CONTROLLED MOBILITY FOR PERFORMANCE ENHANCEMENTS IN WIRELESS SENSOR NETWORKS

By

Jayanthi Rao

A DISSERTATION

Submitted to
Michigan State University
In partial fulfillment of the requirements
For the degree of

DOCTOR OF PHILOSOPHY

Electrical Engineering

2009

ABSTRACT

CONTROLLED MOBILITY FOR PERFORMANCE ENHANCEMENTS IN WIRELESS SENSOR NETWORKS

By

Jayanthi Rao

The objective of this thesis is to investigate the role of controlled node mobility for enhancing data collection and energy performance in wireless sensor networks. The goal is to develop joint mobility and packet routing protocols, and to study the associated tradeoffs from a network design standpoint. An understanding of these tradeoffs help in formulating design guidelines that can be applied to network deployments. Specific contributions of the thesis are as follows.

First, for networks with mobile sensors and static base stations, a new paradigm for extending network life has been explored by introducing energy-aware node mobility along with an adaptive packet routing protocol. A fully distributed and localized mobility control and packet routing framework for collective extension of network lifetime was developed. Via analytical modeling and simulation experiments it was established that using cooperative and energy-aware node mobility, it is feasible to significantly extend a sensor network's operating life even in situations where the energy cost of physical node movement is modeled as high as up to four orders of magnitude larger than the energy cost for packet transmissions. The investigated paradigm in this part of the thesis applies to networks with mobile sensors such as terrestrial and aquatic robotic micro-vehicles with sensing abilities.

The second part of the thesis deals with networks with static sensors and mobile sinks which correspond to applications such as Unmanned Air/Ground Vehicles (UAV, UGV) based data collection from unattended static sensor units. In this part of the thesis, a generalized analytical formulation for mobile sensor based data collection has been developed with varying levels of static sensor to mobile sink multi-hop routing, which is parameterized as a hop-bound factor *k*. The key concept is to appropriately adjust the extent of multi-hop routing and the corresponding sink trajectory by varying the hop bound factor *k*. The main result is that for given sensor data rates and node density, there are clear bounds to the extent of multi-hop routing that can be sustained in a network. In other words, to maintain data collection stability from delay and capacity standpoints, the extent of multi-hop routing (hop-bound factor *k*) needs to be within a feasible range – rendering data collection with static sink often infeasible.

This result was experimentally validated by developing a Network-Assisted Data Collection (NADC), which is a distributed and location-unaware sink navigation and data routing protocol. Finally, the NADC framework was extended for networks with spatially heterogeneous data generation rates, and the routing feasibility results were validated for such scenarios.

To My Family

For All Their

Love and Support

Acknowledgements

I owe my deepest gratitude to my advisor Dr. Subir Biswas for guiding and supporting me throughout the course of my doctoral study. His ideas, expertise and sound advice have been a major influence on this thesis. I have learned a great deal under his able mentorship during this research work. I would also like to thank the other members of my committee namely Dr. Nihar Mahapatra, Dr. Erik Goodman and Dr. Betty Cheng for their helpful suggestions, support and advice.

I am greatly indebted to Dean Udpa, Dr. Reinhard, Dr. Shanblatt, Dr. Barbara O'Kelly, ECE department and the graduate school for their support and encouragement in my endeavor. Also, this research study required long and extensive simulations and would not have been possible without the availability of HPCC computing resources.

Very special thanks go to my colleagues and friends in the NeEWS laboratory - Muhannad Quwaider, Anthony Plummer, Mahmoud Taghizade, Tao Wu and Fan Yu with whom I have worked closely during the last few years and enjoyed sharing research ideas and accomplishments.

Lastly, and most importantly, I thank my family for their understanding, motivation and support throughout this research work.

