logical topologies without the introduction of data traffic.

- Design of scheduling algorithms that tell nodes when to transmit for the two sink environment.

- A dynamic simulation-based evaluation of the proposed topologies using some simple routing schemes with variable number of sources and a dynamic failure model. In the dynamic evaluation, we consider the dynamic behavior

of the network such as variable failure rate, changing link status (*eg.* links going up and down), and dynamic selection of paths to avoid failed links, *etc.*, with respect to time. The performance of the topologies are compared in terms of average time to deliver all packets and network lifetime.

1.4 Thesis Outline

The rest of the thesis is organized as follows. In Chapter 2, we do a background study on the challenges related to robust wireless sensor networks using multiple sinks including placement of multiple sinks in the network, routing with the presence of multiple sinks and optimal scheduling schemes.

In Chapter 3, we design robust topologies in random sensor networks using multiple sinks and degree-constrained shortest path trees. We evaluate the performance of our proposed logical topologies using static analysis with respect to probabilistic robustness and path qualities in the presence of multiple link failures.

In Chapter 4, we design scheduling schemes for routing where data is forwarded to the sink in a two sink network. Finally we perform a detailed dynamic evaluation of the logical topologies using simple routing protocols. We consider multiple sources and dynamic link failures and use simulation to evaluate the performance.

In Chapter 5, we design robust topologies in grid graphs. We design a process of generating robust logical topologies in grid networks having multiple sinks with the presence of diagonal links. We evaluate the performance of our proposed logical topologies using static analysis and dynamic evaluation.

In Chapter 6, we summarize our findings and contributions of our thesis. We also present the scope of future work in this chapter.

Chapter 2

Background Study

Reliability is important to ensure dependability and quality of service to the users of WSN. The wireless medium is far less reliable compared to a wired medium. Some methods of improving reliability are to retransmit lost packets, transmit multiple copies of the same packet, and transmit packets on multiple paths either simultaneously or in reaction to failure. Sending redundant packets is expensive in wireless networks, where transmissions are costly in terms of energy. If the nodes in the sensor network have multiple paths to the sink, which are node/link disjoint, then nodes can still deliver their data to the sink using the alternative paths if the node/link failure causes one path to fail. However the alternative path may not be shortest. Having multiple sinks in a network has several advantages. Nodes in the network can have multiple paths to multiple sinks which increases reliability. Network load can be balanced among the sinks so that one sink does not have to handle a huge number of nodes. The position of sinks in the network and the routing of data from the nodes to the sinks are important factors for reliable data transfer and load balancing purposes. To avoid collisions and reduce the expense of retransmission of collided packets, a scheduling scheme is important.

In this chapter, we present the research work in the current literature about various topics that are related to reliability such as reliable data transfer in WSN, location of multiple sinks in the network, routing with multiple sinks and TDMA scheduling.

2.1 Reliability in Wireless Sensor Networks

In recent years reliability has become an important topic of research in WSN because of the need of ensuring both dependability and quality of service to the users of such networks. Reliability in WSN can be discussed from three different perspectives [48],

- Can the sensors detect all the events? Are there enough sensors present to cover the area? This is termed as *Coverage and Deployment* problem.
- Can the sensors read the data accurately? This gives rise to the *Information Accuracy* problem.
- Can the sensed data be transported to the sinks over multiple hops? This is the instance of *Reliable Data Transport* problem.

In this thesis we are interested only on the reliable data transport problem, where the goal is to ensure that the nodes can deliver the highest amount data to the sinks in the presence of node/link failures. During the communication phase, physical effects like noise, interference, fading are the main cause of data error. Additionally because the nodes may be deployed in a hostile environment, the physical conditions increase the vulnerability of the sensor nodes and their signals [29].

2.1.1 Reliable Data Transfer in WSN

The problem of reliable data transfer over the multi-hop network has several dimensions. First there is a question of the number of packets needed to be delivered. Reliable delivery of a single packet is important for aggregated data. In-node aggregation may be done in the network, where a forwarding node, after receiving all its children's data, combines those with its own data and forwards only one packet to its parent. So the sink receives only one packet from each neighbor. An example of in-node aggregation is the MAX query, where all the sensor nodes report the maximum value of the sensed data to the sink and forwarding nodes will only transmit the maximum value of its data and its children's data to its parent. Reliable

delivery of blocks of packets is necessary for disseminating a new query packet from the sink to the network. Reliable delivery of streams of data is needed for reporting data of periodic queries. In such cases, the sink receives data packets from all nodes or a subset of nodes in each period. Sometimes more than one sensor node senses data of a single event. For reliable delivery of events, it is important to ensure that at least one of the packets generated by these nodes which contains the event data reaches the sink so that the sink is notified about the event. In this thesis, the sinks(s) receive data packets from all the nodes and no aggregation is assumed.

Another important dimension is the percentage of tolerable loss of packets. Some applications do not tolerate any loss, for example, reporting of important events, distributing new queries etc. Other applications have some degree of tolerance, for example, when many sensors deliver correlated data, and some loss is tolerable. In addition, data can travel from sensors-to-sink, sensor-to-sensor and sink-to-sensors. In one to many communication, the sink sends packets to the nodes of the network (*eg.* a query). In many to one communication, nodes of the network send their data to the sink (*eg.* environmental reporting). In one to one communication, one node communicates with another node or the sink of the network. In this thesis, we work on the reliability of the network, where data travels from sensor nodes to the sink (many to one communication).

In the following sections, previous literature on reliability schemes based on retransmission of packets and use of multiple paths are discussed. We also discuss event-based reliability schemes.

2.1.2 Retransmission-based Schemes

Retransmission is one of the main techniques to ensure reliability when loss occurs. The issues related to retransmissions are: a) who will detect the loss b) who will request retransmission and c) who will retransmit. The transmitter uses timers and retransmissions and the receiver uses acknowledgments. Two standard ways of retransmissions are a) Hop-by-Hop MAC layer retransmissions b) End-to-End retransmissions. Karl and Willig [48] stated that, for channels with low bit error rate, end-to-end retransmission works better than hop-by-hop retransmission but with

high error rates end-to-end retransmission consumes too much energy. Given that the wireless medium has high error rates, hop-by-hop retransmission works better than end-to-end retransmission.

In the hop-by-hop reliability approach described in [6], multiple copies of the same packet are sent. The number of copies are estimated from the locally estimated packet error rate, the desired packet delivery probability and hop distance to the sink. In another approach described in [6] called hop-by-hop reliability with acknowledgment, packets are repeated until local acknowledgments have been received. Both schemes work well in the case of independent errors and bursty errors. But they have high resource usage due to duplication of packets. Sending multiple copies at the same time when the error rate is low wastes valuable network resources as more transmission uses more energy.

Block transfers are needed when a large amount of data is transported. The use of Negative Acknowledgments (NACK) in case of block transfer reduces the number of acknowledgment packets. When a packet is received out of order, the receiver can send NACK packets to the sender to request the missing packets. In this case no positive acknowledgment is needed to be sent. If the intermediate nodes cache packets, they can also re-transfer the packet instead of the source. The pump slowly, fetch quickly method [43] delivers a number of packets of one source to a number of nodes. It provides guaranteed delivery for the sequence of data. It has three operations, *pump*, *fetch* and *report*. The main idea of the operation is to distribute data from a source node at relatively slow speed ('pump slowly') but when a node experiences data loss it is allowed to fetch and recover the missing segments quickly from its immediate neighbors ('fetch quickly'). This algorithm is designed to work with one to many communication. Park *et al.*[37] developed a scheme, where the problem of reliable transfer of blocks of data from sink to nodes is addressed. It uses a NACK based scheme and, at the same time, takes extra care for the first packet of a block so that it is reliably delivered to all the sensors. This solves the problem of being able to detect loss of packets by receiving at least one packet from the block. This method is also designed for one to many communication.

2.1.3 Multiple Path-based Schemes

Multiple path schemes are an important way of ensuring reliability. With multiple paths, one approach is to set up the paths in advance and select one of them as the default path. When an error occurs the alternative paths are used. Another approach is to send multiple packets over multiple paths. Approaches using multiple paths require extra route maintenance. Deb *et al.*[11] proposes a scheme where multiple copies of a single packet are sent to randomly chosen paths. Packet duplication can occur in intermediate nodes also. The scheme prefers shorter paths to the sink but otherwise the choice is random. Depending on the error rate, sending multiple packets simultaneously wastes network resources.

Yang and Heinzelman [50] state that multi-path routing can greatly improve the reliability in wireless sensor networks. However, multi-path routing also requires more nodes to be involved in the data delivery, implying more energy consumption and thus a shorter network lifetime. They proposed sleeping multi-path routing, which discovers disjoint multiple paths in a network, selects the minimum number of disjoint paths to meet the reliability requirement, and puts the rest of the nodes in the network to sleep. When the current disjoint paths deplete their energy, sleeping multi-path routing discovers new disjoint paths and selects some of them to support the reliability, until no set of paths can be found to achieve the reliability requirement.

Dulman *et al.*[14] analyze a mechanism that enables the trade-off between the amount of traffic and the reliability. They split the data packet into k sub-packets ($k =$ number of node disjoint paths from source to destination). They show that if only E_k sub-packets ($E_k < k$) are necessary to rebuild the original data packet by adding redundancy to each sub-packet, then the trade-off between traffic and reliability can be controlled. They also show that by splitting the data across multiple paths, the percentage of failed transmissions increases, while the amount of traffic reduces. This gives a way to adjust the reliability while keeping the data traffic low. They calculate the optimal value of E_k to reliably transfer the data packet across the network. If the value of E_k is lower than the optimal, the traffic increases but the percentage of failures decreases.

2.1.4 Event Reliability Schemes

When sensor nodes are deployed to report events, nodes close to the event sense correlated data. Since the neighborhood sensors' data are correlated, some packet losses are acceptable. Akan and Akyildiz [3] argue that reliable event detection at the sink is based on collective information provided by source nodes and not on any individual report, and therefore, conventional end-to-end reliability definitions and solutions are inapplicable and would only lead to a waste of scarce sensor resources. They proposed a scheme named Event to Sink Reliable Transport, which is a transport protocol that tries to achieve reliable event detection with minimum energy expenditure.

Authors in [18, 28] also worked with event reliability. Park *et al.*[18] differentiate the sensor nodes according to their contribution degree (CD) for event detection which depends on distance and relative position of nodes from the event. They propose a quality-based event reliability protocol taking advantage of this CD metric. This protocol consists of two processes, a selection process for selecting sensor nodes to send their reporting data to the sink according to CD and a transport process for differentially transporting them by CD-based buffer management and CD-based load balancing in data congestion. Mahmood and Seah [28] propose the Event Reliability Protocol that enables reliable transfer of packets containing information about an event to the sink while minimizing similar redundant packets from nodes in the vicinity of one another. This protocol is built on the spatial locality condition and employs an implicit acknowledgment (iACK) mechanism. When a sender node detects that its sent packet is lost, it will only retransmit the packet if there is no other packet in its queue from the same region where the event has occurred. The iACK mechanism exploits the broadcast nature of a wireless channel without incurring any additional transmission overhead. The sender, after transmitting the packet, listens to the channel and interprets the forwarding of its sent packet by the next hop node as a receipt of acknowledgment. In a wireless network, iACK mechanism performs better than explicit acknowledgment in terms of reducing the packet overhead, energy efficiency and provides better hop-by-hop reliability.

The problem of these schemes are they only work for event detection mecha-

nisms. So they are designed to work for periodic queries where the sink requires data from all or a subset of nodes in every period and streams of packets flow from the nodes to the sink. In this thesis we apply retransmission over multiple paths to increase reliability of periodic reporting.

2.2 Location of Multiple Sinks

Finding the location of sinks is a well studied problem in the area of WSN. In the thesis we are particularly interested in finding suitable locations of multiple sinks in the network. Finding the location of sinks is similar to the facility location problem, where, given a set of facilities (sinks) and consumers (nodes), the question is where should the facilities be placed and which facility should serve which customer to minimize overall cost (energy consumption, path-length *etc.*) [41]. In addition paths must be found to route the data from the nodes to the sinks.

One way to solve the problem is to formulate the problem as an Integer Linear Programming (ILP) and solve the ILP [32, 41]. Generally solving ILP is NP-hard so heuristic algorithms are used. This method requires some global knowledge such as sensor location, sensor energy level, cost of each link [41]. Another solution is based on Iterative Clustering Algorithms such as K-means. The idea here is to define some initial clusters, place the sinks in the center of those clusters, and then reshape the clusters, so as to allow sensors to choose the nearest sink[41, 31]. This procedure is repeated iteratively until the clusters cannot be reshaped anymore. The main drawback of the approach is again the need for global knowledge. In any case, finding the optimal sink positions with respect to a constraint is NP-hard because it can be converted to a $p - center$ problem [49]. Clustering Algorithms can be designed based on local knowledge but they will only give approximated solutions. When the Euclidean distance is used as the clustering metric, the center of mass of the nodes within a cluster gives the location of the sink nodes. Depending on the priorities of the routing algorithm, power aware distance metrics could also be used [31].

Oyman and Ersoy [31] propose several approaches for the sink location problem

based on different design criteria. They are

- Find the Best Sink Location (BSL): The number of sink nodes is known and the problem is to find an efficient clustering of the nodes.
- Minimize the number of sinks for a predefined minimum operation period: Here a predefined lifetime is given and the number of sinks with efficient clustering has to be determined so that the network has that given lifetime. Here the BSL approach is applied with increasing number of sinks.
- Find the minimum number of sinks while maximizing the network life: This is the approach of extending the lifetime with minimum investment. The initial investment for the sensor nodes should be used for the longest time.

Lin *et al.*[49] propose an optimal solution to find sink positions for small-scale networks and a heuristic approach for large scale ones based on the smallest enclosing circle algorithm that deploys a base station at the geometric center of each cluster. It gives an iterative algorithm for sink position based on the positions of the hot-spot nodes and linear programming is used to compute the optimal routing path.

Vincze *et al.*[41] derive a mathematical model that determines the locations of the sinks minimizing the sensors' average distance from the nearest sink. Using that model, an iterative algorithm is presented that uses global knowledge to position the sinks. The authors then present an iterative algorithm that uses only local information to determine sink position.

Hu *et al.*[42] propose an integer linear programming formulation to maximize network lifetime, prove that it is NP-hard, and introduce a tabu-search algorithm to answer some questions related to hybrid sensor network deployment. They propose that using micro-servers, which have more energy and bandwidth capability then the ordinary nodes, makes a network scalable and cost-effective and their algorithm finds location for optimal micro-server placement.

Hou *et al.*[21] formulate the joint problem of energy provisioning and relay node placement into a mixed-integer nonlinear programming problem and transform it into a linear programming problem. They propose an iterative algorithm

called smart pairing and intelligent disc search which attempts to increase network lifetime by positioning the relay nodes. In the first step, the algorithm uses smart pairing and intelligent disc search to determine possible relay node placements during each iteration so that network lifetime can be increased. This transforms the original mixed-integer nonlinear programming problem into a mixed-integer linear programming problem. Because the mixed-integer linear programming problem is NP-hard, they introduce an equivalence lemma, which shows that if the relay nodes are placed in a smart way, then the mixed-integer linear programming problem can be substituted by a much simpler linear programming problem without any compromise in network lifetime performance and the linear programming problem can be solved in polynomial time.

Pan *et al.*[32] obtain the optimal sink locations for two-tiered WSNs to maximize network lifetime by introducing the concept of virtually stacked planes. In this work, the authors do not assume that every node should have the same initial energy and they do not need to know the bit rate at which data is being generated by the nodes. They analytically derive the upper and lower bounds of maximal topological lifetime by using some properties of WSNs.

Das *et al.*[16] work with the problem of placing a given number of sinks in a given convex region, and to assigning range to each of them such that every point in the region is covered by at least one sink, and the maximum range assigned is minimized. The goal is to cover a region by a given number of equal radius circles where the objective is to minimize the radius. They adopt a clustering algorithm based on Voronoi diagrams.

The target of all these schemes is to minimize path cost, maximize network lifetime or coverage of the network but they do not consider the reliability of the network. In this thesis, work has been done to find location of sinks such that the logical topology is reliable.

2.3 Routing with Multiple Sinks

In this thesis, we consider the multiple sink many to one problem, where the network has multiple sinks and for each sink there should be a routing tree spanning all the sensor nodes. Each sensor node of the network has a link-disjoint path to each of the two sinks.

Path-based algorithms can be used for multi-sink routing. In path-based algorithms the routing path is calculated for the entire topology beforehand and deployed to each node at the start of operation. The routing paths are calculated using different metrics to achieve different routing goals. In the thesis, we are considering path-based algorithms because we will implement TDMA scheduling for the network where paths from each node to the sinks are fixed at the start of operation.

Chen *et al.*[9] propose multi-path routing in wireless sensor networks with multiple sink nodes, which includes topology discovery, cluster maintenance and path switching. Topology discovery [5] is done based on an algorithm where path cost is calculated from distances between nodes, hop-count to sink and the residual energy of sensors. To balance energy consumption, cluster head rotation and path switching is done.

Das and Dutta [10] analytically measure the energy savings for using multiple sinks. The expected energy savings in a d-dimensional sensor region due to a random placement of k sinks and n sensors is proportional to $k^{\frac{1}{d}}$. They also create a logical graph using a virtual sink from the sensor connectivity graph having multiple sinks. In the logical graph, all the sinks are replaced by a single virtual sink and edges connecting a particular sensor node to any sink is replaced by an edge from that node to the logical sink with the new edge weight being minimum distance of the sensor from any of the sinks. This is done so that the existing routing protocols for single sink can be used in multi-sink scenarios.

Dubois-Ferriere *et al.*[13] uses Voronoi scoping where the message will be forwarded to nodes according to the Voronoi cluster. Thus each query is forwarded to the smallest possible number of nodes, and per-node dissemination overhead does not grow with network size or with number of sinks. A neighborhood tracking

method is employed to avoid long-range, high loss links in favor of short-range, reliable links. This could eventually increase path lengths but messages are sent through a more reliable path.

2.4 TDMA Scheduling

A scheduling scheme is necessary to arbitrate the sharing of the broadcasting wireless channel among multiple nodes. The goal of a schedule is to ensure low end-to-end delay and fair sharing. Two popular types of schemes exist: contention based schemes and Time Division Multiple Access (TDMA) schemes.

In contention-based schemes, no resource is allocated to a node a priori. When several neighboring nodes need to transmit some packets, they contend with each other for access to the shared channel and the winning node gains access to the channel and the losing nodes wait for channel access. In many case, no nodes win and this is called a collision. In this case everyone must retransmit thus wasting the energy and delaying all transmissions. This is a random access protocol in the sense that nodes are not guaranteed regular access to the channel. Nodes may also need to transmit control packets like RTS/CTS packets to reserve the channel. Nodes may back-off even when idle to be less aggressive because collisions are so expensive. In contention-based schemes, decisions to transmit are made locally and collisions are locally resolved. Therefore the schemes are flexible enough to work with topological changes. They also have a low access delay in low load situations.

TDMA is a collision free protocol that shares the available bandwidth in the time domain. The channel is divided into time-slots. Nodes are allocated slots in such a way that no two conflicting nodes are allocated the same time-slot. It is assumed that the length of each time-slot is enough to transmit the full message. The number of slots needed to give all the nodes a chance to transmit is called a period. Usually the period repeats itself so if one node is scheduled to send in slot number i in the first period then it will do so in all consecutive periods. Usually each node with n children is given one slot for transmission and n slots for reception but sometimes nodes are assigned multiple slots for transmission.

The TDMA scheme has many advantages compared to the contention-based scheme. For example, it does not waste energy due to control packets like RTS/CTS packet transmission, idle listening or collisions and retransmission [35, 15, 22, 23, 44]. TDMA can also provide end to end delay guarantees and quality of service [12, 22, 23, 44]. It also works better in high load conditions.

Although a TDMA scheme has many advantages, there are a few drawbacks of this scheme. In this scheme, perfect synchronization is required between the sender and the receiver. Guard intervals are introduced between the time-slots to prevent synchronization error but this increases the length of the time-slots and acts as overhead for the system [30]. The TDMA scheme is not flexible enough to work with changing topologies because each change due to node failure requires the set up of the schedule which consumes energy. Again multi-hop communication is needed between the nodes to create TDMA schedule which consumes energy.

The main objective of TDMA scheduling is assigning time-slots to nodes that avoid collision and also minimize the number of slots within a period. This spatial reuse also minimizes end to end packet delay. In a network with TDMA scheduling, a node can be in transmit, receive, idle or sleep state [46] where transmitting and receiving have the highest energy cost. When a node is not receiving or transmitting, it can go to the idle or sleep state to save energy.

Sensor nodes have a certain transmission range that depends on the power of the transmission signal. In the wireless radio medium, communication is broadcasting. So when a node transmits some data, every other node within the range of the transmitting node, gets that data. When a node receives packets from several nodes at the same time, a collision occurs. So, for collision free transmission, the scheduling scheme has to make sure that nodes receive just one packet at a time. Wireless communication is half-duplex and that means a node cannot transmit and receive at the same time. Two types of conflicts are handled in TDMA scheduling: primary and secondary. In primary conflicts, a node sends and receives at the same time or two or more nodes send to the same node at the same time. In secondary conflict, one intended receiver is within range of another transmission which makes it an unintended receiver [15].