TABLE OF CONTENTS

List of Tables

List of Figures

List of Abbreviations

CDC	Continuous Data Collection
COM	Collect-on-Move
CSA	Centralized Swapping Algorithm
DG	Designated Gateway
EAR	Energy Aware Routing
EASP	Energy Aware Swap Protocol
EDR	Energy Drainage Rate
IN	Intermediate Navigation agent
LATD	λ-Aware Trajectory Determination
MAPC	Mobility Assisted Power Control
MDH	Mobile Data Harvester
MST	Minimum Spanning Trajectory
NA	Navigation Agent
NADC	Network Assisted Data Collection
PDC	Periodic Data Collection
RN	Run-time Navigation
RW	Random Walk
SCAV	Swarm Capable Autonomous Vehicle
SDRE	Standard Deviation of Remaining Energy
SOD	Stop-on-Demand
SPR	Shortest Path Routing
SPR	Shortest Path Routing
TSP	Travelling Salesman Problem

UAV	Unmanned Aerial Vehicle
UGV	Unmanned Ground Vehicle

Chapter 1. Introduction

1.1 Wireless Sensor Networks

Developments in wireless and mobile communications combined with the constant advancements in electronics that enable the integration of complex components into smaller devices, have contributed to the emergence of a new class of wireless ad hoc networks – sensor networks. A sensor network consists of a large number of small, low-cost devices with sensing, processing, and transmitting capabilities. A sensor board mainly consists of one or more sensors, a microprocessor and a low-power radio transceiver. The on-board sensors may be motion detectors, thermistors, light sensors, microphones, accelerometers, magnetometers, humidity and barometer sensors, GPS receivers, etc. Sensor networks are expected to be used for a wide variety of applications such as surveillance, habitat and environmental monitoring, target tracking, search and rescue, disaster relief and so on.

Sensor networks [1-3] are expected to be used in regions where there is no infrastructure support. Sensors are usually deployed in large numbers often in inaccessible or hostile regions. Hence, they are expected to operate untethered and unattended for long periods of time. Sensor nodes are powered by batteries and therefore, they tend to be extremely energy constrained. Replacing the batteries or recharging them when the network is in operation can be very expensive or impractical depending on the terrain. For this reason, energy efficiency and network lifetime elongation are important considerations while designing protocols for sensor networks. Energy consumption for communication is several orders of magnitude higher than the energy required for computation [4]; and wireless communication is foreseen to continue to dominate the energy consumption in a sensor node in the near future. To address this, a lot of research effort has centered on minimizing the amount of communication that is required during sensor network operations.

Protocols that have been developed for traditional ad hoc networks may not be suitable for sensor networks. This is because sensor networks present certain unique differences and challenges that require specific design efforts[2, 5].

Application driven requirements: Sensor networks are expected to be deployed for very specialized and specific applications such as environmental/habitat monitoring, public facility monitoring, health monitoring, target tracking, battle-field surveillance and intrusion detection, which all benefit from combination of sensing, computing and communication technology. Each application is likely to have different performance objectives. Providing bounded message latency and high delivery ratio may be crucial in certain timing constraint applications such as intrusion detection and disaster warning. However, for some other applications like environment monitoring it is enough to have only occasional packet delivery but long lifetime.

Resource Constraints: Sensor nodes are powered by battery and replacement is usually impossible in many applications. Thus, energy supply is scarce and energy consumption is a primary consideration for wireless sensor network protocol and algorithm design. Since communication is significantly more expensive than computation [4], reducing energy dissipation used in communication becomes very important. Typically the non-essential energy consumptions are contributed from four sources. (1) Protocol overhead: besides valid data certain control messages have to be exchanged to access communication resources; (2) Message collision: unnecessary additional energy is wasted by retransmitting as well as possible duplicate receiving same message; (3) Overhearing: sensor nodes spend energy to receive messages for which they are not intended receivers because that message transmission usually is broadcast within WSN; (4) Idle listening: when there is no centralized scheduler to manage sequence of transmission, sensor nodes have to keep the wireless radio interface up (power on) for possible message receptions. Even when sensor nodes do not transmit or receive any message the consumed energy can be significant. Researchers [4, 6] have shown that idle listening actually accounts for most of the energy expenditure in low traffic situations, which is quite common for most applications in WSN.

Mobility: Although wireless sensor networks are often considered as stationary, some applications may require one or several powerful mobile nodes to achieve better performance or other purpose [7, 8]. Some applications like mapping[9, 10], mine-removal [11], etc., require mobile sensors that are capable of moving to accomplish their primary mission. Mobility may be random (ex. animals [12, 13], humans), predictable (ex. public transport vehicle) [14] or controlled by the network (ex. robotic sensors [15]).