Two approaches are usually taken for TDMA scheduling: node scheduling and link scheduling. In node or broadcast scheduling each node in the routing tree is scheduled. Here it is assumed that all the neighbors will receive when a node is transmitting. In link scheduling, each link in the routing tree is scheduled. Link scheduling is better because it allows more concurrency of transmission. If a node is assigned a slot, none of the two-hop neighbors can be assigned that slot in node scheduling which is called the *exposed terminal problem*. But with a careful link scheduling two-hop neighbors can send or receive simultaneously without violating the constraints. In node scheduling, each node can have only one slot per period irrespective of the number of logical neighbors the node might have. In link scheduling, each node can be allocated one slot per logical neighbor so the bandwidth share is proportional to the number of neighbors. Also each neighbor has to listen to a transmission in node scheduling whether it is the intended receiver or not and therefore uses more energy but with link scheduling only the intended receiver listens to a transmission in a slot[17].

TDMA scheduling schemes can also be divided into centralized versus distributed. In a centralized scheme, the sink or any other node collects information from other nodes, such as neighbor information and hop distance from the sink. It builds the schedule using that information and distributes the schedule to the nodes. In distributed scheduling, each node collects its one and two-hop neighbor information and determines its own slot time based on this information. To avoid conflict, a node shares its neighbor information and assigned slot times with its two-hop neighbors. In this thesis, the approach taken is centralized link scheduling. This is reasonable for WSNs because here the sink nodes can collect all the necessary information from the other nodes, compute the schedule and disseminate the result to the nodes.

The approach taken for finding the minimal non-conflicting schedule is as follows. First a conflict graph of the routing tree is built given the physical network. Jain *et al.*[24] first used the model of conflict graph to represent such interference. The conflict graph indicates which groups of links mutually interfere and hence cannot be active simultaneously. For link scheduling, the node of the conflict graph

represents the links to be scheduled in the routing tree and there is a link between two nodes in the conflict graph if scheduling the two corresponding links in the same time-slot will create a conflict. Second, the conflict graph is colored with a minimal number of colors so that none of the adjacent nodes of the graph have the same color. Finding a TDMA schedule of WSN with the minimum number slots (colors) in a period is a NP-Hard problem as this problem can be transformed into the problem of finding the chromatic number of the conflict graph [15, 45]. Therefore heuristics are used for solving this problem.

Gandham *et al.*[17] propose a distributed edge (link) coloring algorithm, that needs at most $\delta + 1$ colors where δ is the maximum degree of the graph and showed that for acyclic topologies at most $2(\delta + 1)$ time-slots are required for conflict free scheduling. Wang *et al.*[45] discuss different kinds of models used to describe interference in WSNs including protocol interference model, RTS/CTS interference model, and physical interference model. They developed both centralized and distributed link scheduling algorithms by using conflict graphs under the RTS/CTS interference model and protocol interference model. By using the conflict graph, they were able to devise a TDMA scheduling algorithm that will work for any type of graph.

Delay occurs if the outbound link is scheduled to transmit before the inbound link. Djukic and Valaee [12] show that the scheduling delay can be interpreted as a cost collected over a cycle on the conflict graph. The authors formulate a min-max program for finding the delay across a set of multiple paths and prove it is NP-Hard. The authors design heuristics to select appropriate transmission orders. Once the transmission orders are known, a modified Bellman-Ford algorithm is used to find the schedules.

Wang *et al.*[44] took a different approach by implementing genetic algorithms in the scheduling problem. It combines the high search efficiency and a global search ability of particle swarm optimization with good local search ability of simulated annealing.

The state transition, *eg.* from the sleep state to the active state, should be considered for an energy efficient TDMA sleep scheduling in WSNs because state tran-

sitions require energy. Ma *et al.*[22] propose an interference-free TDMA sleep scheduling problem called contiguous link scheduling, which assigns links of a particular sensor with consecutive time-slots to reduce the frequency of state transitions. The authors also propose centralized and distributed algorithms to schedule nodes considering this problem that uses time-slots at most a constant factor of the number of time-slots used in the optimum TDMA scheduling.

Zhang *et al.*[54] uses a genetic algorithm to obtain a graph coloring strategy to color links of the network which would balance energy saving and end to end delay.

Most of the previous scheduling algorithms work on the assumption that there are many independent point to point flows in the network. But in sensor networks, a data gathering tree is generally used where the nodes need to forward their data to the root of the tree which is the sink. Ergen and Varaiya [15] propose a TDMA scheduling algorithm that takes into consideration the many-to-one nature of communication and the interference model of the wireless medium. They propose two centralized and one token-based distributed scheduling algorithm. The first centralized scheduling is link scheduling using conflict graph coloring and the second centralized scheduling is based on coloring a linear graph where each node represents a level based on hop count in the original network. These algorithms work only with logical topologies having one sink. In this thesis the degree heuristic, described by Ergen and Varaiya [15] is applied in the algorithm for coloring the conflict graph. In this heuristic, the nodes are sorted in descending order of node degree and the node with the highest degree is colored first. Coloring the most constrained node at every stage means a high number of nodes are free to be colored in later stages.

2.5 Chapter Summary

In this chapter, we have seen that reliability is an important topic of research in WSN because of the need of ensuring both dependability and quality of service to the users of such networks. We have described research work that designs algorithms to reliably transfer data from sensor nodes to sinks over multiple hops.

We have seen that by using multiple paths to sinks and retransmissions, data reliability can be increased significantly. Next we have discussed the problem of finding suitable locations for multiple sinks in the network that fulfill some objectives. Previous research work has shown that this problem is NP-Hard and we describe various heuristics used to solve this problem. Next we discussed the problem of routing in the presence of multiple sinks using gradient-based and path-based algorithms. Finally, we discuss the problem of finding minimal a TDMA schedule for a WSN. Previous research has also shown that it is a NP-Hard problem and we have described various heuristics used to find the solution to this problem.

In the next chapter, we design robust topologies in random sensor networks using multiple sinks and degree-constrained shortest path trees. We evaluate the performance of our proposed logical topologies using static analysis with respect to probabilistic robustness and path qualities in the presence of multiple link failures.

Chapter 3

Design of Robust Topologies in Random Sensor Network Graphs

In this chapter we design robust topologies for WSN with two sinks. We are concerned with sensor networks where sensor nodes periodically report data to a sink. The robustness is ensured by having for each sensor node other than the sinks, two link-disjoint paths, each one to a different sink. In addition, the design finds the shortest path to the corresponding sink from the nodes, and there is a limit on the number of children each node can have. The shortest path will ensure that, in an error free environment, communication between the nodes and the sinks can be done with reduced energy consumption and delay. We hope to reduce hot-spot problem by limiting the number of children. The design is done in such a way that, given a physical topology, the logical topology includes all the nodes and the topology has the following properties as described by Reza [36],

- **Robustness:** In the logical topology there should be redundant links so that the maximum possible nodes are connected if one or more links fail. We only consider the case of link failures for this thesis. The design is done in such a way that two types of robustness can be maintained. *Deterministic Robustness* is maintained through having two link-disjoint paths to each of the sinks. This ensures that if one link is down in a path from any node to a sink, then there will be another path from this node to the other sink. Again *Probabilistic Robustness* increases the percentage of nodes that are connected in the event of multiple link failure.

- **Path Quality:** In the logical topology, the path length from each of the nodes to the closest sink should be minimized. This decreases delay and number of transmissions. This ensures reduced energy consumption in the case of an error free environment. The length of path from the nodes to the alternative sink should also be minimized.

- **Number of Children:** In the topology, there should be a constraint on the number of children each node can have. This is done to reduce the hot-spot problem.

In this chapter we consider the design of a logical topology and evaluate its properties from the given physical topology. We also evaluate the robustness and path quality of our proposed topologies by simulating them using failure models. Network operation schemes for selecting paths, detecting faults and choosing alternative paths are considered in later chapters where we perform dynamic evaluation of our deployment topologies.

In Section 3.1, we give an overview of the DCSPT algorithm. In Section 3.2, we describe the properties of robust logical topologies in random sensor network graphs. In Section 3.3, we design sink placement policies for the two sinks in the network to obtain robust logical topologies. In Section 3.4, we evaluate the performance of our proposed logical topologies using static analysis with respect to probabilistic robustness and path qualities with the presence of multiple link failures.

3.1 DCSPT Algorithm

The basic mechanism that is used in this thesis is Degree-Constrained Shortest Path Tree (DCSPT). The DCSPT of a graph is a tree, where for each node, the path to the root of the tree is shortest and there is a constraint on the number of children each internal node of the tree can have. Suppose we are given an undirected graph $G = (V, E)$, a root $v_r \in V$ and a bound d, where V is the set of vertices (nodes) and E is the set of edges. A shortest path tree is a tree with root v_r where paths from all non-root vertices to v_r are shortest. The Degree-Constrained Shortest Path tree is a

shortest path tree, where the degree of each vertex including the root is equal to or less than bound d.

The present algorithm used in this thesis was originally developed Bai *et al.*[7]. The algorithm can find a degree-constrained shortest path tree in polynomial time. The algorithm takes as input an unweighted graph and a degree bound d. It outputs all possible d-degree-constrained shortest path trees. Considering each of the nodes of the input graph at a time, the algorithm first constructs an unweighted shortest path graph rooted at the node. Then it uses the maximum flow algorithm to solve an off-line bipartite matching problem over a bipartite graph derived from the shortest path graph. This produces a degree-constrained shortest path tree rooted at the node or returns false if no such tree is possible. This whole process is repeated considering each node in the graph as the root. In this thesis, we use the degree-constrained shortest path trees produced by the algorithm and apply it in a multiple sink scenario where the roots of the DCSPTs work as sinks of the network and more than one DCSPT are used to produce a reliable logical topology for the network. We consider cases where the two sinks have the same degree constraint as the other nodes and cases where it is different (*eg.* larger).

3.2 Robust Topologies in Random Sensor Network Graphs

In this section we define the method of designing robust topologies. In this thesis, we assume that there are n nodes in the network and these nodes are scattered randomly on grid points in a grid of arbitrary size. We have chosen to use grid-based random topologies instead of pure random topologies because we can control the density of physical topology in this method. Having control on the density is important because the probability of obtaining a DCSPT out of a network depends on the density of the network. With a highly dense network it is very hard to get a SPT with high constraint on the degree. In the network graphs there is at most one link between two nodes. The distance between grid points is considered as $u = 1$ unit and the transmission range of each sensor nodes is $r = 6$ units. In this thesis

we use the unit disk model, where a node can communicate with another node if the other node is in the range of the first node. We assume that in the network graph there is a link between two nodes that can communicates with each other. The physical topology is defined as the nodes and their links based on the transmission range. Here, we consider physical topologies that are 2 link-connected. That is, the graph that represents the physical topology, remains connected if a single link is removed from the graph. This property is helpful to find a robust logical topology. We assume that the roots of the DCSPTs work as the sinks in the network. We choose the two sinks such that in the union of the two DCSPTs, for each node, its path to one sink is link disjoint from its path to the other sink. We call this logical topology 2 link-disjoint sink-connected. An *eligible* sink pair is two sinks of DCSPTs that admits a 2 link-disjoint sink-connected topology.

In this thesis, we chose heuristics to find sink positions and we analyze the logical topology resulting from them to compare their benefits. All these heuristics are centralized in the sense that a central control point is assumed that knows the physical topology and that can calculate and then distribute the logical topology to the other nodes.

We initially assumed that the node positions were the possible locations for the sinks. The problem with this scheme is, the two sinks are dependent on each other. That is a sink is a node (not always a leaf) in the other sink's tree. Therefore, it is difficult to obtain logical topologies that are 2 link-disjoint sink-connected. The algorithms and experimental results of this scheme is presented in Appendix A.

From the initial experiments of choosing eligible sink pairs from ordinary nodes in the graphs; we have seen that reducing dependency gives us better results. Since the original nodes are positioned randomly among grid points separated by one unit distance, sink position are also chosen from grid positions where there is no node. First one possible sink position is chosen in the graph, links joining it to the rest of the graph are added based on the transmission range of the sink. The DCSPT algorithm is run on the resultant graph and a tree rooted at the sink, if it exists is saved. Next another sink position is chosen and its links are added to the original graph. The DCSPT algorithm is run again ignoring the first sink and a resultant tree

rooted at this sink is saved. Then we take the union of the two trees to obtain the logical topology. As only 30 nodes are placed on a 25×25 grid, there are many empty spots left for the sink.

Several experiments have been done to find a good heuristic for choosing sink positions in this method that has a better chance to produce link-disjoint sink-connected logical topologies. All the experiments are done with 50 random topologies. Each topology is a 25×25 grid, with 30 nodes placed randomly on the grid points. The DCSPT algorithm is run on the topologies using node degree constraint (NDC) 4 and 5 and sink degree constraint 5. All the nodes have a range of 6 and sinks are also assumed to have range 6. The range is chosen in such a way so that the density of the network is not very high. If the density of the network is very high than it is difficult to produce a DCSPT from a physical topology. The node degree constraints are chosen 4 and 5 because if we chose a constraint smaller than 4, it is difficult to obtain a DCSPT and if we have a constraint larger than 5 than the obtained DCSPT are all the same.

The metrics used for comparing the various schemes are average Hop-Count, average Alternative hop-count. Hop-count is the measure of distance from each node to its closest sink. This is averaged over all nodes. The Alternative hop-count is the measure of distance to the farthest sink and it is also averaged over all nodes. These are the measures of path quality. We want to compute these values because it is better to have a logical topology that decreases these values which would reduce transmission cost and delays.

3.2.1 Experimental Results

From the results found in Appendix A, it is found that, the possibility of finding an eligible sink pair location is higher if the sinks are positioned far apart. This would seem to be a good heuristic since nodes near one sink are more likely to be leaves with respect to the farthest sink and therefore the probability of sharing links in the two paths to the sinks is less.

We run an experiment by fixing the choice of location for the first sink to one edge of the network and then searching for eligible sink positions in the whole grid.

For Sink 1, we have tried all locations from column 1 to 5 of the grid and for Sink 2, we have considered the whole grid. We found that, the probability of getting an eligible sink pair is higher if we chose the second sink from the last 10 columns of the grid. This motivates us to try a sink location heuristic, called *Edge-position*. where the location for Sink 1 is chosen from the first 5 columns of the grid, and Sink 2 is chosen from the last 5 columns of the grid.

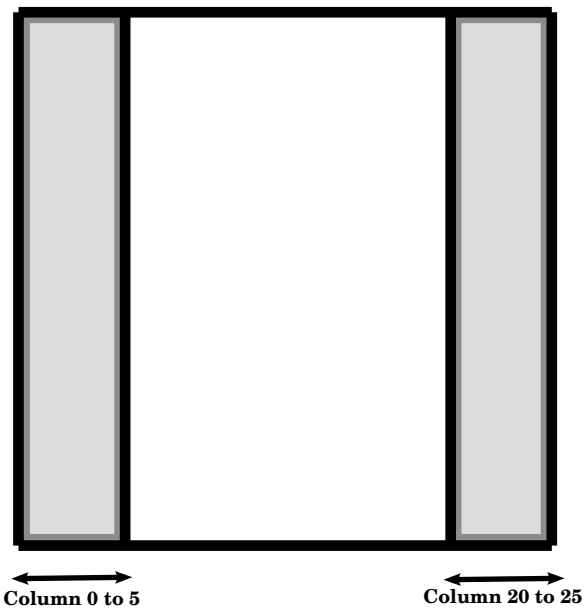


Figure 3.1: Search space for Edge-position scheme

Experiments with Unlimited Sink Degree

Constraining the sink may not be realistic. The purpose of having a constraint on the number of children is to mitigate the hot-spot problem thus extending the life time of the network. Sinks are assumed to have unlimited power compared to the ordinary nodes so having an unconstrained number of children may not be a big problem, but it may create better trees out of the physical topology. The trade off of a sink having more children is schedule length. The number of time slots needed for the sink to gather all its children data is proportional to the number of children it has, so increasing the number of children may increase the TDMA schedule length which in turn may increase the delay of delivering a message to the sink.

The next set of experiments is done with sink degree constraint 15. Experiments

done with Edge-position scheme show that the possibility of obtaining eligible sink pairs increases slightly for NDC 4 with sinks with limited degree constraint, but the probability does not change for higher node degree constraints. The results are shown in Table 3.1.

Node Degree Constraint	# Eligible Pair (Sink degree constraint 5)	# Eligible Pair (Sink degree constraint 15)
4	36	38
5	45	45
6	45	45

Table 3.1: Number of eligible pairs (out of 50) for varying node degree (sink degree constraint = 50)

We measured the sink degree for both the successful and unsuccessful cases. In most of the cases the degree is 3 or 4. Sometimes it is higher than 5 when there are many neighbors for the sink and they have to connect to the sink directly in the shortest path tree. Therefore, increasing the sink degree constraint is not much beneficial unless the sink has many neighbors.

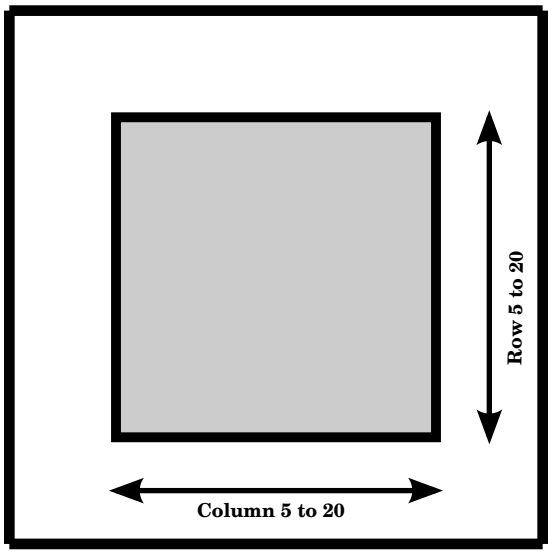


Figure 3.2: Search space for Middle-position scheme

Exhaustive Search on the Grid

The next experiments are done to find the locations in the grid where the probability of finding an eligible sink pair is higher. Experiments are done with Sink and Node Degree Constraint of 5 and number of nodes 30 with 50 random topologies. Figure 3.3 shows the result of the exhaustive search for eligible sink positions for one of the 50 topologies. Each point represents the grid points where we can put a sink to get an eligible sink pair. In this experiment we have put the first sink in the left half of the grid, that is, in every empty grid positions from column 1 to 15 in a 25×25 grid. and recorded all the grid positions where we can put the second sink to obtain an eligible sink pair. The location of the nodes is shown as well. The result is symmetric if we position the first sink on the right half of the grid.

From the figure, we can observe that there is a large gap in the middle of the grid where no good location pair can be found. If we put the first sink in the empty grid positions in this gap, there is no empty grid position to put the second sink corresponding to this position which will produce a 2 link-disjoint sink-connected logical topology. This is because positioning sinks in the middle creates lots of neighbors for them which decreases the probability of finding the DCSPT. Also, positioning the sinks close to each other decreases the probability of getting 2 link-disjoint sink-connected graphs.

Search for Sink Positions in the Middle of the Grid

Next an experiment is set up to find eligible positions in the middle of the network so that the two sinks can be close together. We have tried this experiment because positioning sinks in the middle will decrease average path lengths of the nodes to the sinks. For Sink 1, a search is done from column 5 to 20 and row 5 to 20 in the grid, and for Sink 2, a search is done from column 20 to 5 and row 5 to 20 in the grid. The experiment is done with two cases. In the first case, Middle, the search stops as soon as the first eligible sink pair is found. In the second case, Middle-best, the search continues to find the best sink pair to obtain the minimum hop-count distance from the nodes. In Table 3.2, it can be seen that both the hop-count and the alternative hop-count decreases with the Middle-best scheme. Both the Middle-

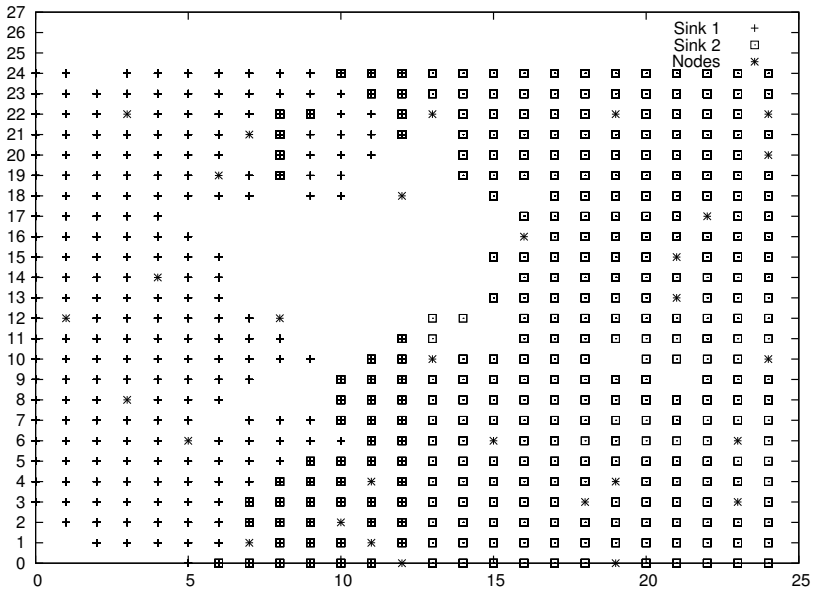


Figure 3.3: Locations of eligible sink pairs

position schemes have better trees compare to the Edge-position scheme in terms of path quality. However, the probability of getting eligible sink pair is lower for the Middle-position schemes. And, it takes longer to search for eligible pair in the Middle-position schemes.

	Count(Out of 50)	Hop-Count		Alternative Hop-Count	
		Average	CI(95%)	Average	CI(95%)
Middle	39	2.209	0.0001	5.123	0.005
Middle-best	39	1.796	0.001	4.67	0.005
Edge-position	45	2.626	0.07	5.88	0.15

Table 3.2: Comparison between different sink position schemes

3.3 Static Analysis of the Logical Topologies

In the previous sections we have designed robust topologies for random wireless sensor networks. In this section, we evaluate the probabilistic robustness of these logical topologies. In the static evaluation, we consider the properties of the logical topologies without the introduction of data traffic. Here we do not consider the

dynamic aspects of the network such as dynamic selection of routing paths and network operations like routing, fault detection, *etc* that change with time. Dynamic evaluation of the topologies are described in a later chapter. For the evaluation, we have considered only the topologies for which we can find eligible sink pairs so the topologies are 2 link-disjoint sink-connected. The metric used for the evaluation are:

- **Quality of Path**: Quality of path is measured by *Path Length* and *Alternative Path Length* on the logical topologies. Path length means the average path length by hop-count to the closest sink and the alternative path length means the average path length by hop-count to the alternative sink. The shorter the path lengths the better the routing tree becomes because it reduces the delay to transfer a message from a node to the sink.

- **Sink Load**: In a scenario where nodes report to their nearest sink, the Sink Load defines the number of nodes reporting to each sink. If the logical topology is well-balanced then each of the two sinks will have a similar percentage of load.

- **Histogram of the Degrees of Each Node**: Here we show the degree of each node on the logical topology.

- **Effect of Failure**: Here, the quality of the topology is determined in the presence of link failure. For this, multi-link failure is considered. To assess the quality of the topology in the presence of failure, *Degree of Robustness* and *Path Quality* are measured. Degree of Robustness is defined as the percentage of nodes disconnected due to link failure. A node is disconnected if it is unable to reach any of the two sinks. The lower the percentage of nodes disconnected, the more robust the topology is against link failure. Path Quality is defined as the total path length from all the nodes to their nearest reachable sink. This length is averaged by number of nodes in the network. It indicates the quality of available paths in case of link failures. We do not consider the disconnected nodes that cannot reach any of the two sinks in this calculation. When there is no failure, the path quality is the average path length.

3.3.1 Experimental Setup

The analysis is done with 30 nodes placed in a grid of size 25×25 randomly with node degree constraint 5 and sink degree 15. The range of each node is $r = 6$. For locating the sink position, we have used the Edge-position and the Middle-best scheme. Each experiment is done with 50 different physical topologies and their average is taken and 95% confidence intervals are calculated. In the experiments with link failures, multiple link failure is considered where, with a certain probability, each link in the logical topology is removed.

3.3.2 Experimental Results

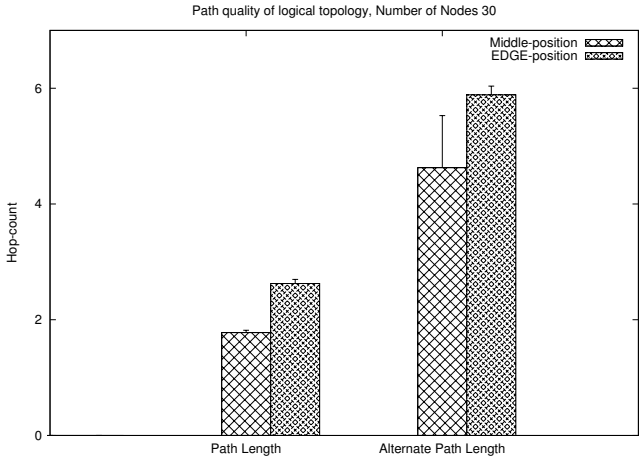


Figure 3.4: Path quality of the logical topology

In the first experiment, we calculate the average path length and average alternative path length for the logical topologies. This is important because the shorter the lengths are, the less delay a message would face to reach the sinks from a node. From the experiment, it is found that the Middle-position scheme has better path quality because it produces shorter path lengths and alternative path lengths. This behavior is expected since, with the sinks positioned at the edge, the distance from the nodes is higher in the Edge-position scheme as seen in Figure 3.4. The next experiment measures the load of each of the two sinks in the network. From Figure 3.5 it can be seen that, with Middle-position scheme, the load is more balanced between the two sinks than the Edge-position scheme. This is because, in the Middle-

position scheme, the sinks are positioned in the middle of the network and almost an equal number of nodes report to the each of the sinks. In the Edge-position scheme, the first sink is positioned more on the edge of the network and the second sink is positioned more on the inner side, thus the second sink has potentially more one-hop neighbors and more nodes report to the second sink. The degree histogram of the nodes in Figure 3.6 shows that most of the nodes in the logical topology have degree of two and only a few have degree greater than 5, and these nodes are the sinks which are not as constrained as the sensor nodes.

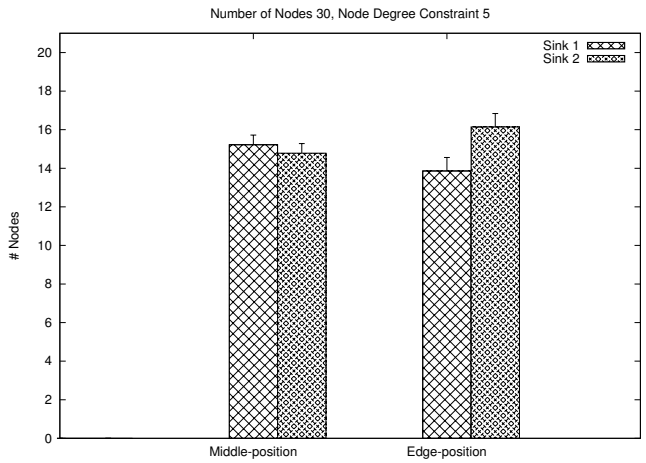


Figure 3.5: Sink load

Figure 3.7 presents the robustness of the topology to multi-link failure. In multi-link failure cases, the number of disconnected nodes increases as the probability of link failure (p_i) increases. The topologies are robust to multi-link failure because only around 10% of the nodes are disconnected from the sinks even with a high p_i of 30%. But, as p_i increases further, there is almost an exponential increase in the number of disconnected nodes. From the experiments it can also be seen that the Middle-position scheme has a higher degree of robustness compared to the Edge-position scheme as seen in Figure 3.7. This is because the sinks are positioned in the middle and therefore, they have more neighbors and the nodes have a better chance of being connected to one of the sinks.

Figure 3.8 shows the average path length of the connected nodes from the closest sink in the presence of multi-link failure. It can be seen that path length increases

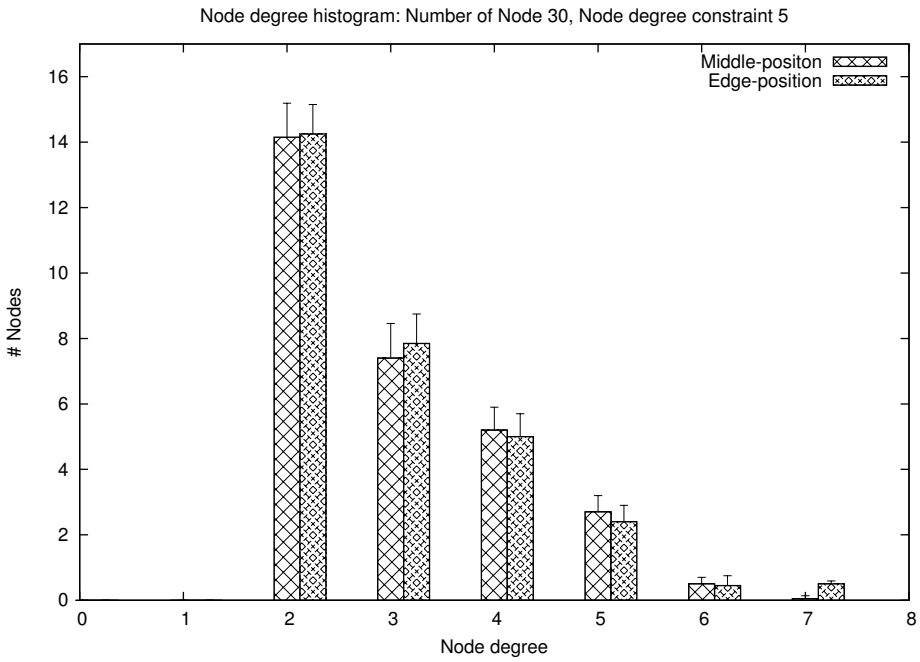


Figure 3.6: Degree histogram

initially as p_i increases since the longer alternative paths are taken but then the average path length decreases as p_i increases further. This occurs because, as p_i increases, more nodes become disconnected and nodes close to the sinks are more likely to be the ones that are connected. The path lengths of the disconnected nodes are not calculated so the average path length decreases. From the experiments it is found that the topology can handle around 30% percent of link failures with alternative paths but it fails to perform under higher link failures. It can be seen that, with the Middle-position scheme, the average path length decreases more slowly than the Edge-position scheme. This is because, as the sinks are positioned in the middle in the former case, they have more neighbors and thus nodes have better chance of being connected to one of the sinks.

3.4 Chapter Summary

In this chapter, we have designed robust topologies with random graphs for wireless sensor networks. The network consists of two sinks. Robustness is ensured because for each node, there are two link-disjoint paths, one to each sink. The topologies

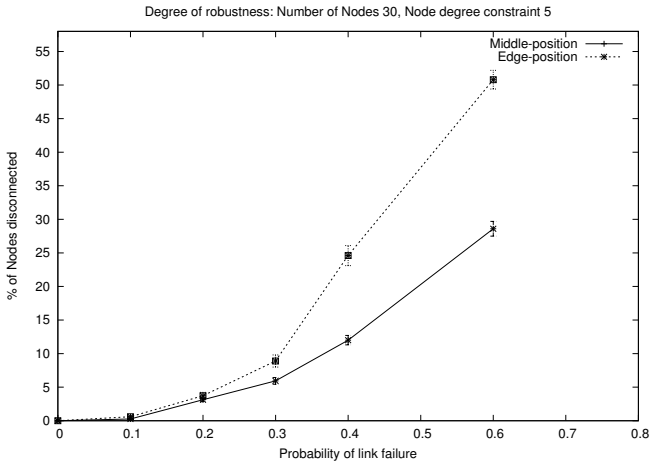


Figure 3.7: Degree of robustness of union graph in multi-link failure

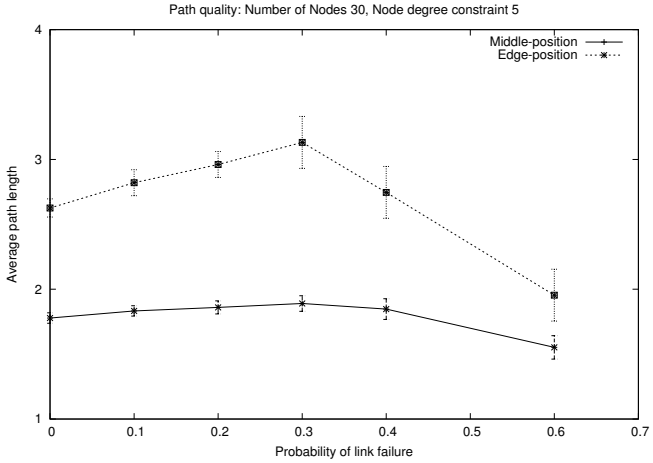


Figure 3.8: Path quality in multi-link failure

are such that, for each sink, the routing tree rooted at that sink, spanning all the other nodes is a shortest hop based path tree and the tree is also degree-constrained so that there is a limit on the number of children each node can have in the tree. We have formulated sink location strategies so that the logical topologies built using two DCSPTs, can maintain the link-disjoint path property. We have shown that if the two sinks are positioned on the opposite edges of the grid, the probability of obtaining an eligible sink pair is higher. To examine the quality of the designed topologies, we have done static analysis on them using failure models. Results show that the topologies maintain a good degree of robustness and the topologies can perform well under moderate link failure rate. It is also found that, the Middle-position scheme is more robust than the Edge-position scheme to link failures.

Chapter 4

Dynamic Analysis

In this chapter the dynamic behavior of the 2 link-disjoint sink-connected logical topologies that are designed in Chapter 3 are evaluated. The dynamic behavior includes changing link status (links going up and down) and the dynamic selection of paths to avoid failed links. A scheduling scheme is necessary to share the broadcasting wireless channel among multiple nodes in order to control end to end delay and provide fair sharing. Because a Time Division Multiple Access (TDMA) scheme is used, there is no contention among the nodes for channel access. Therefore the focus is on the Network Layer (NL) which is responsible for the routing of the packets towards the destination.

Routing is required to deliver data to a sink in the presence of link failure. Because of the unreliable characteristics of the wireless medium, nodes/links sometimes fail. The current work deals with the case of link failure where interference causes nodes to temporarily not be able to communicate with neighbors. If the link between the sender and the receiver fails when the slot has arrived for the sender, the sender cannot send the data to its designated receiver and the routing protocol should use the alternative path to deliver the packet to the other sink. Because the logical topologies are 2 link-disjoint sink-connected, the alternative path to the other sink is link disjoint. In routing, unnecessary delay should be avoided so that quality of service can be maintained. Performance of the logical topologies is evaluated with two routing protocols in terms of resiliency to link failures, average hop-count and network lifetime.

In Section 4.1 we design TDMA scheduling algorithms for logical topologies

with two sinks. In Section 4.2, we describe the routing protocols used in the dynamic analysis. In Section 4.3, we describe how the experiments are set up for dynamic evaluation including the frame generation and link failure model. In Section 4.4, we describe the simulation metrics used. In Section 4.5, all the assumptions, parameter settings and design of experiments are presented. In Section 4.6, we present the results from the dynamic evaluation.

4.1 TDMA Scheduling

TDMA scheduling is used in this thesis. The motivation for this choice of medium access can be found in Section 2.4. The strategy to find the schedule consists of creating a conflict graph and then coloring the graph. In TDMA scheduling, a period is a set of consecutive slots and the nodes in the network are allocated slots in the period for conflict free transmission. The objective of the scheduling algorithm is to find a schedule that decreases the length of a period. The nodes of the conflict graph are the links of the logical topology and the links of the conflict graph indicate interference between links in the original topology. A coloring algorithm will assign colors to the nodes of the conflict graph, so that no adjacent nodes can have a common color. Therefore, this assigns colors to the links of the logical topology. Links of the logical topology with the same color can be assigned the same slot since they will not interfere. A schedule can be created by assigning a slot to each color.

In the next sub-section we discuss different ways to build conflict graphs in two-sink networks.

4.1.1 Conflict Graphs for a Single Routing Tree

In this section, we discuss the method of building a conflict graph for a single routing tree which represents the logical topology.

The physical network is represented by the graph $G = (V, E)$, where V is the set of nodes including the sink node and E is the set of links. In the wireless radio model, an edge exists between two nodes in the physical network if one is within

the transmission range of the other. The routing tree rooted at the sink is represented by the graph $G_T = (V, E')$, where $E' \subseteq E$ is the edge set that represents the logical connection between the nodes. $neighbor(u)$ represents the neighbor set of node u in graph G.

We define the conflict graph as $G_C = (V', E'')$. In link scheduling, all the edges in the logical topology represent the vertices in the conflict graph. So the vertex set V' represents the links E'. In the conflict graph, the edges represent the conflicts. An edge in E'' represents two edges in the logical topology that cannot be active at the same time.

Let us assume that node u of the network is sending and node v is receiving. For conflict free transmission, u cannot receive or v cannot send as node cannot send and receive at the same time. v cannot receive from nodes other than u, as nodes can only receive from one sender at a time. The algorithm for constructing the conflict graph from a single routing tree is listed in Appendix B.

The conflict graph built in this way considering only one tree is called the ***Full-TreeScheduling***. In this method, there is only one sink in the network and nodes have only one path to the sink.

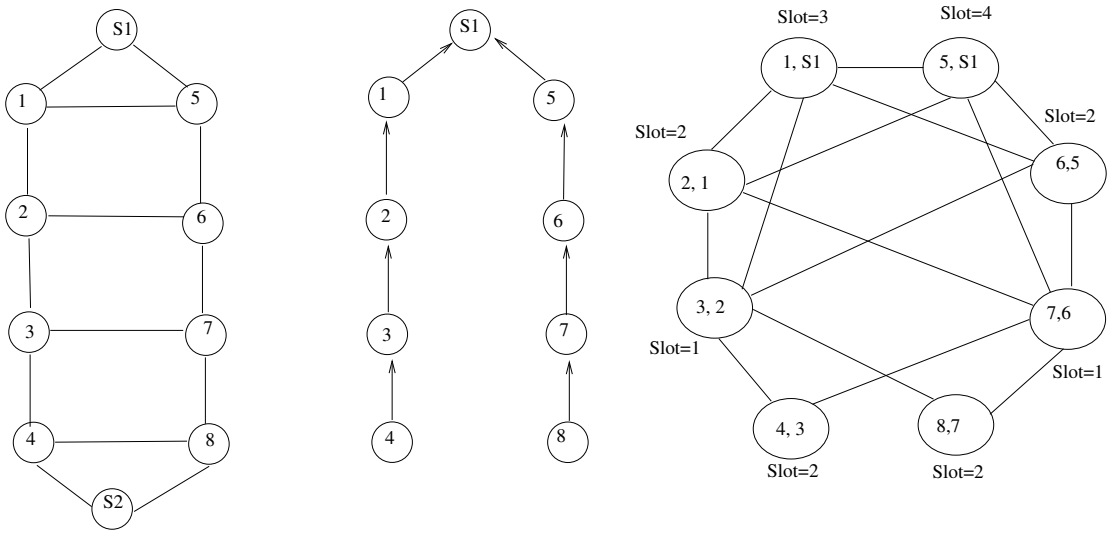


Figure 4.1: Full tree scheduling. The physical topology, the logical topology, the conflict graph

After the conflict graph has been created, the conflict graph is colored to determine the TDMA schedule. The coloring algorithm, which is listed in Appendix B,

is proposed by Ergen and Variya [15]. In this algorithm, the nodes of the conflict graph are sorted based on their degree in descending order. Then each node is colored with the minimal number of colors, such that no two neighbors have the same color. After finding the color, the number of colors used is the schedule length and different colors represent different slots in the TDMA period.

In Figure 4.1, an example of FullTreeScheduling is given. Here, given a physical network with its logical topology, the conflict graph is presented with a coloring. In the figure, $Slot$ represents the color assigned to it.

4.1.2 Algorithms for Building Conflict Graphs for Multiple Trees

The TDMA scheduling problem discussed so far works only with one routing tree. In this thesis, effort has been made to design a two-sink TDMA schedule. In the logical topology, each node has two distinct parents (one for a route to each sink) because the topology is 2 link-disjoint sink-connected. The difference between the approaches is how they transmit to their parents. The different routing strategies are:

- Nodes send data only to the closest sink. Although there are two sinks in the network, nodes will only send to the closest sink. No alternative paths are used. Each node sends to only one parent.

- Nodes send data to both of the parents alternatively. In this strategy, nodes can send to both of the sinks but not simultaneously. Every node has two link disjoint paths to the sinks. A node generally sends to its closest sink but can chose to send to the other sink when the path to the closest sink fails.

- Nodes send data to both of the parents simultaneously. Because wireless communication is broadcasting in nature, nodes can simultaneously send to both parents. This will reduce the length of schedule but increase the number of redundant packets if both parents forward the data.

The different approaches result in different conflict graphs. In these approaches we assume that for each node u, there are two parents $v1$ and $v2$ corresponding to

tree1 and tree2 respectively. Here $v1 \neq v2$ as the logical topology is 2 link-disjoint sink-connected.