Mobile sensor networks or robotic sensor networks [16] [17, 18] have enormous potential for positive impact in our society. They can be employed for search-and-rescue operations in emergency situations. In environmental monitoring, they can be used to detect forest fires or monitor rare species. In health care, they can monitor patients or the elderly over extended periods of time without confining them to a small area. A number of automation tasks, such as surveillance, monitoring energy consumption and quality inspection, can benefit from robotic sensor network technology.

Traffic Patterns: Unlike communication in traditional ad hoc networks based on point-to-point connections, sensor networks mainly use a broadcast communication paradigm and almost all applications of sensor networks require the flow of sensed data from multiple sources to a particular base station. This, however, does not prevent the flow of data from being in other forms (e.g., multicast or peer to peer). The reason that all messages are sent to all the radio neighbors is because a single sensor node lacks global knowledge of the whole network. The traffic in the sensor networks can be continuous, event-driven, query-driven, or hybrid based on applications. A low data traffic rate over a large time scale but with some very bursty traffic sometimes can be quite a common situation in the wireless sensor networks.

Data Generation: Sensor nodes may generate data in the following manner.

a. **Continuous**: Data is generated continuously and routed to the sink. For instance, habitat monitoring requires sensors to monitor physical parameters such as temperature, humidity, etc., of a geographical region periodically and route that information to a base station.

b. **Event-driven**: Data generation by a sensor is triggered by an event of interest. Only sensors around the event generate data that has to be sent to the base station.

c. **Demand-driven**: Data is sent to the sink in response to an explicit request that is propagated into the network in the form of a query [19].

Data Centric: The random deployment of sensor nodes often introduces a redundant nodal distribution. The redundancy is desirable for reliable coverage of the sensor field with low-cost and low-energy sensor nodes. Moreover, data collected by many sensors in WSN is typically based on common phenomena, so there is a high probability that this data has some redundancy. Such redundancy needs to be exploited by the routing protocols. Attribute-based naming for example, can be used to improve both energy and bandwidth utilization. In WSN, sometimes getting the data pertaining to sensor node observation is more important than knowing the IDs of which nodes sent the data. As a result, the importance of an individual node is reduced in contrast to traditional networks. This changing of role makes the design change from a node-centric to a data-centric architecture.

Location Awareness: Position awareness of sensor nodes is important since data collection is normally based on the location. Currently, it is not feasible to use Global Positioning System (GPS) hardware for this purpose. Methods based on triangulation [20], for example, allow sensor nodes to approximate their positions using radio strength from a few known points. According to [20], algorithms based on triangulation or multilateration can work quite well under conditions where only very few nodes know their positions a priori. Still, it is preferable to have non-GPS solutions [21] for the location problem in WSN or to design protocols/algorithms that are location independent.

1.2 Applications of Wireless Sensor Networks

Initially, sensor networks were motivated by defense applications such as intrusion detection, battlefield surveillance and object tracking. Since then, there has been a rapid increase in the number of civilian applications like environment data collection, health monitoring, vehicular sensor networks, disaster relief and the like.

Sensors are expected to be deployed in large numbers for monitoring applications. These applications can be broadly classified as *continuous monitoring* and *event monitoring*. Continuous monitoring applications are typically used for scientific research and require nodes to generate sensing data periodically over a large period of time. Using sensor networks enables data to be monitored at a scale and resolution that is difficult to achieve using traditional monitoring technologies. The data is used to analyze long term trends and spot interdependencies. On the other hand, event monitoring applications are intended to detect the occurrence of pre-defined events in the sensing environment, for instance, detecting intrusion, temperature below a certain threshold, etc. Therefore, sensors may sense their environment periodically, but they do not constantly generate data destined for the sink.

1.2.1 Continuous Monitoring

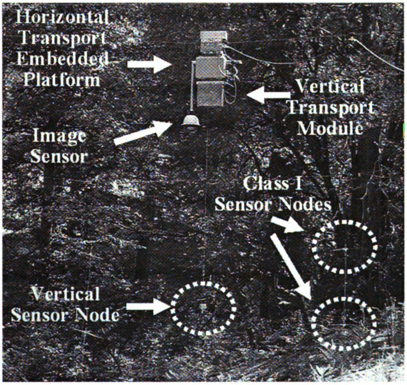

Horizontal Transport Embedded Platform
Vertical Transport Module
Image Sensor
Class I Sensor Nodes
Vertical Sensor Node

Figure 1-1: NIMS deployment in James San Jacinto Mountains Reserve (source: http://research.cens.ucla.edu/areas/2005/NIMS)

There are a number of projects[13, 22, 23] that have deployed sensor networks for continuous monitoring applications. [24] provides a comprehensive list of environmental applications. In this section, two examples using mobile sensors/elements are presented to outline the general nature of these deployments and their features.