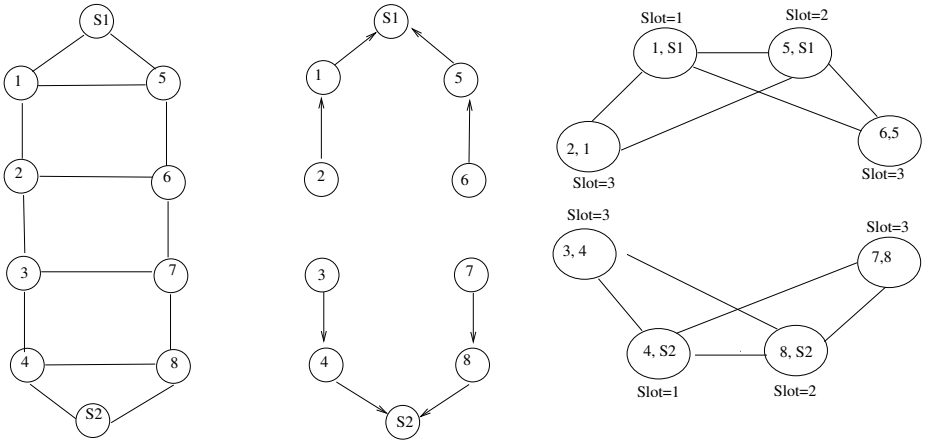


Figure 4.2: Pruned tree scheduling. Physical topology, logical topology, conflict graph

In the first approach, called *PrunedTreeScheduling*, the nodes will only try to send to the closest sink and there is no alternative path for the nodes to send the data. In this approach, the two trees are pruned in such a way that nodes will only send to its closest sink and therefore to one parent. The link to the other parent is pruned away. The TDMA scheduling algorithm is run on both the pruned trees together. The algorithm for *PrunedTreeScheduling* is given in Appendix B. In Figure 4.2 an example of PrunedTreeScheduling is shown. Here we have a physical topology with the two routing trees and the conflict graph. The assigned slots are also mentioned in the figure.

In the second approach, called *SingleTreeScheduling*, the nodes can send to both sinks in the network. In this approach nodes can send data to two sinks alternatively. In this approach the TDMA schedules for two trees are determined one after another using the approach for the single tree considering the two trees separately. In this approach, we first build a conflict graph using the first tree, considering link to the first parent as the node of the conflict graph and color the graph. Next we build another conflict graph using the second tree, considering link to the second parent as the node of the conflict graph and color the graph with a new set of colors. The disadvantage of this scheme is the schedule length is very high as no parallel

transmission between the links of the two trees are possible in this approach. In this approach, nodes can choose to send message to one or both of the parents within on scheduling period.

In the third approach, called *TwoParentScheduling*, nodes can send to both of the parents simultaneously. This approach also considers both the trees together where parallel transmissions in both the trees are possible. In this approach, the conflict graph nodes do not represent the links of the routing trees. Instead, a conflict graph node represents the two links that should be activated together when a node tries to send to both of its parents simultaneously. So in the conflict graph nodes represent links to both the parents together. In this approach the alternative paths can be activated in the same period as the primary path. The algorithm is listed in Appendix B. In Figure 4.3 an example of TwoParentScheduling is shown. Here we have a physical topology with the two routing trees and the conflict graph. The assigned slots are also mentioned in the figure.

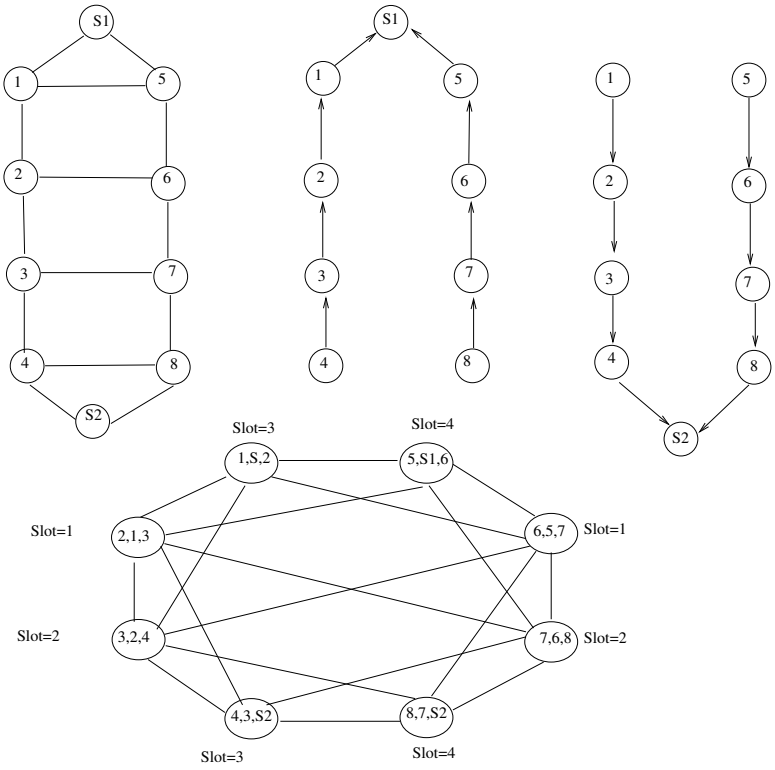


Figure 4.3: Two parent scheduling. Physical topology, logical topology, conflict graph

In the fourth approach, called *OneParentScheduling*, nodes can send to both of the parents alternatively. Here we consider both the trees together where parallel transmission in both the trees is possible. Here for each node u, we have two links to both of the parents while constructing the conflict graph. The algorithm for this approach is same as algorithm for the PrunedTreeScheduling, where the difference is with the input. Instead of giving the pruned routing trees, this algorithm takes as input the original routing trees. Here alternative paths can be used but may not be within the same period as the primary path.

4.1.3 Experiments to Compare the Scheduling Schemes

We have applied these algorithms to 50 eligible logical topologies for random sensor network graphs to compare the resultant schedule length. The experiments are done in a network with 30 nodes with node degree constraint 5. Links do not fail in these experiments. All the data points show 95% confidence intervals. From the experiments, shown in Figure 4.4 we have found that allowing parallel transmission (OneParentScheduling & TwoParentScheduling) between two trees can reduce the schedule length considerably compared to the SingleTreeScheduling approach. The gain is about $30 - 40$ percent. The TwoParentScheduling produces a shorter schedule than OneParentScheduling because we utilize the broadcasting nature of the wireless transmission. There is a trade-off in sending to both of the parents together. On one side, we are trying to utilize the broadcasting nature of the wireless medium and, on the other hand, by sending to two parents simultaneously, we are introducing more conflicts. From the experimental results it has been found that the gain is significant and therefore it is of interest to investigate the TwoParentScheduling algorithm. The schedule length for PrunedTreeScheduling is lower than all other approaches. This can be explained easily if we examine the nature of the trees scheduled together. Because the trees are pruned, the number of links have decreased so there is less interference and therefore fewer links in the conflict graph. However, the disadvantage of this scheme is that there are no alternative paths for the nodes so the reliability of the network will be lower. The TwoParentScheduling scheme may explicitly cause redundant packets that may improve

reliability but reduce throughput. The effect of reliability will be investigated in the next section. Now as the OneParentScheduling has a higher scheduling length and the results from the experiments also show that it does not produce better results than TwoParentSchedling, the results using this scheme are excluded in later sections. Also we have not used SingleTreeScheduling because of the large schedule lengths which would have resulted in high delays.

Next we evaluate the impact of the node degree constraint on the scheduling length. In the graph shown in Figure 4.5, scheduling length with TwoParentScheduling is shown with varying node degree constraint. The degree constraints are chosen as 4, 6 and 8 because with degree constraint lower than 4, not enough DCSPTs can be found to run the experiments and with degree constraint higher than 8, the result of the experiments remain similar to degree constraint 8. From the graph it can be seen that a higher constraint on node degree does not effect the schedule length very much.

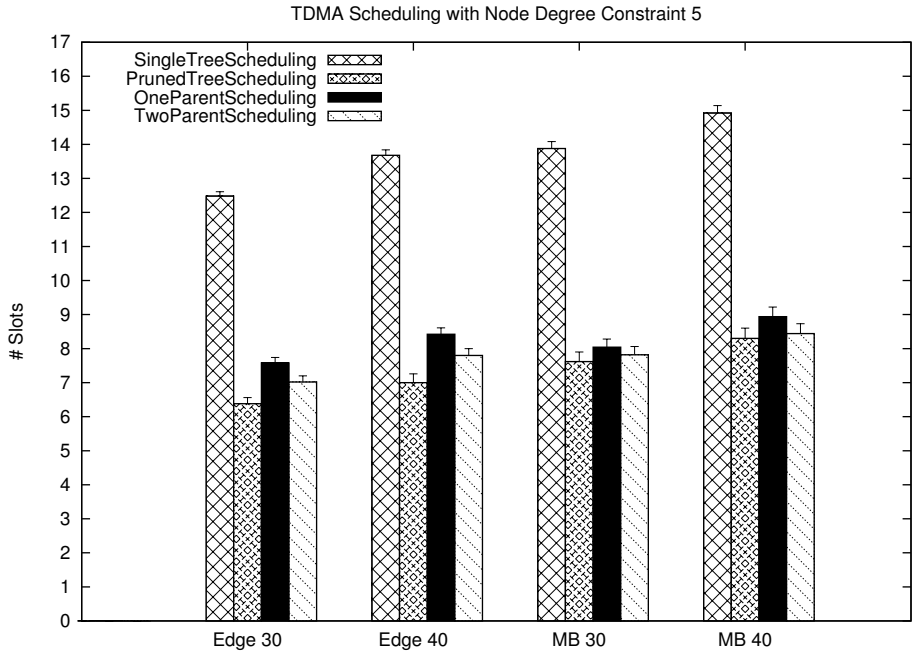


Figure 4.4: Average slot length of TDMA schedule

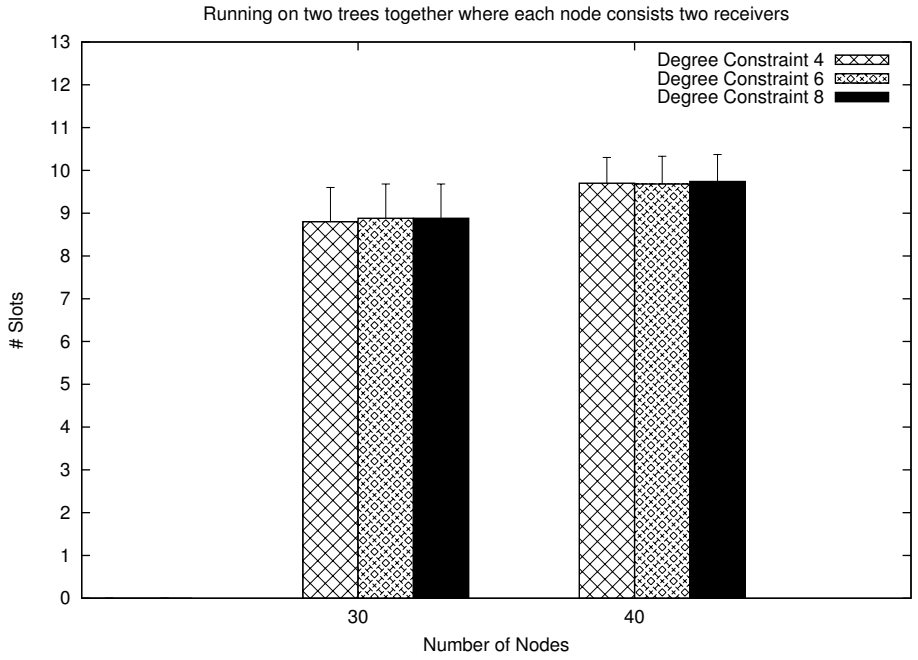


Figure 4.5: Average slot length of TDMA schedule - impact of node degree on schedule length

4.2 Routing Protocols

For the dynamic evaluation, we have designed several routing protocols. All of these routing protocols assume that there are two sinks in the network and there are two link-disjoint paths from each node to the sinks. Each node can forward its packets onto one of two designated paths: one is the primary which is the path to the closest sink and the other one is the secondary which is the path to the alternative sink.

For comparison, a routing scheme using only one path is also designed. This scheme is used with FullTreeScheduling, where there is only one sink in the network and with PrunedTreeSceduling, where there are two sinks in the network but each node can send to only one sink (the closest). In this scheme, the node stores only a single parent ID and the slot in which to send to this parent. A node trying to send a packet will use this slot to send the packet to its parent. If the link fails then it will not receive any acknowledgement and the node will just retry in later periods until it can successfully transmit the packet. In this scheme, the packet header

contains the ID of the source of the packet, the ID of the next hop neighbor and the sequence number of the packet.

A sender of a packet knows about a successful packet delivery by explicitly receiving an acknowledgement packet from the receiver. In a wired network, communication is full duplex so a node can sense the channel while transmitting to know the status of the channel. But, in wireless radio networks, the node cannot sense the channel while transmitting. Therefore, a missing acknowledgement packet is interpreted as a link failure. Time-slots are large enough to send a packet and receive back acknowledgements. For TwoParentScheduling, if the routing algorithm wants both parents to receive the packet, the slot requires space for 2 acknowledgements and the parents must be locally synchronized to know when to transmit their acknowledgement. Otherwise, the slot has space for one acknowledgement. Note that a node only knows the status of a link by trying to send a packet and receiving or not receiving an acknowledgement.

The routing protocols are link-based, in that they react locally to link failures. When, a node does not receive an acknowledgement for a packet, it is responsible for retransmitting the packet until it is successful. Four variations are considered based on how they react to link failure. The first two, Next Hop Neighbor Routing (NextHR) and Destination Routing (DestR), will react to a failed link by forwarding packets on both the alternative and primary path. This potentially causes a duplicate packet to be sent on multiple paths but gives the earliest indication that the primary link is up. The last two, Sampling Routing and Probabilistic Routing, do not create duplicate packets.

In the Next Hop neighbour Routing protocol (NextHR), each node forwards the packet on the primary path. When a node detects a link failure, it tries to send the packet on both of the paths, that is to both the parents, in the next period. It will continue to do this until it receives an acknowledgment. It sends to both parents, because it does not know for how long the link is down. The failure mode is packet-based, so, when a new packet arrives, it is, at least initially, sent to the primary parent only. In this protocol, if the packet is diverted towards a different sink, there is no requirement that the next node continues to forward it to that sink. Instead,

it will first try to send it on its primary path which may be to a different sink. To avoid short routing loops, a node will not send back to the node it just received the packet from. For example, if node a originally sends a packet to node b but the link is broken, then node a will send to node c. For node c the best path to use may be via node a. It will not send the packet back to node a since this will create a routing loop, instead it will send this packet on its alternative path (retransmitting to that parent until it is successful).

The packet header for this protocol holds the ID of the source of the packet, the ID of the current node (sender) the sequence number of the packet, and the ID of the parent (receiver). In the case of TwoParentScheduling, two receiver fields are needed. Both parent IDs are included in the receiver field after a failure occurs when the node wants both parents to respond. Otherwise, for TwoParentScheduling, only one receiver field is used and the other is invalid. For other scheduling algorithms, different slots are allotted to different links and, therefore, only one receiver is required.

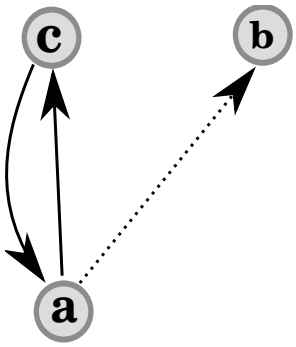


Figure 4.6: Routing loop in LinkAck-NextHR

For the DestR protocol, the packets will hold the destination (sink) of the packet. A node will look at the destination of the packet and send the packet using the link to that destination. Here the source will set the destination to the closest sink and if all the links are up then this packet will travel through the shortest path to that sink. If there is a link failure, the node will send to both parents. It will send a copy of the packet to the alternative parent with the alternative sink as the destination. It will also try (again) to send the packet to the primary parent with the primary sink as the destination. Therefore, the parent may send the packet to a sink that is not

the closest one from the parent.

Here, the packet header consists of the ID of the source of the packet, ID of the current node (sender), the sequence number of the packet, the ID of the parent (receiver) and the ID of the sink associated with that parent. In the case of TwoParentScheduling, there are two receiver fields and two destination fields. Under this scheduling scheme, when the protocol wants both parents to receive the packet, it puts both parent IDs in the receiver fields and their associated sink IDs in the destination fields; otherwise only one of the receiver fields and one of the destination fields are valid.

The next two protocols send to one parent at a time. In Probabilistic routing, a node, will send a packet on its primary link with probability l_p and send on its alternate link with probability $1 - l_p$. The value of parameter l_p determines the percentage of time a node will use its primary link. The choice of link does not depend on link failures. If a link failure occurs, the same probabilistic approach is applied to the next transmission of the packet from this node. The decisions at each node are also independent of each other. The packet header contains the ID of the source of the packet, ID of the current node, the sequence number of the packet and the ID of the parent.

The fourth protocol is Sample-based Routing. In this protocol, the link-choosing decision is controlled by the parameter n. A node starts sending a new packet on its primary link. When a link failure occurs, it will switch to its alternative link. Then it will try n times on its alternative link before it switches back to try its primary link. So, rather than continually testing if the primary link is up, it uses the alternative link for a pre-determined number of tries and, if that is not successful, it checks back with the primary link. If n = 0, the node will continue to send on its best link and never send on the other link. When n = 1, a node will continue to alternate between the links if primary path fails without any retries on the same link. The packet header contains the ID of the source of the packet, ID of the current node, the sequence number of the packet and the ID of the parent.

In Single path routing using FullTreeScheduling and PrunedTreeSceduling, the logical topologies have the same node degree constraint as the individual routing

trees but in the other routing schemes the nodes in the logical topologies may have degree higher than the nodes of individual routing tees. This is because as the two routing trees are merged, there may be more links incident to each of the nodes.

4.3 Experimental Setup

Like the static simulation, dynamic analysis is done on *random sensor network graphs* with 30 nodes uniformly randomly placed in a grid of size 25×25 with node degree constraint of 5 and sink degree constraint of 15. The range of each node is $r = 6$. The position of two sinks are chosen from the previous experiments such that the logical topology has two link-disjoint paths to both of the sinks from each of the nodes. For experiments that do not have alternative paths, two scheduling schemes are used. For the first type there is just one routing tree covering the full network (FullTreeScheduling) and in the second type there are two pruned trees so that each node sends to its closest sink only (PrunedTreeScheduling). For the experiments with alternative paths, TwoParentScheduling is used. The simulator is written in the JAVA programming language. Each simulation has a length of $10,000$ periods. There are 10 topologies. There are 50 trials for each of the 10 topologies and the average is taken. All the experiments are done with the $Middle - best$ scheme for choosing the sinks. 95% confidence intervals were calculated for each result. DCSPT is built from physical topologies using hop-count distance.

4.3.1 Frame Generation

The experiments are done to check the performance of the network in a full load situation. In the experiments, all the nodes in the network work as sources. At the start of the simulation, all the nodes generate one packet each and then stop generating packets. This is called a session. The nodes again generate another packet once each of the previously generated packets are received by the sink(s) and this continues until the simulation ends.

4.3.2 Link Failure

Because of the unreliable nature of wireless connections, links between nodes can fail so that data transmission is not possible. These failures are simulated by randomly choosing multiple links in each slot and mark them as DOWN. In the beginning of the simulation all links have a status of UP indicating that they are working. In each slot, with failure probability P_f, each UP link is failed and the status is changed to DOWN. The duration of failure is exponentially distributed with a mean inactive time, MIT slots. Links that are already Down are checked in each slot if they are inactive for their allocated time, and if yes their status is changed to UP. So the parameter P_f denotes how frequently a link should fail and the parameter MIT denotes for how long a link will be in a failed state.

4.4 Simulation Metrics

In the simulations the following metrics are measured for different protocols on different topologies : Average time to deliver all packets, Network Lifetime, Average Hop-Count.

4.4.1 Average time to deliver all packets

Average time to deliver all packets is defined as:

Sum of the number of slots taken to deliver all the packets of a session/number of sessions.

In the numerator we sum the delays to deliver the packets generated during a session. The denominator represents the number of times all the nodes generate packets.

This metric represents the amount of delay faced by the packets in the network. There are two ways packets are delayed in the network. First is the delay due to link failure. Here packets wait in the node's queue because they cannot be delivered because of link failures. The second reason is the load of the network. The higher the load of the network, the more number of packets a node has to handle. This includes both its own packet and its successors packets. These packets accumulate

in the queue because in one slot the node can transmit only one packet and in one period a node only get to transmit in one slot. In the worst case packets accumulated in the queue in one period are the number of children plus one. This happens when the node receives packets from all its children and generates one itself in a period. The lower the delay the better because lowering delay ensures better quality of service and for some applications a delay bound should be maintained.

Network Lifetime

Network Lifetime (NLT) is defined as the time elapsed until the first node dies due to lack of energy. It is measured in number of slots. Every node in the network except the sinks have the same initial amount of energy and for transmission and reception some of the energy is lost. So a node will die when it loses all of its initial energy.

This is very important in WSNs because it indicates how energy efficient the protocol is when applied to a certain topology. Protocols with high NLT are desirable because sensor nodes have a limited amount of energy and it is difficult to recharge them.

Average Hop-Count

Average Hop-Count (AHC) is defined as

AHC = Total hops traveled by each unique packet from source to sink / number of unique packets received at sinks

The unit of this metric is hops per packet. In this calculation retransmissions are not considered so for multiple copies of the same packet only the first one that comes to one of the sinks is considered. This indicates the average path length needed to be traveled by the packet from source to the sink.

4.5 Experiment Methodology

In this section, all the assumptions, parameter settings and design of experiments are presented.

4.5.1 Assumptions

Several assumptions are made when implementing the routing protocols. Infinite buffer space is assumed in the nodes so no packets should be dropped because of lack of space. It is assumed that all the nodes have knowledge of the best sink and alternative sink and the next hop neighbors to reach these sinks. Nodes also know the slots to reach these next hop neighbors. It is assumed that by not receiving an acknowledgment packet, the sender knows about link failure and there is no other way to know about links status without transmitting a full packet. It is assumed that computation is very fast so the time needed for computation in the nodes is not recorded. It is assumed that nodes are well synchronized so the TDMA scheme can be implemented perfectly and each slot time is enough for transmitting a full packet and receiving its acknowledgment(s). The time required for the nodes to change states (sleep-awake, awake-sleep) is not considered and it is assumed that state changes occur without any delay.

When measuring the network lifetime the power required for computation is not measured but only for communication. State changes of nodes also do not consume any energy. It is assumed that nodes know in which slots to send and receive. In TDMA scheduling idle listening is not required. So they remain active during those slots only and in other slots they remain in sleep mode. In sleep mode they consume negligible amount of power. Sending and receiving a data packet only consumes energy. Energy consumption because of sending and receiving acknowledgment packets is not considered.

For example if we consider the size of the payload in a packet is $100B$ [35], size of node address, sequence number is $6B$ and $2B$ respectively [38]. Each data packet also includes a $2B$ CRC[51]. The size of the acknowledgment packet is $8B$[51]. Then the slot size in LinkAck-NextHop routing, LinkAck-Dest routing, Probablistic routing, Sample routing and one-source routing are $136B$, $148B$, $130B$, $130B$, $124B$.

4.5.2 Design of Experiments

For the simulation, some parameters are used. For measuring network lifetime each node is assigned with a power of 3035000 units in the beginning of simulation. Transmission and reception of a single packet requires 29 and 31.7 unit of power respectively as stated in [35]. Each experiment is run for $10,000$ periods and the average is taken. The results presented are shown within 95% confidence interval.

Each of the experiments is performed with 30 sources where there are 30 nodes in the network. An important parameter of the simulation is the link failure probability P_f. For each experiments this is varied from 0.0 to 0.04 to evaluate the performance with no link failure up to a high link failure.

The mean inactive time (MIT) denotes how long the links should remain down. Experiments with MIT of 32, 64, 128 slot times are done to check the effect of length of link failure on the protocols.

For each experiment set there are three scheduling schemes. Two with no alternative paths for the nodes - FullTreeScheduling and PrunedTreeScheduling, and one with having alternative paths for the nodes - TwoParentScheduling. With TwoParentScheduling the routing schemes used are NextHR (TPS-NextHR), DestR (TPS-DestR), Sample(1) (TPS-Sample(1)), and Probability(0.8) (TPS-Prob(0.8)). We have run some experiments to check the performance of Sample and probabilistic routing with different parameters and found that Sample(1) and Probability(0.8) work best with the two respective routing schemes. So we have used the schemes with these two parameter values in the rest of the experiments.

4.6 Analysis of Results

4.6.1 Experiments with All Possible Sources

We run experiments considering all the nodes in the network as sources to check the performance of the network in a full load situation. For this experiment, there are 30 sources in the network. In the first slot, all the sources generate one packet each and then they stop generating packets. All the sources again generate one packet each after all the 30 packets are received by the sink(s) and this continues until the

simulation ends. We record the number of slots it takes to receive all the 30 packets by the sink(s) and the average time is calculated after the simulation. For this set of experiments MIT is set to 64.

From the experiment shown in Figure 4.7[1], it can be seen that with no link failure, topologies with single sink needs more time to receive all the packets than topologies with multiple sinks. This is because, with multiple sinks, the load is divided between the sinks and the path length from the nodes to the sinks is reduced which reduces the delay. Again with increasing failure-probability, the time needed to deliver all the packets is increased as nodes need to retransmit the packet and use longer alternative links to send the packets. Also with low failure-probability, TPS-NextHR requires less time than PrunedTreeSceduling as using alternative links it can transfer the packets quickly. But as failure-probability increases, PrunedTreeScheduling requires less time to deliver compared to TPS-NextHR. This is because, in high failure-probability with a high load, using alternative links creates congestion in the network, increasing the delay. TPS-DestR requires more time than the previous two schemes because here intermediate nodes can choose to select longer alternative paths though their primary path is active. For TPS-Sample(1), delay is higher as the scheme might choose alternative links when the primary link is active. With TPS-Prob(0.8), nodes chose links randomly so packets can bounce in the network which increases the delay.

With no link failure, average hop count (AHC) for topologies with single sink is higher than the topologies with multiple sinks which is shown in Figure 4.8[2]. This is because, with the presence of multiple sinks the path to nearest sink from the nodes is lower for the topologies with multiple sinks than topologies with single sink. With the increase in link failure, AHC remains same for the topologies with only one path from the nodes to the sink(s) (FullTreeScheduling and PrunedTreeSceduling) but AHC increases for topologies with alternative paths from the nodes to the sinks (TPS-NextHR, TPS-DestR, TPS-Sample(1), TPS-Prob(0.8)). This happens

[1] 95% confidence intervals were calculated but left of the graphs so that the lines can be more easily differentiated. The confidence intervals are within 1.5% of the mean.

[2] 95% confidence intervals were calculated but left of the graphs so that the lines can be more easily differentiated. The confidence intervals are within 0.6% of the mean.

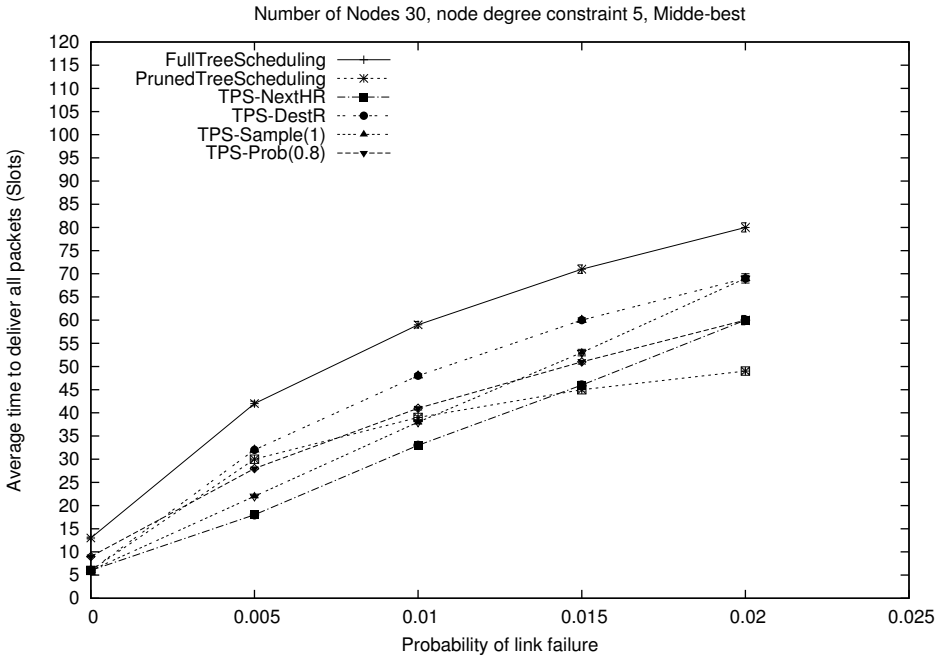


Figure 4.7: Average time to deliver all the packets to the sink(s)

because with the presence of link failure for the previous schemes nodes have to use a single path but the latter schemes use longer alternative paths, increasing AHC. For low failure-probability (≤ 0.01) AHC is lower for TPS-NextHR and TPS-Sample(1) than FullTreeScheduling.

Next experiment is performed to measure the network lifetime (NLT). For this experiment, nodes are allocated 103500 units of power in the beginning of the simulation. From the experimental result shown in Figure 4.9 [3], it can be seen that, without link failure, topologies with multiple sinks have slightly higher NLT. This happens because as load is divided between the two sinks, no node has to carry a huge number of downstream message thus increasing NLT. As links starts to fail, less number of packets reach the nodes near the sinks so they have to transfer less packets thus increasing NLT. With higher link failure probability though links fail more frequently, higher number of retransmission and redundant transmissions through alternative links means that the number of packets transfered by the nodes near the sinks do not change significantly with the increase of link failure

[3] 95% confidence intervals were calculated but left of the graphs so that the lines can be more easily differentiated. The confidence intervals are within 2% of the mean.

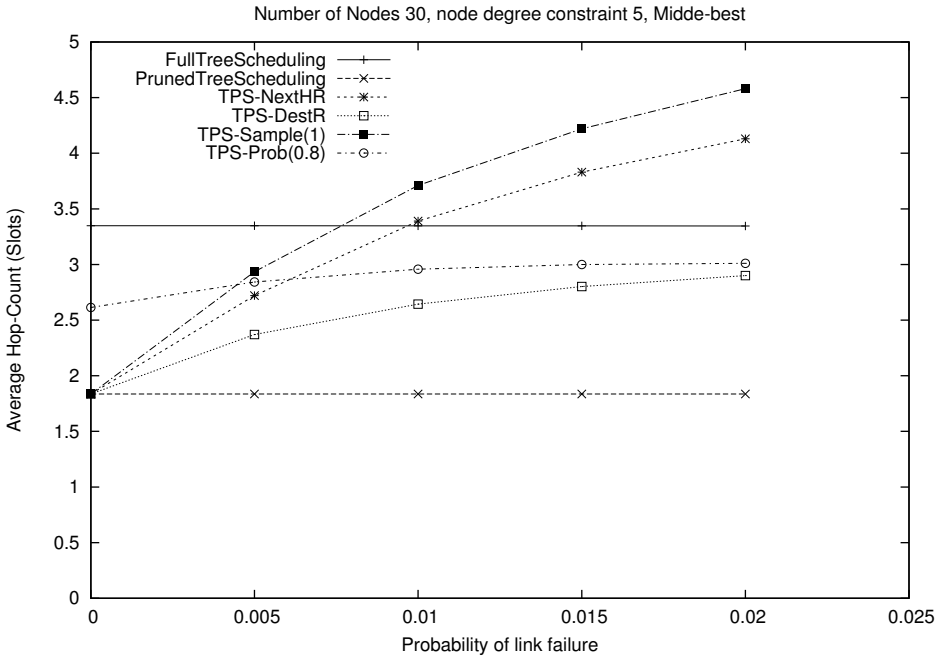


Figure 4.8: Average hop-count

so NLT does not change significantly with the changing link failure probability. As in PrunedTreeSceduling, load is divided between two sinks and nodes do not send packets to both of the links, nodes near the sinks have less load to carry so it has the highest NLT, with failure-probability. Because of sending two packets on both the links in case of link failure, TPS-NextHop creates higher number of duplicate packets than other schemes thus increasing the load of the nodes near the sink(s). This is why NLT is lowest for TPS-NextHR.

To check the effect of MIT on the time to deliver all packets, we have run some experiments changing the MIT. For this experiment MIT is set to 32, 64 and 128. From the results shown in Figure 4.10[4], we can see that, the time to deliver increases as MIT increases. This happens because as MIT increases, links fail for longer period of time, there are higher packet retransmissions and nodes need more time to deliver the packets to the sink(s). Also fewer packets reach the nodes closer to the sink(s) so NLT increases as MIT increases which is shown in Figure 4.11 [5]

[4]95% confidence intervals were calculated but left of the graphs so that the lines can be more easily differentiated. The confidence intervals are within 1% of the mean.

[5]95% confidence intervals were calculated but left of the graphs so that the lines can be more easily differentiated. The confidence intervals are within 1.6% of the mean.

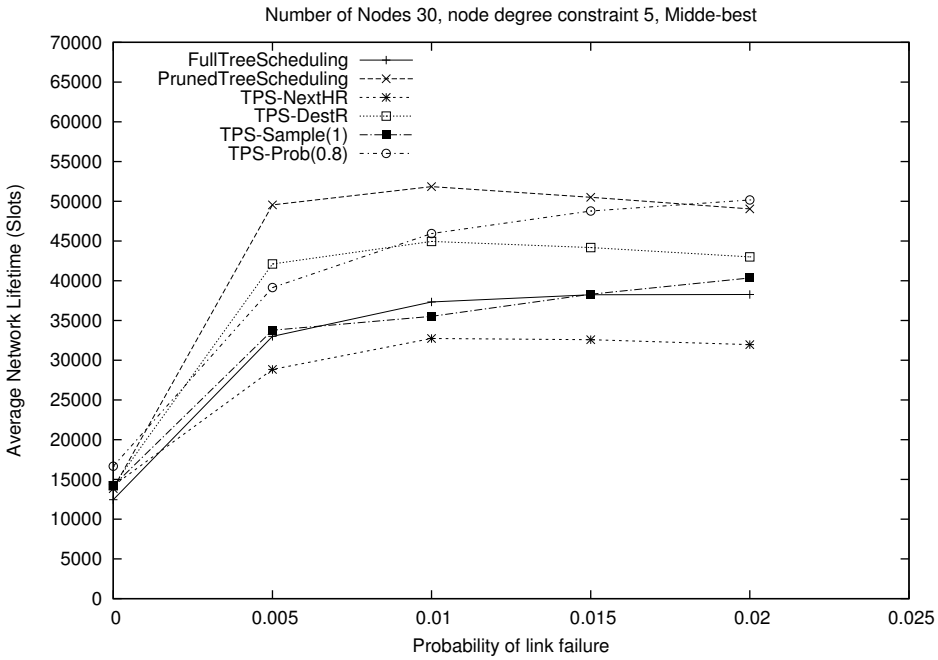


Figure 4.9: Network lifetime

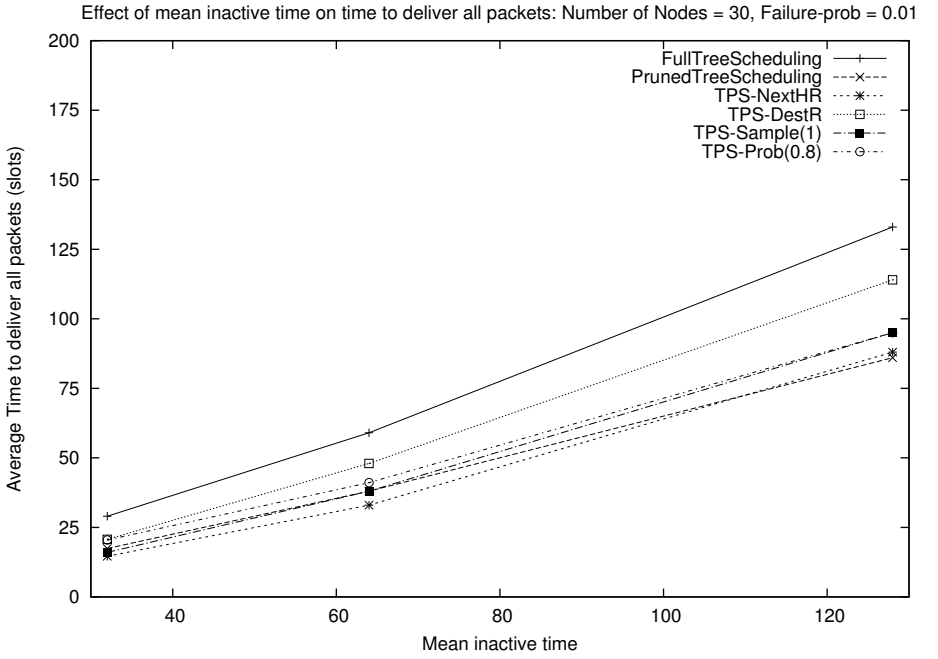


Figure 4.10: Effect of MIT on time to deliver all packets

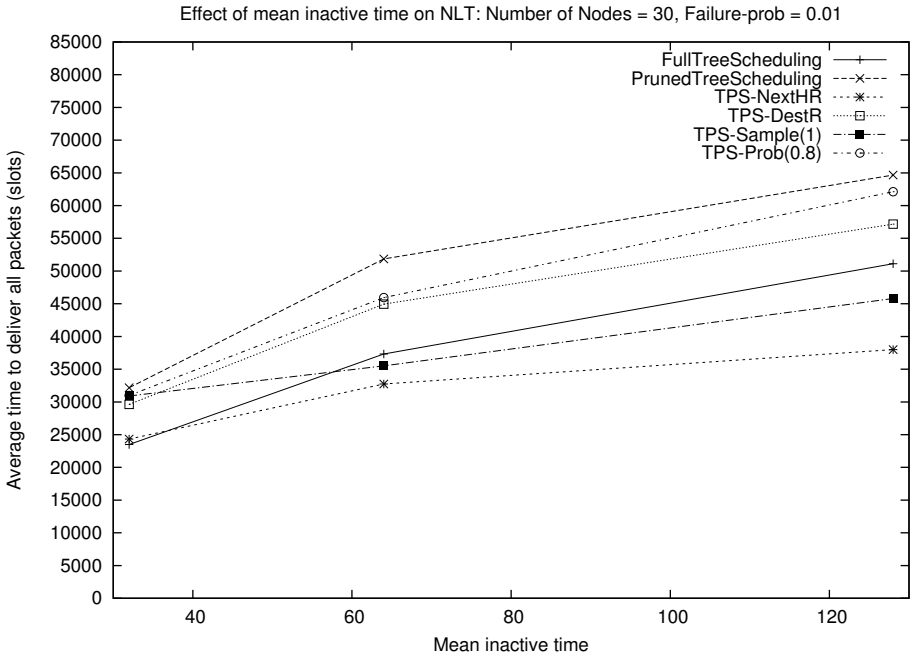


Figure 4.11: Effect of MIT on NLT

To explore the effect of load on the network, we have run experiments changing the load of the network by changing the number of sources in the network. The number of sources selected for this experiment are 8, 16 and 30. From the experiment, shown in Figure 4.12, it is found that increasing number of sources increase the delay to deliver all the packets to the sink(s) because we need more time to deliver more packets. But, as the load of the network increases, the difference between performance of TPS-NextHR and PrunedTreeSceduling decreases. With low load, TPS-NextHR works better than with higher loads, because with higher loads TPS-NextHR creates congestion in the network and its performance degrades.

4.7 Chapter Summary

In this chapter, we have designed effective Time Division Multiple Access (TDMA) schemes for scheduling of the proposed logical topology. The schemes use two sinks and try to minimize schedule length by using parallel transmissions between the two routing trees. We have compared the scheduling lengths resulting from these schemes in our proposed logical topologies with two other scheduling schemes,

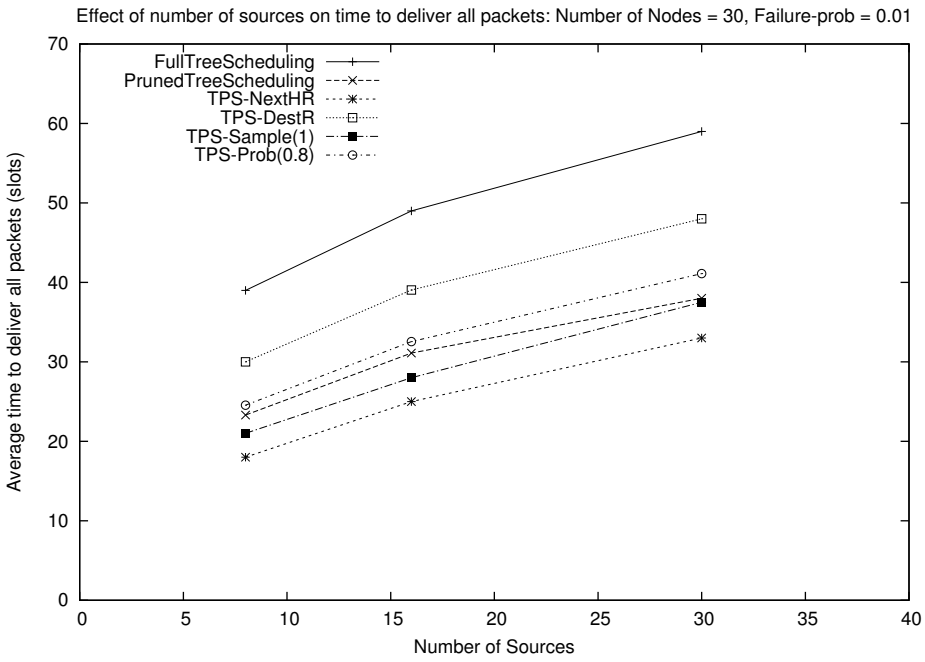


Figure 4.12: Effect of number of sources on time to deliver all packets

FullTreeScheduling and PrunedTreeScheduling which does not consider parallel transmissions between the two routing trees. From the experiments done it can be seen that, our proposed scheme can produce a TDMA schedule, which is smaller than that given by FullTreeScheduling.

Also, we have performed a dynamic evaluation of our proposed logical topologies using four simple routing protocols, Link Acknowledgement - Next Hop Routing (LinkAck-NHR), Link Acknowledgement - Destination Routing (LinkAck-DestR), Sample Routing and Probabilistic Routing with TwoParentSchedling. We have presented the design of these protocols and performed experiments with varying failure rate, node degree constraint, number of sources, mean inactive time to evaluate and compare these protocols on different topologies in terms of time to deliver all the packets, network lifetime and average hop-count. We have compared the protocols using multiple paths with protocols using single path (Single sink routing with FullTreeScheduling and PrunedTreeSceduling).