The Networked Infomechanical Systems (NIMS) [25, 26] project at UCLA integrates distributed, embedded sensing and computing systems with infrastructure-supported mobility to enable direct uncertainty characterization, autonomous adjustment of spatiotemporal sampling rate, and active sensor fusion. It has deployed robotic sensors suspended from cables which can move along the cable. The cableway system allows controlled mobility in horizontal and vertical directions in an energy efficient and precise way. In contrast to wheeled and tracked vehicles, the mobile element here is suspended as shown in Figure 1-1.

The NIMS test bed has been deployed to collect dense environmental and ecological data about populations of rare species and their habitats within a mountain stream ecosystem and the surrounding conifer forests and meadows. The sampling protocol that allows NIMS to collect and bring samples to the lab autonomously will allow researchers to monitor and measure short-lived, intense events in the environment as they occur.

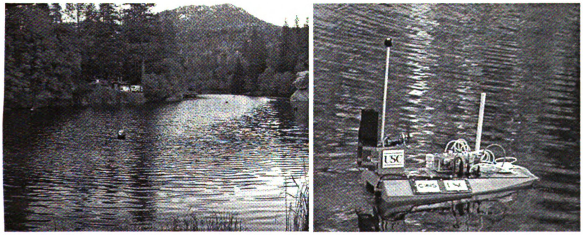


Figure 1-2: NAMOS at Lake Fulmor and a robotic boat
(source: http://robotics.usc.edu/~namos/gallery.html)

Figure 1-2 shows a Networked Aquatic Microbial Observing System (NAMOS) [25, 27, 28] deployed at Lake Fulmor. Buoy nodes are deployed across the surface of the lake and provide a spot measurement of temperature and chlorophyll. This data can be used for an estimation of these parameters in the complete extent of the lake. The robotic boat is guided by the buoy data that provides not only a reconstruction of field variables, but also a directive that determines the location of NAMOS deployment providing the greatest enhancement in fidelity of reconstruction.

1.2.2 Event Monitoring

Battlefield Surveillance: Important terrains, routes, and paths can be monitored by deploying sensor nodes to the dedicated field [29, 30]. Sensor nodes can sense the activities or predefined critical events, and send raw or post-processed information to the base station. Real-time situation requests can also be generated at the base station, and disseminated to the requested area.

Target Tracking: Collaborative mobile sensing can cooperate with a navigation system. For example, networked tracking of adversarial mobile targets using unmanned ground and airborne sensors is an emerging enabling application for the militaries' network-centric warfare capability. Target tracking [31, 32] is a prevalent military requirement across a wide spectrum of surveillance and reconnaissance scenarios including those in remote and urban battlefields, inside large buildings and public places such as airports, train stations, and inside underground tunnels.

1.3 Performance Metrics

1. Energy and Network Lifetime

The operating or functional lifetime of a network is an important performance metric for sensor networks. In the literature, various definitions for network lifetime have been considered – time until death of the first node, loss of coverage occurs in an area, a certain percentage of nodes die, or network becomes partitioned and hence useless. However, most of the studies use time until the death of the first node as the decisive factor. It is important to note that when the network becomes ineffective because of the death of a few crucial nodes, other nodes may still have a considerable amount of residual energy. Theoretically, maximum lifetime is achieved when all nodes in the network die at the same time - i.e, the amount of energy remaining in the network is zero. Thus, the total amount of residual energy remaining in the network is indicative of the extent of longevity that could be achieved.

Network lifetime is a joint result of two energy metrics [33], namely, the energy dissipation rate of the nodes and the variance of energy dissipation rate across the nodes. Thus, reducing the energy dissipation rate or distributing the load uniformly across the network could be beneficial to the longevity of the network.

2. Data Delay

Delay incurred by data packets generated in the network is another important metric especially for event monitoring or surveillance type of applications where an event that is detected has to be relayed to the base station with the lowest delay possible. In habitat or environmental monitoring type of applications, it is less important. Nevertheless, even in continuous monitoring applications, there is usually an upper bound on the delay that can be tolerated.