The results from the dynamic simulations show that, TPS-NextHR works better than other routing schemes in terms of delivering all the packets to the sink(s) with the presence of moderate link failure as it can efficiently chose optimal paths

to the sinks to deliver packets with lower delay. With high link failure, however PrunedTreeSceduling works better than this scheme as TPS-NextHR will create congestion in the network increasing the delay. If load of the network is decreased by decreasing the number of sources in the network, TPS-NextHR works better than PrunedTreeSceduling even with high link failure probability. Increasing mean inactive time also increases the delay to deliver all the packets to the network. Experiments with network lifetime shows that, PrunedTreeSceduling has higher NLT than TPS-NextHR as it does not create duplicate packets in the network. Also increasing MIT increases the NLT as fewer packets reach the nodes closer to the sinks. There is a trade-off between delayand NLT in using PrunedTreeSceduling and TPS-NextHR where PrunedTreeSceduling has higher delay but higher NLT also. As number of retries to send one packet is lower for TPS-NextHR, if we limit the number of replies than TPS-NextHR has better performance than PrunedTreeSceduling. With high MIT and low load, Sample(1) scheme has lower delay than PrunnedTreeScheduling and it has higher NLT than NextHR. So with a high MIT, the Sample scheme can be used instead of NextHR if NLT is more important than delay.

Chapter 5

Design of Robust Topologies in Grid Graph

In this chapter we design robust logical topologies for two-sink WSNs with a grid physical topology. The range of a node is just enough to communicate with its nearest nodes. Therefore nodes can communicate with the nodes positioned on other grid points vertically, horizontally and diagonally one hop away. The robustness is ensured by having, for each sensor node other than the sinks, two link disjoint paths, each one to a different sink. In addition, the design finds the shortest path to the corresponding sink from the nodes, and there is a limit on the number of children each node can have. The shortest path will ensure that, in a error free environment, communication between the nodes and the sinks can be done with a reduced energy consumption and delay and we hope to reduce the hotspot problem by having limit on the number of children. Here we are designing in such a way that no node can have more than three logical neighbors and the number of nodes with three logical neighbors is also kept low. In this thesis, we assume that, sinks have the same range as the ordinary nodes so they can only communicate with one-hop neighbors and they have no restriction on the number of logical neighbors they can choose from among their physical neighbors.

In this chapter, we consider the design of a logical topology and evaluate its properties. We also evaluate the robustness and path quality of the proposed topology by simulation in the presence of failed links. Network operation schemes for selecting paths, detecting faults and choosing alternative paths are considered in the

last section where we perform dynamic evaluation of our deployment topologies.

5.1 Problem Definition

In this chapter, we consider deploying the sensor nodes on points of a grid shown in Figure 5.1. The grid has M columns and N rows and all the unit squares have a side of length r. Our design is independent of the value of r, M and N. We assume that each grid point has a sensor. The sink can be placed on the grid points. We assume that each of the sensor nodes has a communication range of $\sqrt{2}r$, so that each node can communicate with nodes horizontally, vertically and diagonally. So, a node can have a maximum of 8 neighbors unless it is on the boundary, where it has a maximum of 3 neighbors if it is a corner or a maximum of 5 neighbors on a side.

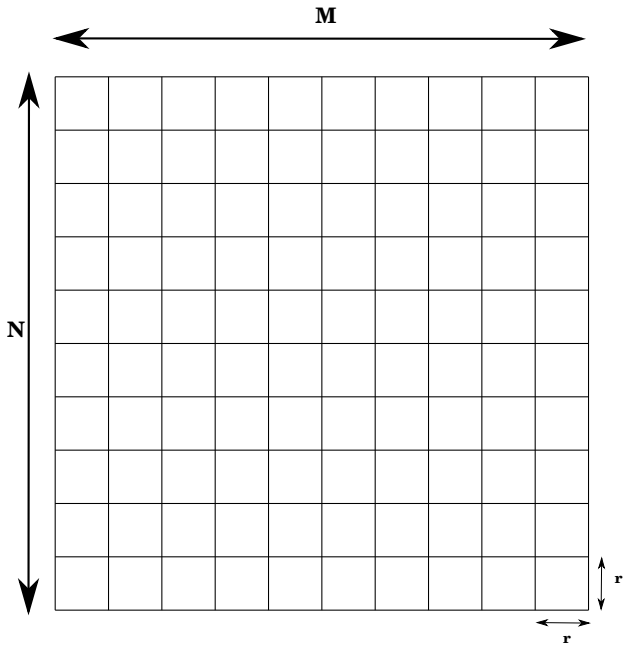


Figure 5.1: Grid based topology

The problem is to select logical neighbors for each of the nodes or select appropriate links in the network so that the logical topology spans all the nodes, has robustness, maintains path quality and the number of neighbors for each of the nodes is minimized. For this thesis, we define d-degree-constrained topologies as topologies where each node can have a maximum of d logical neighbors. In this

thesis, we expand the work developed by Reza[36] where the author only considered one sink and horizontal and vertical links. We are extending this to the case where there are two sinks in the network and the nodes also have diagonal links.

5.2 Grid Graphs with Horizontal and Vertical Links

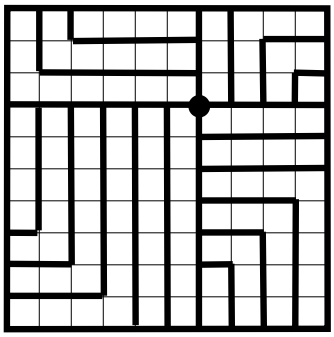


Figure 5.2: Pattern for the grid network with the leaves and the four quadrants connected

Reza [36] worked in the context of the reliability of underwater sensor networks, where each of the sensor nodes are positioned in a grid pattern. In designing the logical topology of the network, the author tried to minimize the number of logical neighbors of each of the sensor nodes. The author proposes a pattern for a grid graph which is defined as a 3-degree-constrained shortest path tree rooted at the sink and spanning all nodes in the grid with $(LB + 2)$ number of 3-degree nodes in the worst case where LB is the lower bound on the number of 3-degree nodes in such a tree. The network has only one sink positioned at the middle of the grid and for each node only horizontal and vertical links are allowed. The pattern ensures that there are no 4-degree nodes in the network. Additional links are needed to be added to the design to give 2 edge-connected topology. The final logical topology is shown in Figure 5.2. The addition of the links still maintains the 3-degree node constraint, but increases the number of 3-degree nodes in the network.

5.3 Grid Graphs with Multiple Sinks

In this section we design a pattern for multiple sink grid graphs with horizontal, vertical and diagonal links. The grid has a dimension of $M \times N$. The two sinks are placed on the two opposite corners of the grid. So the first sink is positioned on point $(0, 0)$ and the second sink is positioned on point $(M-1, N-1)$. All the boundary links are included in the pattern because these links are part of shortest paths to the sink on that axis. There are two diagonals going through the two sinks which we call the two main diagonals. The first main diagonal starts at $(0, 0)$, joining the nodes on points (x_i, y_i) where $x_i = y_i$, hitting the boundary on point $(M-1, M-1)$ if $M \leq N$ or $(N-1, N-1)$ if $M > N$. The second main diagonal starts at $(M-1, N-1)$, joining the points (x_i, y_i) where $y_i = x_i + (N-M)$ if $N \geq M$, hitting the boundary on point $(0, N-M)$. If $M > N$, it joins points (x_i, y_i) where $x_i = y_i + (M-N)$ and hits the boundary on point $(M-N, 0)$. These two diagonals must be included in the logical topology because they are the only shortest path to the sinks that nodes on the diagonals go through.

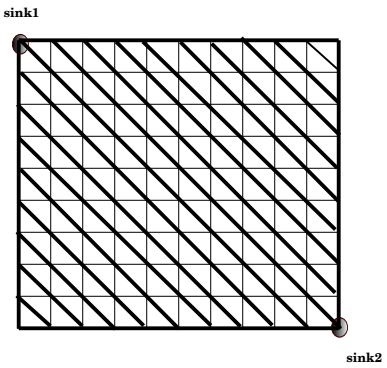
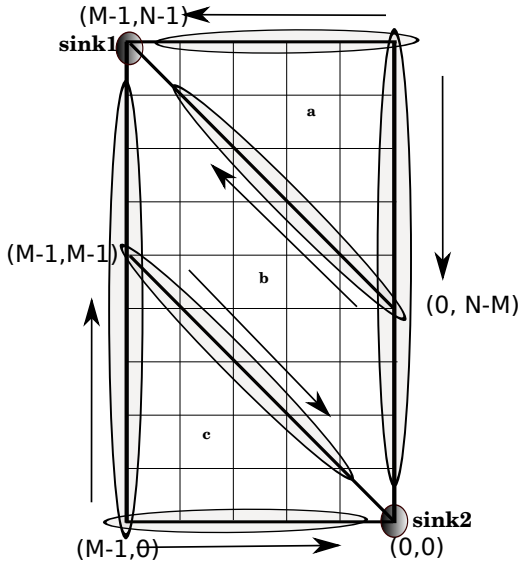


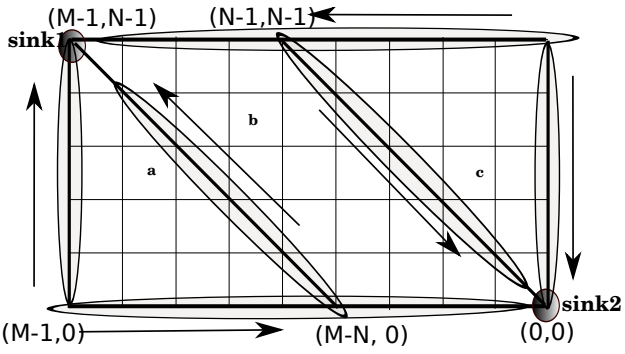
Figure 5.3: Multiple sinks in grid graph with diagonal links

A solution to find the logical topology is to use all the diagonals and the axes as seen in Figure 5.3. The number of 3-degree nodes in the pattern is $2\,(M+N-4)$ not including the sink's degree. With rectangular grids, we can reduce the number of 3-degree nodes in the pattern under certain conditions. We assume that the grid dimension is $M \times N$ and $U = max(M, N)$, $V = min(M, N)$ as shown in Figure 5.4. In this pattern all the links on the axes and the two main diagonals are included.

We consider three separate regions, denoted by a, b and c, created by the two



(a)



(b)

Figure 5.4: Multiple sinks in grid graph a)$M < N$ b)$M > N$

main diagonals as shown in Figure 5.4. If $M < N$, in region c, the points included are (x_i, y_i), where $x_i > y_i$. In regions b and a, the points included are (x_i, y_i) where $y_i < x_i + U - V$ and $y_i > x_i + U - V$ respectively. If $M > N$, in region c, the points included are (x_i, y_i), where $x_i < y_i$. In regions b and a, the points included are (x_i, y_i) where $x_i < y_i + U - V$ and $x_i > y_i + U - V$ respectively. For the nodes in region b, we take all the links parallel to the longer axis. That is we connect from the first main diagonal to the second main diagonal. This creates 3-degree nodes at the main diagonals including where the main diagonals connect to the axes (other than the sinks). So there are $2(V - 1)$ 3-degree nodes. For the nodes in region a and c, we take all the diagonals in these two regions that connect from axis to axis. In each of the two region, this creates $2(V - 2)$ 3-degree nodes on the axis other than the sinks. So there are $2(V - 1) + 4(V - 2) = 6V - 10$ 3-degree nodes in the pattern. The resulting topology is shown in Figure 5.5.

Suppose $M < N$. This pattern has less number of 3-degree nodes than pattern with all diagonal links if

$$\begin{aligned} 2(M + N - 4) &> 6M - 10 \\ M + N - 4 &> 3M - 5 \\ N &> 2M - 1 \end{aligned}$$

Similarly we can design a pattern for a grid where $M > N$ and this design will have less number of 3-degree nodes than the design with all diagonals if $M > 2N - 1$.

5.4 Static Analysis of the Pattern

In this section, we evaluate the probabilistic robustness of the logical topologies designed. The evaluation done here is static because we do not consider the dynamic aspects of the network such as dynamic selection of routing paths and network operations like routing, fault detection that changes with time. Dynamic evaluation of the topologies is described in the next section. The metric used for the evaluation are *quality of path, sink load, effect of failure* which are described in Chapter 3.

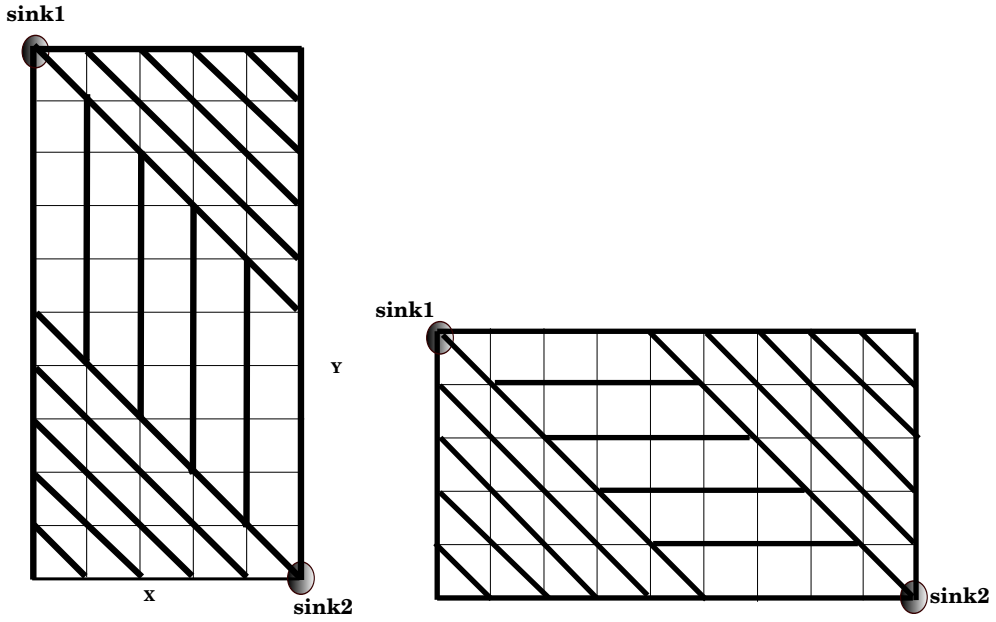


Figure 5.5: Pattern with reduced number of 3-degree nodes a)$M < N$ b)$M > N$

5.4.1 Experimental Setup

We have conducted the static analysis on *grid graph* with horizontal, vertical and diagonal links, having two sinks. The analysis is done with two grids. The first one is a rectangular grid of size 4×9 and the second one is a square grid of size 6×6. Both have 34 nodes and 14 and 16 3-degree nodes in the logical topology. The distance between the grid points is 1 unit and the two sinks are positioned in the two opposite corners of the grid. The range of the nodes is such that a node can communicate with the nodes that are within one hop distance horizontally, vertically and diagonally. Each experiment is repeated 50 times with the same physical topology and their average is taken to have an acceptable 95% confidence interval. In the experiments with link failures, multi-link failure is considered, where with a certain probability each link in the logical topology is removed.

5.4.2 Experimental Results

From the experiments with the grid graph it is found that load is well-distributed between the two sinks with 19 nodes being closer to one sink and 15 nodes being closer to the other. According to the design, the logical topology only has nodes

with degree 2 and 3 with both the sinks having a degree of 3. It can be found that for both the grids the path lengths are similar but for the rectangular grid the alternate path length is larger. This is because, for square grids, we have more diagonal links than the rectangular grid. So nodes use diagonal links to reach the alternative sink which reduces the alternative path length.

	Path Length	Alternate Path Length
4×9	2.47	6.11
6×6	2.64	4.41

Table 5.1: Quality of path

Figure 5.6 presents the robustness of the topology to multi-link failure. In multi-link failure cases the number of disconnected nodes increases as the probability of link failure(p_i) increases. The topologies are robust to multi-link failure because around 10% of the nodes are disconnected from the sinks even with a high p_i of 20%. But as p_i increases further there is almost an exponential increase in the number of disconnected nodes. From the experiments it can also be seen that both the grids have a similar degree of robustness.

Figure 5.7 shows the average path length of the connected nodes from the closest sink in the presence of multi-link failure. It can be seen that path length increases initially as p_i increases since the longer alternative paths are taken but then the average path length decreases as p_i increases further. This occurs because as p_i increases, more nodes become disconnected and therefore the average path length decreases. Path length from disconnected nodes are not included in the calculation and nodes close to the sinks (*eg.* shorter paths) are more likely to be connected. From the experiments it is found that the topology can handle around 25% percent of link failures with alternative paths but it fails to perform under higher link failures. From the experiments it can also be seen that both the grids have similar characteristics with respect to path quality.

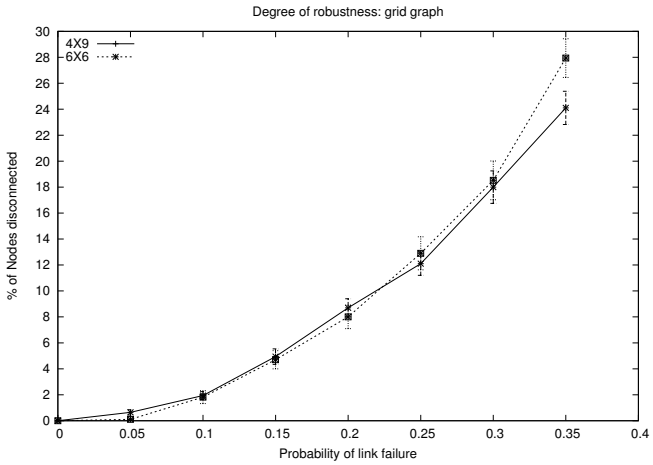


Figure 5.6: Degree of robustness in multi-link failure grid graph

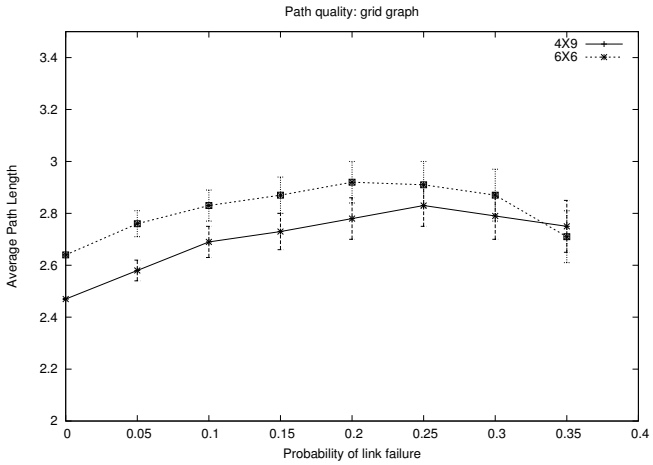


Figure 5.7: Path quality in multi-link failure grid graph

5.5 Dynamic Analysis of the Pattern in Grid Graph

In this section, the dynamic behavior of the logical topology designed for grid graphs is evaluated. The dynamic behavior includes variable failure rate, changing link status (links going up and down), dynamic selection of paths to avoid failed links. Performance of the logical topologies is evaluated in terms of resiliency to link failures, network lifetime.

5.5.1 Experimental Setup

In these experiments grid graph with two sinks is used as the topology. For these experiments, we have used a single topology of a rectangular grid of size 4×9 with unit distance between the grid points. The two sinks are positioned in the two opposite side of the grid (Figure 5.5). So there are 34 nodes and two sinks in the network. The range of the nodes is such that a node can communicate with the nodes that are within one hop distance horizontally, vertically and diagonally. Each experiment is replicated 500 times with the same physical topology and their average is taken. All the results shown are shown with 95% confidence interval. The experiments are first done with all the nodes considered as sources then we have done experiments with 16, 8 and 4 sources in the network. The sources in the beginning of the simulation generate one packet each and stop generating packets. After the sinks have received all the packets, the sources again generate one packet each. This continues until the simulation ends. Mean inactive time is set to 16. Experiments are done with FullTreeScheduling and PrunedTreeScheduling using single path routing and with TwoParentScheduling NextHR (TPS-NextHR), TwoParentScheduling DestR (TPS-DestR), TwoParentScheduling Sample(1) (TPS-Sample(1)) and TwoParentScheduling Probability(0.8) (TPS-Prob(0.8)). In the FullTreeScheduling, there is only one sink positioned in one corner of the network so each node has only one path to the sink. In PrunedTreeScheduling, the nodes report only to the closest sink so there is no alternative path for the nodes. In the TwoParentScheduling, nodes have two link disjoint paths, one to each of the sinks.

5.5.2 Experimental Results

From the experiment shown in Figure 5.8[1], it can be seen that with no link failure, topologies with single sink need more time to receive all the packets than topologies with multiple sinks. This is because, with multiple sinks, the load is divided between the sinks and the path length from the nodes to the sinks is reduced which reduces the delay. Again with increasing failure-prob, the time needed to deliver all the packets is increased as nodes need to retransmit the packet and use longer alternative links to send the packets. TPS-NextHR requires more time to deliver the packets compared to PrunedTreeSceduling as generating duplicate packets in a high load network creates congestion in the network, increasing delay. TPS-DestR requires more time than the PrunedTreeSceduling because here intermediate nodes can choose to select longer alternative paths though their primary path is active. For TPS-Sample(1), delay is higher as the scheme might choose alternative links when the primary link is active. With TPS-Prob(0.8), nodes chose links randomly so packets can bounce in the network which increases the delay.

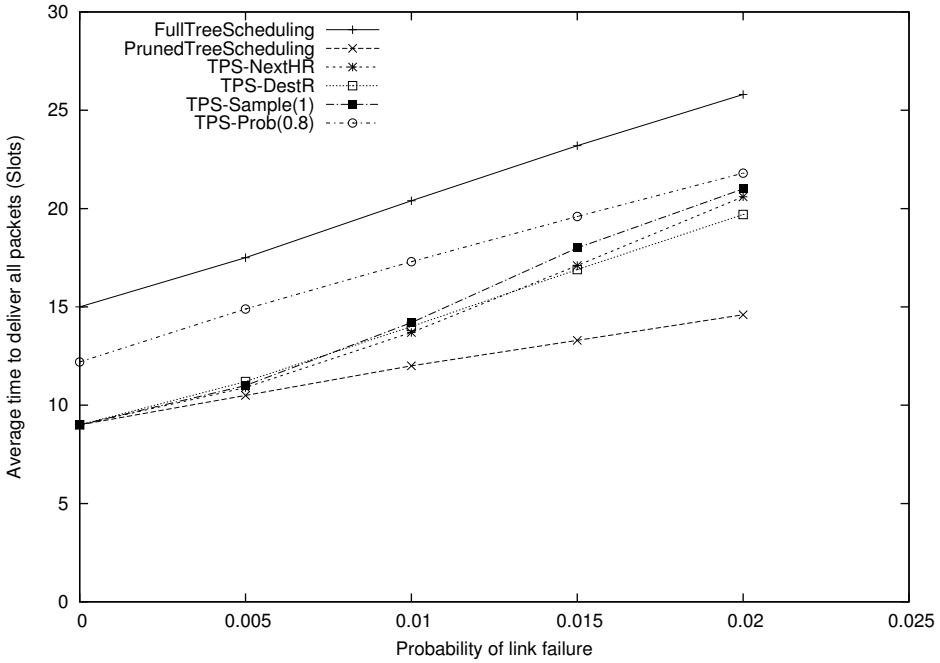


Figure 5.8: Average time to deliver all the packets to the sink(s)

[1]95% confidence intervals were calculated but left of the graphs so that the lines can be more easily differentiated. The confidence intervals are within 0.5% of the mean.

With no link failure, average hop count (AHC) for topologies with single sink is higher than the topologies with multiple sinks which is shown in Figure 5.9[2]. This is because, with the presence of multiple sinks the path to nearest sink from the nodes is shorter for the topologies with multiple sinks than topologies with single sink. With the increase in link failure, AHC remains same for the topologies with only one path from the nodes to the sink(s) (FullTreeScheduling and PrunedTreeSceduling) but AHC increases for topologies with alternative paths from the nodes to the sinks (TPS-NextHR, TPS-DestR, TPS-Sample(1), TPS-Prob(0.8)). This happens because with the presence of link failure for the previous schemes nodes have to use a single path but the latter schemes use longer alternative paths, increasing AHC. For low Failure-prob (≤ 0.02) AHC is lower for TPS-NextHR and TPS-Sample(1) than FullTreeScheduling.

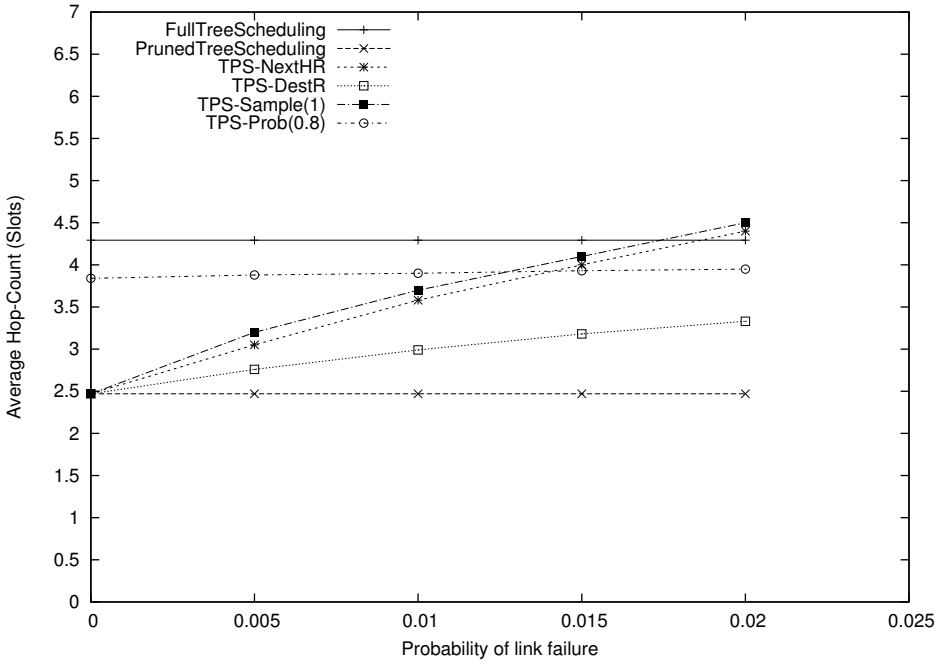


Figure 5.9: Average hop-count

Next experiment is done to measure the network lifetime (NLT). For this experiment, nodes are allocated 103500 units of power in the beginning of the simulation. From the experimental result shown in Figure 5.10 [3], it can be seen that,

[2]95% confidence intervals were calculated but left of the graphs so that the lines can be more easily differentiated. The confidence intervals are within 0.6% of the mean.

[3]95% confidence intervals were calculated but left of the graphs so that the lines can be more

there is no significant difference among the values of NLT with different schemes with low link failure probability. As link failure probability increases, NLT increases as fewer packets reach the nodes near the sink(s) due to link failure. Also the difference between the values are much higher with higher link failure probability. With high load and high link failure TPS-NextHR creates congestion in the network so fewer packets reach the nodes closer to the sinks. On the other hand, PrunedTreeSceduling can route packets to the nodes closer to the sinks. So NLT is lower for PrunedTreeSceduling compared to TPS-NextHR. For the same reason, other schemes that uses alternative routes have higher NLT compared to PrunedTreeSceduling.

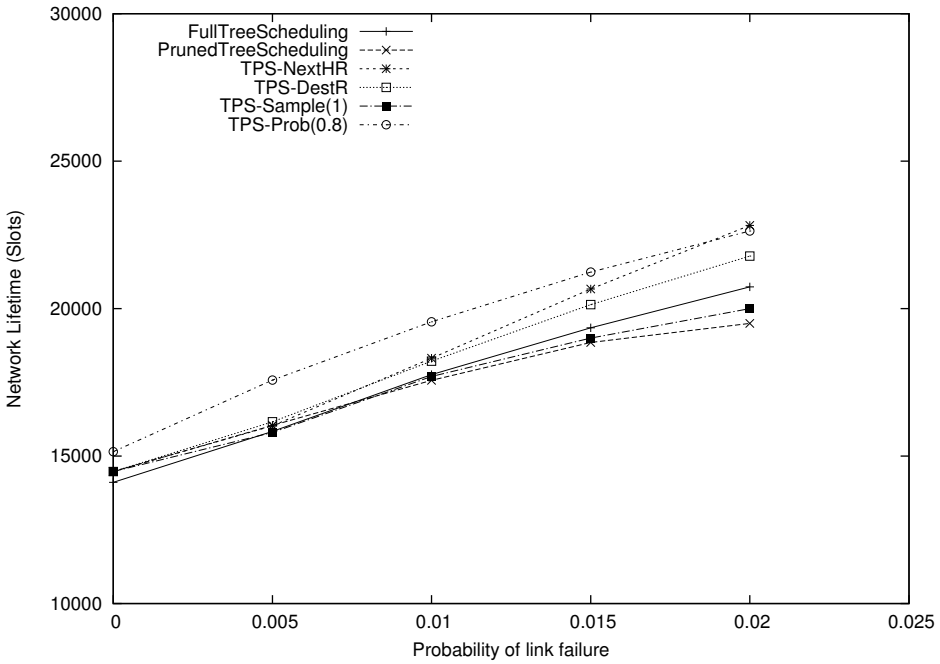


Figure 5.10: Network lifetime

To explore the effect of load on the network, we have run experiments changing the load of the network by changing the number of sources in the network. The number of sources selected for this experiment are 4, 8, 16 and 34. From the experiment, shown in Figure 5.11, it is found that increasing number of sources increase the delay to deliver all the packets to the sink(s) because we need more time to deliver higher number of packets. But as the load of the network decreases,

easily differentiated. The confidence intervals are within 2% of the mean.

the difference between performance of TPS-NextHR and PrunedTreeSceduling decreases. With low load, TPS-NextHR works similar to PrunedTreeSceduling as it can use alternative paths to deliver the packets quickly to th sinks. But with high load, TPS-NextHR creates congestion in the network and its performance degrades.

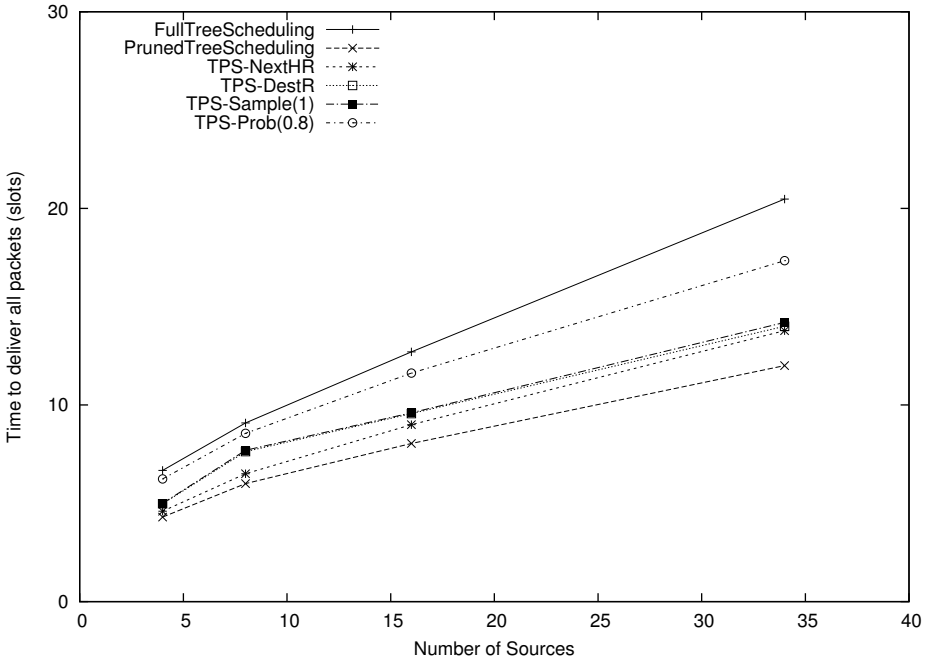


Figure 5.11: Effect of number of sources on average time to deliver packets

5.6 Chapter Summary

In this chapter, we have designed robust topologies with grid graphs for wireless sensor networks. In the design nodes can have horizontal, vertical and diagonal links and there are two sinks in the network. For each of the nodes, there are two link disjoint paths, one to each of the sinks. The topologies are designed in such a way that the path to the sinks from the nodes is shortest based on hop-count. In the logical topologies there are no 4-degree nodes and the number of 3-degree nodes is kept low. To examine the quality of the designed topology, we have done static analysis on them using failure models. Results show that the topology maintains a good degree of robustness and the topology can perform well under moderate link failure.

Also, we have performed a dynamic evaluation of our proposed logical topology using various routing protocols. The evaluation is done based on average time to deliver all packets, network lifetime and average hop-count . For the evaluation logical topologies with single path and multiple paths are chosen which uses different TDMA scheduling schemes. It is found that, with very high load routing schemes with multiple paths creates congestion in the network increasing delay. They also can not deliver high number of packets to the sinks so the network lifetime is also lower compared to PrunedTreeSceduling. But with lower load, their performance improves.

Chapter 6

Conclusion and Future Work

6.1 Conclusions

In this thesis, we have designed robust logical topologies with the presence of multiple sinks for wireless sensor networks. We have considered random sensor network graphs and grid graphs for physical deployment of sensor nodes, and with the presence of two sinks in the network, we have designed a process of creating logical topologies from the physical topology which supports robustness. The logical topology of a network provides probabilistic robustness in the presence of link failure because for each node in the network, there are two link disjoint paths, from that node to each of the two sinks in the network. We have also ensured that in the logical topology, the paths from each node to each of the sinks are shortest to ensure lowest cost communication in a failure free environment. In addition, for the routing tree of each of the sinks, the number of children of each node is constrained to minimize the hot-spot problem.

With random sensor network graphs, we take two degree-constrained shortest path trees rooted at the two sinks as the routing tree of each sink. We position the two sinks in such a way that the union of the two trees have two link disjoint paths for each of the nodes to each of sink. The union of the two trees acts as the logical topology for the network. For the grid graphs, we assume that the two sinks are positioned in the two opposite corners of the grid. We also assume that the nodes can communicate with their one hop distance nodes horizontally, vertically and diagonally. Then we add the links between the nodes in such a way that there

are no 4-degree node in the logical topology and the number of 3-degree nodes in the logical topology is $6V - 10$, where V is the minimum of the two dimensions of the grid.

To examine the probabilistic robustness of the proposed logical topologies, we have simulated our topologies with multiple link failures. The results show that the topologies offer a high degree of robustness and can tolerate moderate link failure.

We have designed TDMA scheduling algorithms that support two sinks. In order to examine the dynamic behavior of our proposed topologies, we have performed a dynamic evaluation of these topologies by simulating them with simple routing schemes. We have used multiple packet generating sources and multiple link failure and evaluated the performance of the routing schemes in terms of average time to deliver all packets produced by all sources, average hop-count and network lifetime. The results show that link acknowledgment with next hop routing (LinkAck-NextHR) scheme, which works in a multiple path logical graph can reliably transfer data compared to pruned tree scheme (PrunedTreeScheduling) which works in a single path logical graph with moderate link failure. Network lifetime is lower for for LinkAck-NextHR scheme compared to other schemes as it produces duplicate packets in the presence of link failures. The performance of the network also depends on how long the links fail and the load of the network. The longer the links fails, the higher is the delay to send the packets to the sinks. Again as the load of the network increase, the difference in the performance between LinkAck-NextHR and PrunedTreeScheduling decreases.

6.2 Future Work

In this thesis we have worked with multiple sinks but we have assumed that there are only two sinks in the network. It would be interesting to extend the number of sinks in the network, which can increase the reliability of the network and also help balance the load of the network. Extending the number of sinks creates various interesting problems. In case of random sensor network graphs, there is a problem of locating these sinks in the network in such a way that the union of their routing

tree would create a robust logical topology such that from each node there would be link disjoint paths, to each of the sinks. For grid graphs, the problem is to add links between the nodes in such a way that robustness is ensured as well as node degree constraint is maintained. It would be interesting to see if this type of topology can be designed in grid graphs without the presence of any 4-degree nodes and keeping the 3-degree nodes minimal. Next the problem is to design TDMA scheduling algorithm based on more than two sinks. As increasing the number of sinks, increases the number of links, it will eventually increase the number of constraints in the conflict graph. So it would be interesting to see the effect of this increase on the scheduling length.

Another interesting problem is to work with larger networks in case of random sensor network graphs. In the current thesis, the experiments are done with 30 and 40 nodes. It would be interesting to work on a larger network with 100 or 200 nodes. We have to measure the probability of getting robust logical graphs with suitable sink positions. Determining the sink positions in a large network is computationally more difficult because of the increased search space. Next problem is to check the effect of increased network size on scheduling length.

In this thesis, work has been done to design robust network topologies that can work reliably in the presence of link failure. Future work can be done to design logical topologies that can work with node failures. To work in the presence of node failure, the network must have, from each node, multiple node disjoint paths to each of the sinks in the network. In random sensor network graphs, the sink has to be positioned in such a way that the resulting logical topology ensures this property and for grid graphs, links have to be connected accordingly.

In dynamic simulation, we have used explicit acknowledgments. We can also use implicit acknowledgments so that the number of packets transmitted is reduced thus reducing energy consumption. It also has the added benefit that the size of the slot length can be decreased as nodes do not have to send and receive any acknowledgment packets, thus reducing delay. Also, we have assumed that all the nodes have a fixed energy and we have defined network lifetime as the number of slots before the first node dies because of the exhaustion of it's energy. We can also as-

sume that each node have some energy recharging mechanism with which a certain amount of energy is regained periodically. An analysis on the network lifetime can be done with this added assumption.

In this thesis, we have only worked with converge-cast where no message aggregation is node in the intermediate nodes during data transmission from the nodes to the sinks. Future work can be done with aggregation converge-cast where each intermediate node aggregates data received from children with its own data before sending it to the parent. We need to design algorithms for finding minimal length TDMA schedule for aggregation converge-cast with multiple sinks. Also dynamic analysis have to be done using the schedule to examine the efficiency of the logical topologies in aggregation converge-cast.

Bibliography

[1] J. Polastre R. Szewczyk A. Mainwaring, D. Culler and J. Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, WSNA '02, pages 88–97, New York, NY, USA, 2002. ACM.

[2] J. Agre and L. Clare. An integrated architecture for cooperative sensing networks. *Computer*, 33:106–108, May 2000.

[3] O.B. Akan and I.F. Akyildiz. Event-to-sink reliable transport in wireless sensor networks. *Networking, IEEE/ACM Transactions on*, 13(5):1003 – 1016, October 2005.

[4] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38:393–422, 2002.

[5] S Bhatnagar B. Deb and B. Nath. A topology discovery algorithm for sensor networks with applications to network management, 2002.

[6] S. Bhatnagar B. Deb and B. Nath. Information assurance in sensor networks. In *Proceedings of the 2nd ACM international conference on Wireless sensor networks and applications*, WSNA '03, pages 160–168, New York, NY, USA, 2003. ACM.

[7] B. Bai, L. Stewart, and J. Harms. Degree-constrained shortest path tree and its application to multicast protocols. University of Alberta.

[8] J. Burrell, T. Brooke, and R. Beckwith. Vineyard computing: Sensor networks in agricultural production. *IEEE Pervasive Computing*, 3:38–45, 2004.

[9] Y. Chen, E. Chan, and S. Han. Energy efficient multipath routing in large scale sensor networks with multiple sink nodes. In Jiannong Cao, Wolfgang Nejdl, and Ming Xu, editors, *Advanced Parallel Processing Technologies*, volume 3756 of *Lecture Notes in Computer Science*, pages 390–399. Springer Berlin / Heidelberg, 2005.

[10] A. Das and D. Dutta. Data acquisition in multiple-sink sensor networks. *SIGMOBILE Mobile Computing Communication Review*, 9:82–85, July 2005.

[11] B. Deb, S. Bhatnagar, and B. Nath. Reinform: reliable information forwarding using multiple paths in sensor networks. In *Local Computer Networks, 2003. LCN '03. Proceedings. 28th Annual IEEE International Conference on*, pages 406 – 415, October 2003.

[12] P. Djukic and S. Valaee. Link scheduling for minimum delay in spatial re-use tdma. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 28 –36, May 2007.

[13] H. Dubois-Ferriere, D. Estrin, and T. Stathopoulos. Efficient and practical query scoping in sensor networks. In *Mobile Ad-hoc and Sensor Systems, 2004 IEEE International Conference on*, pages 564 – 566, October 2004.

[14] S. Dulman, T. Nieberg, J. Wu, and P. Havinga. Trade-off between traffic overhead and reliability in multipath routing for wireless sensor networks. In *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, volume 3, pages 1918 –1922 vol.3, March 2003.

[15] S. Ergen and P. Varaiya. Tdma scheduling algorithms for wireless sensor networks. *Wireless Networks*, 16:985–997, 2010. 10.1007/s11276-009-0183-0.

[16] S Nandy G. Das, S. Das and B. Sinha. Efficient algorithm for placing a given number of base stations to cover a convex region. *Journal of Parallel and Distributed Computing*, 66(11):1353 – 1358, 2006.

[17] S. Gandham, M. Dawande, and R. Prakash. Link scheduling in sensor networks: distributed edge coloring revisited. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 4, pages 2492 – 2501 vol. 4, march 2005.

[18] S. Oh Y. Yim S. Kim H. Park, J. Lee and K. Nam. Quality-based event reliability protocol in wireless sensor networks. In *Consumer Communications and Networking Conference (CCNC), 2011 IEEE*, pages 730 –734, January 2011.

[19] J. Hart and K. Martinez. Environmental sensor networks: A revolution in the earth system science? *Earth-Science Reviews*, 78(34):177 – 191, 2006.

[20] J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):pp. 100–108, 1979.

[21] Y.T. Hou, Yi Shi, H.D. Sherali, and S.F. Midkiff. Prolonging sensor network lifetime with energy provisioning and relay node placement. In *Sensor and Ad Hoc Communications and Networks, 2005. IEEE SECON 2005. 2005 Second Annual IEEE Communications Society Conference on*, pages 295 – 304, September 2005.

[22] Y. Wu X. Li J. Ma, W. Lou and G. Chen. Energy efficient tdma sleep scheduling in wireless sensor networks. In *INFOCOM 2009, IEEE*, pages 630 –638, april 2009.

[23] Z. Wu J. Mao and X. Wu. A tdma scheduling scheme for many-to-one communications in wireless sensor networks. *Comput. Commun.*, 30:863–872, February 2007.

[24] V. Padmanabhan K. Jain, J. Padhye and L. Qiu. Impact of interference on multi-hop wireless network performance. *Wireless Networks*, 11:471–487, 2005. 10.1007/s11276-005-1769-9.

[25] H. Kang and X. Li. Power-aware sensor selection in wireless sensor networks. In *In The 5th International Conference on Information Processing in Sensor Networks, Work-in-Progress*, 2006.

[26] S. Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129 – 137, March 1982.

[27] P. Nimbhorkar M. Mahajan and L. Varadarajan. The planar k-means problem is np-hard. In *Proceedings of the 3rd International Workshop on Algorithms and Computation*, WALCOM '09, pages 274–285, Berlin, Heidelberg, 2009. Springer-Verlag.

[28] M. Mahmood and K. Seah. Event reliability in wireless sensor networks. In *Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2011 Seventh International Conference on*, pages 377 –382, December 2011.

[29] S. Mukhopadhyay, C. Schurgers, D. Panigrahi, and S. Dey. Model-based techniques for data reliability in wireless sensor networks. *Mobile Computing, IEEE Transactions on*, 8(4):528 –543, April 2009.

[30] C. Murthy and B.S. Manoj. *Ad Hoc Wireless Networks: Architectures and Protocols*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2004.

[31] E.I. Oyman and C. Ersoy. Multiple sink network design problem in large scale wireless sensor networks. In *Communications, 2004 IEEE International Conference on*, volume 6, pages 3663 – 3667 Vol.6, June 2004.

[32] J. Pan, L. Cai, Y.T. Hou, Y. Shi, and S.X. Shen. Optimal base-station locations in two-tiered wireless sensor networks. *Mobile Computing, IEEE Transactions on*, 4(5):458 – 473, September-October 2005.

[33] M. Perillo, M. Cheng, and W. Heinzelman. On the problem of unbalanced load distribution in wireless sensor networks. In *Global Telecommunications Conference Workshops, 2004. GlobeCom Workshops 2004. IEEE*, pages 74 – 79, November-December 2004.

[34] G.J. Pottie. Wireless sensor networks. In *Information Theory Workshop, 1998*, pages 139 –140, June 1998.

[35] Y. Guo R. Bai, Y. Qu and B. Zhao. An energy-efficient tdma mac for wireless sensor networks. In *Asia-Pacific Service Computing Conference, The 2nd IEEE*, pages 69 –74, December 2007.

[36] A.A. Reza. *Robust grid-based deployment schemes for underwater optical sensor networks*. University of Alberta (Canada), 2008.

[37] Akyildiz I.F. S. Park, R. Sivakumar and R. Vedantham. Garuda: Achieving effective reliability for downstream communication in wireless sensor networks. *Mobile Computing, IEEE Transactions on*, 7(2):214 –230, February 2008.

[38] IEEE Computer Society. Ieee standard for information technology - telecommunications and information exchange between systems - local and metropolitan area networks - specific requirements - part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications. *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)*, pages 1 –1076, 12 2007.

[39] M. Tahani and M. Sabaei. A distributed data-centric storage method for hot spot mitigation in wireless sensor networks. In *Telecommunications (IST), 2010 5th International Symposium on*, pages 401 –408, December 2010.

[40] I. Vasilescu, K. Kotay, D. Rus, M. Dunbabin, and P. Corke. Data collection, storage, and retrieval with an underwater sensor network. In *Proceedings of the 3rd international conference on Embedded networked sensor systems*, SenSys '05, pages 154–165, New York, NY, USA, 2005. ACM.

[41] Z. Vincze, R. Vida, and A. Vidacs. Deploying multiple sinks in multi-hop wireless sensor networks. In *Pervasive Services, IEEE International Conference on*, pages 55 –63, July 2007.

[42] S. Jha W. Hu, C. Chou and N. Bulusu. Deploying long-lived and cost-effective hybrid sensor networks. *Ad Hoc Networks*, 4(6):749 – 767, 2006.

[43] C. Wan, T. Campbell, and L. Krishnamurthy. Psfq: a reliable transport protocol for wireless sensor networks. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, WSNA '02, pages 1–11, New York, NY, USA, 2002. ACM.

[44] T. Wang, Z. Wu, and J. Mao. Pso-based hybrid algorithm for multi-objective tdma scheduling in wireless sensor networks. In *Communications and Networking in China, 2007. CHINACOM '07. Second International Conference on*, pages 850 –854, August 2007.

[45] W. Wang, Y. Wang, X. Li, W. Song, and O. Frieder. Efficient interference-aware tdma link scheduling for static wireless networks. In *Proceedings of the 12th annual international conference on Mobile computing and networking*, MobiCom '06, pages 262–273, New York, NY, USA, 2006. ACM.

[46] Y. Wang and I. Henning. A deterministic distributed tdma scheduling algorithm for wireless sensor networks. In *Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on*, pages 2759 –2762, September 2007.

[47] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh. Deploying a wireless sensor network on an active volcano. *Internet Computing, IEEE*, 10(2):18 – 25, March-April 2006.

[48] A. Willig and H. Karl. Data transport reliability in wireless sensor networks a survey of issues and solutions, praxis der informationsverarbeitung und kommunikation, 2005.

[49] X. Cai Y. Lin, Q. Wu and N.S.V Rao. Optimizing base station deployment in wireless sensor networks under one-hop and multi-hop communication models. In *Parallel and Distributed Systems (ICPADS), 2009 15th International Conference on*, pages 96 –103, December 2009.

[50] O. Yang and W. Heinzelman. Sleeping multipath routing: A trade-off between reliability and lifetime in wireless sensor networks. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pages 1 –5, December 2011.

[51] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient mac protocol for wireless sensor networks. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1567 – 1576 vol.3, 2002.

[52] Chang Wu Yu, Shan Shiung Wei, and Bing Jiun Shie. Activating the hot spot problem by switching multiple sinks in wireless sensor networks. In *Mobile Ad-hoc and Sensor Networks, 2009. MSN '09. 5th International Conference on*, pages 204 –211, dec. 2009.

[53] K. Yuen, B. Liang, and L. Baochun. A distributed framework for correlated data gathering in sensor networks. *Vehicular Technology, IEEE Transactions on*, 57(1):578 –593, January 2008.

[54] Y. Zhang, S. Zheng, and S. Xiong. A scheduling algorithm for tdma-based mac protocol in wireless sensor networks. In *Education Technology and Computer Science, 2009. ETCS '09. First International Workshop on*, volume 3, pages 148 –151, March 2009.

Appendix A

Sink Location from Sensor Node Position

A.1 Algorithms

In this scheme, all the current node positions are considered as possible sink positions. We have tried several heuristics to find two suitable sink positions so that the resultant logical topologies are 2 link-disjoint sink-connected. If we can find two suitable locations, we have to put the two sinks in those two positions replacing the two nodes. The heuristics to find the locations are:

- **Random**: The random method is tried to compare other heuristics with a purely random sink selection. In this method, two nodes are chosen at random from potential DCSPT roots as sinks and, considering those as cluster heads two clusters are created by nodes connecting to the nearest sink.

- **KMeans Clustering**: K-Means clustering is a classic algorithm to find sink locations in the network. It is a method which tries to partition n nodes into k clusters with k cluster heads, such that each node belongs to the cluster with nearest cluster head [20]. This partitions the n nodes into Voronoi cells. In networking, the cluster heads are selected as the sinks and the nodes report to its nearest sink. The goal is to find cluster heads in such a way that the sum of the distances between the nodes in a cluster and the respective cluster head is minimized. This would result in minimizing the communication cost from the nodes to the sinks. This problem is NP-hard [27] but many heuristics

have been suggested to find the solution. The standard algorithm for KMeans clustering is given by Lloyd [26].

In this thesis, we have used a small modification of this algorithm to find the two sinks in the network. In this method there are three steps, selection, assignment and update. In the selection step, two nodes from n nodes are selected randomly as cluster heads. In the assignment step, the rest of the nodes are assigned to the clusters, in such a way that a node is assigned to the closest cluster-head. Our algorithm differs from the standard algorithm on how the update step works. In the standard algorithm, the midpoint of the cluster, which is called the centroid, is chosen as the new cluster-head in a cluster. This ensures that, within the cluster, the distance between the nodes and the cluster head is minimized. The algorithm proceeds by alternating between the assignment and the update step until the assignments no longer changes. As it is a heuristic algorithm, there is no guarantee that the algorithm will converge to global optimum and the result may depend on initial cluster. So the algorithm is repeated with multiple initial assignments to avoid getting stuck into local minima.

In our algorithm, we cannot consider the midpoint of the cluster as the new cluster head because we are considering the existing node positions as the sink positions. In our algorithm, we check every node in the cluster as the potential cluster-head and chose a node as cluster-head, such that the sum of the distances between the node and the rest of the nodes in the cluster is minimum. Here the algorithm is run with 10 initial assignments to avoid local minima. After all the repeats, the pair of sink locations which resulted in the minimum sum of distance from the nodes to their cluster-heads is returned.

The goal of the KMeans clustering is to minimize the distance between the nodes to the sinks but it does not consider that the resulting logical topology should be reliable or the logical topology should be 2 link-disjoint sink-connected.

- **Farthest**: We want to obtain a 2 link-disjoint sink-connected graph as the

logical topology, so we put the sinks far away from each other in this scheme. Then, the two paths from a node to the two sinks are less likely to have common links. In this method, one node from the graph is chosen as the first sink at random from the potential DCSPT roots and then another node is chosen as the second sink from the leaves of the first sink's DCSPT, which has the highest hop-count from the first sink. These are considered as cluster heads and two clusters are created by nodes connecting to the nearest sink.

- **Edge-far**: The farthest method tries to select two sinks that have highest distance between them. But in this method, if the first sink is selected from the middle of the network then the method may not chose two sinks with highest distance in the network. So, in the Edge-far method, we try to maximize the distance between the two nodes. Let us define the edge of a network. We assume that the network is square with nodes positioned on grid points. The edge is defined as the nodes around the perimeter of the network. In this thesis, the network is a 25×25 grid, so there are 25 columns and 25 rows. In this method, one node from the edge of the graph is chosen at random from the potential DCSPT roots as the first sink, and then another node is chosen as the second sink from the leaves of the first sink's DCSPT, which has the highest hop-count from the first sink. These are the cluster heads and nodes connect to the nearest sink.

A.2 Experimental Method

We have run experiments to compare the effectiveness of each of the heuristics. For the experiments, we have 50 random topologies with 30 nodes in a 25×25 grid space. All the nodes in the topologies have a range $r = 6$. We have run the DCSPT algorithm on each of the physical topologies using node degree constraint 5, sink degree constraint 5 and obtained all the possible sinks for each topology. Then using each of the heuristics, we have selected two sinks out of all the possible sinks. Taking the union of the two degree-constrained shortest path trees, called routing trees, we have obtained the logical topology from each physical topology

using the heuristics previously discussed. Then properties of the logical topology are analyzed.

The metrics used for comparing the various schemes are average Hop-Count, average Alternative Hop-Count and Number of Cut-points. Hop-Count is the measure of distance from each node to its closest sink. This is averaged over all nodes. The Alternative Hop-Count is the measure of distance to the farthest sink and it is also averaged over all nodes. These are the measures of path quality. A cut-point is a node whose deletion increases the number of connected components in a graph. The number of cut-points are measures of degree of reliability. In these experiments we consider the sink pair to be **eligible** for building a logical topology if the union of the two trees is 2 link-disjoint sink-connected. Both eligible and ineligible results are shown in the performance graphs. All the results shown are the average of 100 trials, each run on a separate topology and 95% confidence intervals are also calculated.

A.3 Experimental Results

In Table A.1, the proportion of eligible logical topologies is shown. From the experiments it can be seen that both Farthest and Edge-far have significantly better chance to find robust topologies. This happens because, in these methods, we are choosing the two sinks which are far apart. From Figure A.2, it can be seen that, the number of cut-points are also much less for these heuristics compared to the other heuristics. So this heuristic provides us with more reliable topologies. The only drawback of these schemes is, they tend to increase the Hop-Count as seen in Figure A.1. This happens because, in KMeans, the goal is to minimize the distances from the sinks to the nodes but in the other schemes the goal is to separate the sinks which increases the distances from the nodes to the sinks.

Random	KMeans	Farthest	Edge-far
7	11	43	49

Table A.1: Comparison between different sink location schemes - eligible topologies (out of 100)

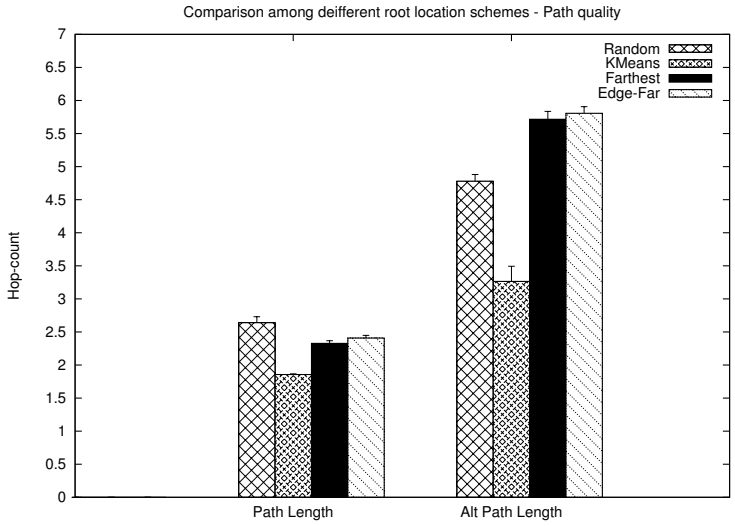


Figure A.1: Comparison between different sink location schemes- path quality.

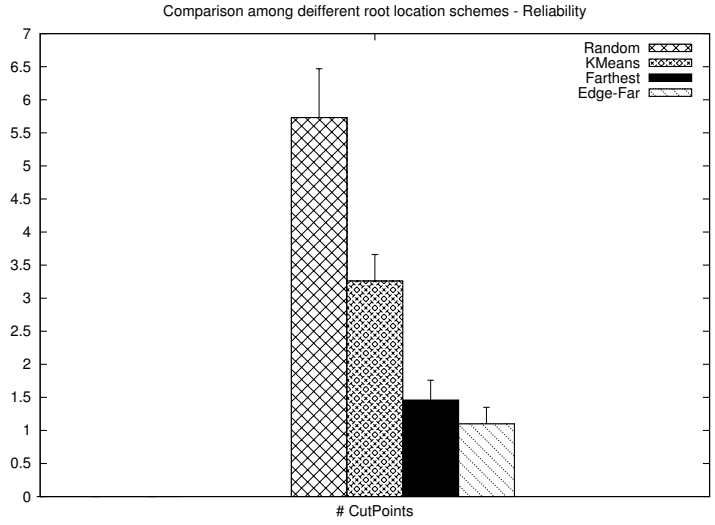


Figure A.2: Comparison between different sink location schemes- reliability.

Choosing the second sink from the highest hop-count distance has better probability of obtaining an eligible sink pair than the KMeans or Random schemes; while KMeans results in lower average Hop-Count. This is because KMeans tries to decrease distance between the nodes and the sinks but it also makes one sink dependent on the other. The less dependent a sink is on the other sink to reach other nodes, the better probability we have to obtain an eligible sink pair. Analyzing the unsuccessful cases reveals that the sinks are very close, sometimes neighbors. There are many common links in the two DCSPTs and therefore two sinks are very much dependent on each other.

When we use the Farthest or Edge-far method for selecting the two sinks, we ensure that the second sink is a leaf of the DCSPT of first sink. The other methods do not hold this property and the probability of obtaining an eligible pair is higher for the Farthest and Edge-far method. From this we can assume that if one sink is not a leaf of the DCSPT rooted at the other sink, the probability of obtaining an eligible pair decreases. Here we will show that if $r2$ is not a leaf of $r1$ then the probability of getting a 2 link-disjoint sink-connected graph from the union of the two trees decreases. Suppose $r2$ is not a leaf in $r1's$ DCSPT. It has the child v in $r1's$ DCSPT. By construction, the DCSPT of sink $r1$, holds the shortest path from $r1$ to v. So the shortest path goes through $r2$. And as v is a child of $r2$, the shortest path from $r2$ to v is the one hop link $(r2, v)$. So the two shortest paths have a common link $(r2, v)$. If we delete the link $(r2, v)$ then v is disconnected from both the sinks and the logical topology will not be a 2 link-disjoint sink-connected graph. Let the length of path from $r1$ to v through $r2$ be d. The only way to make sure that two shortest paths do not have a common link is to have a path from $r1$ to v with length d that does not include $r2$. But in $r1's$ DCSPT, there is only one path from $r1$ to v. So if the second path from $r1$ to v, avoiding $r2$ exist then the link $(r2, v)$ does not exist. So v is not a child of $r2$ in $r1's$ DCSPT. So $r2$ is a leaf. Similarly, $r1$ is a leaf is $r2's$ DCSPT. So in an eligible logical topology, one sink is in the leaf of another sink's DCSPT.

A.3.1 Comparsion with Edge-position and Edge-far scheme

We have done an experiment to compare Edge-far scheme which is the best heuristics so far and Edge-position scheme which is designed in Chapter 3. By limiting the sink degree to 5, it is found that the Edge-position scheme has better probability to find an eligible sink pair location than the Edge-far method. This happens because, by separating the sinks from the nodes we eliminate the dependency of one sink on another so the probability of finding link-disjoint paths increases. The comparison between these two schemes is given in Table A.2.

Node Degree	Edge-position	Edge-far
4	36	13
5	45	25

Table A.2: Comparison between Edge-far and Edge-position by possibility of getting eligible pair (out of 50), sink degree constraint = 5

Appendix B

Algorithms for Building and Coloring Conflict Graphs

The algorithms for building and coloring conflict graphs are described here. The notations followed in the algorithms are listed in Table B.1

Symbol	Meaning
V	Nodes in the physical topology
E	Links between two nodes in physical topology
E'	Links between two nodes in the routing tree
E'_i	Links between two nodes in the i^{th} routing tree
V'	Nodes of conflict graph, represents the links E'
E''	Edges of conflict graph, represents the conflicts
$v1$	First parent of node u
$v2$	Second parent of node u

Table B.1: Notations Used in the Algorithms

Input: E', E
Output: Conflict Graph $G_C = (V', E'')$

begin
  $V' \leftarrow \emptyset$;
  $E'' \leftarrow \emptyset$;
  for *each* $e = (u, v) \in E'$ **do**
    $v_e \leftarrow (u, v)$;
    for *each* $v'_e = (u', v') \in V'$ **do**
      if $(u = v' \vee v = u' \vee u = u' \vee v = v' \vee (u', v) \in E \vee (u, v') \in E)$ **then**
        $E'' \leftarrow E'' \cup (v_e, v'_e)$;
    $V' \leftarrow V' \cup v_e$;
end

Algorithm 1: Building the conflict graph from one tree

Input: E'_1, E'_2, N
Output: Conflict Graph $G_C = (V', E'')$

begin
  $V' \leftarrow \emptyset$;
  $E'' \leftarrow \emptyset$;
  for *each* $e = (u, v) \in (E'_1 \cup E'_2)$ **do**
    $v_e \leftarrow (u, v)$;
    for *each* $v'_e = (u', v') \in V'$ **do**
      if $(v = u' \vee u = v' \vee u = u' \vee v = v' \vee (u', v) \in E \vee (u, v') \in E)$ **then**
        $E'' \leftarrow E'' \cup (v_e, v'_e)$;
    $V' \leftarrow V' \cup v_e$;
end

Algorithm 2: Building the conflict graph from two pruned trees

Input: E'_1, E'_2, N
Output: Conflict Graph $G_C = (V', E'')$

begin
  $V' \leftarrow \emptyset$;
  $E'' \leftarrow \emptyset$;
  for *each* $(e1 = (u, v1) \in E'_1) \wedge (e2 = (u, v2) \in E'_2)$ **do**
    $v_e \leftarrow (u, v1, v2)$;
    for *each* $v'_e = (u', v1', v2') \in V'$ **do**
      if $((v1 = u' \vee v2 = u') \vee (u = v1' \vee u = v2') \vee ((u', v1) \in E \vee (u', v2) \in E) \vee ((u, v1') \in E \vee (u, v2') \in E)$ **then**
        $E'' \leftarrow E'' \cup (v_e, v'_e)$;
    $V' \leftarrow V' \cup v_e$;
end

Algorithm 3: Building the conflict graph from two trees, where for each sender there are two receivers

Input: Conflict Graph $G_C = (V', E'')$
Output: A color assigned to each node

begin
Order the nodes in descending order of their degree $(n_1, n_2.....n_M)$, M number of nodes;
for $l \leftarrow 1$ **to** M **do**
$i \leftarrow 1$;
while $\exists j$ *assigned to color* i *such that* $(j, n_l) \in E''$ **do**
$i \leftarrow i + 1$;
assign color i to n_l;
end

Algorithm 4: Coloring algorithm for conflict graph [15]