3. Accuracy

In continuous monitoring applications where a large amount of data is periodically generated, researchers have considered aggregating data from multiple nodes with an acceptable compromise in data accuracy. Accuracy can also be affected by packet delivery performance of the routing protocol. The larger the packet loss for a sensing area, the lower the accuracy.

4. Scalability

Sensor networks are expected to be large in terms of the number of nodes deployed. Thus, scalability is an important performance criterion for protocols and algorithms that are developed.

5. Fault tolerance

Sensors are energy constrained and thus are more prone to failures. Also, the nodes are expected to be deployed in hostile terrain and hence can be exposed to various physical, chemical and biological forces leading to failures. Hence, fault tolerance is an important performance metric.

1.4 Energy in Sensor Networks

1.4.1 Sources of Energy Expenditure

In a sensor network, nodes incur energy cost for sensing, processing the sensed data and transmitting the data to the base station. Radio communication is considered to be the single biggest source of energy consumption when compared to sensing and processing [2].

Therefore, researchers have focused on devising energy efficient strategies for saving energy at various layers. The radio interface uses energy when it transmits, receives or is idle. The energy required to transmit is usually the highest. According to the Mica2 sensor platform specifications [34], the current draw for the radio interface is as follows: 27mA for transmission with maximum power, 10mA for receive and $< 1\mu A$ for sleep.

The transmit energy of a node is expended due to the transmission of its own data towards a base station and routing data generated at other nodes. In networks with uniform data generation rate, the variation in communication energy drainage across nodes is dictated primarily by their individual routing loads.

1.4.2 Energy Conservation Mechanisms

As explained above, the operating lifetime of a sensor network is highly dependent on the rate at which nodes drain their energy. Thus, a lot of research effort has been focused on designing energy efficient mechanisms at all protocol layers: physical, medium access control, routing and application. In order to prolong network lifetime, different strategies should be adopted at all network layers including hardware improvement at physical layer, energy efficient medium access control protocol, topology control and routing. These strategies include avoiding packet collision [35] and idle listening [36], limiting carrier-sense energy consumption [37], optimal routing path selection and content filtering or error tolerance [38, 39] and data aggregation [40] at application layer.

In order to reduce the idle listening and up time of radio interfaces, sleep scheduling [41] has been studied in low data rate sensor networks. To reduce the transmission energy, several power-control-based schemes have been proposed. Since communication energy typically dominates the nodal energy budget, it is possible to extend network operating life by evenly distributing the data routing load across the network. Removal of routing hot-spots by load distribution can be achieved by deploying constrained-routing schemes [42-45] that aim to balance network traffic based on remaining nodal energy [46], link power budget [47, 48], or a combination of both [42]. Cluster-based routing protocols reduce energy consumption by data aggregation and balance energy consumption by rotating clusterheads [47, 49, 50]. Topology control algorithms [51-54] that attempt to use minimum transmission power than can guarantee the required level of network connectivity have also been studied extensively. While these mechanisms work reasonably well, they have limited effectiveness in networks with insufficient path diversities. Moreover, the energy-constrained routes tend to be longer, and hence they incur higher delivery latencies [43] and overall energy consumption on a per-packet basis.

Recently, the use of mobile elements for enhancing various network performance metrics has started to gain attention. The mobility of these mobile elements is controllable and planned to achieve desired performance objectives - especially energy savings for sensor nodes. The use of mobile elements does not preclude use of other energy saving mechanisms that have been used at the MAC, routing and other layers. Thus, controlled mobility can further enhance energy performance.

Chapter 2. Controlled Mobility and Network Performance

2.1 Introduction

Until recently, researchers have primarily focused on networks where sensor nodes and the base station are assumed to be static. Even when mobility was considered, it was random and unpredictable in nature. Therefore, mobility was an extra overhead to which the network must adapt, possibly at a loss of performance.

With progress in technology, diverse sensing and mobility capabilities will become more readily available to devices. Many mobile robots are capable of various sensing capabilities. It is not difficult to imagine large systems of such mobile autonomous agents [55] or hybrid networks consisting of static nodes and mobile elements performing various tasks in the near future. In this context, researchers have started investigating ways of exploiting controlled mobility to enhance network performance. Controlled mobility refers to the ability of one or more components in a sensor network to move autonomously. Components that are capable of mobility are more expensive than ordinary sensor nodes in terms of energy, hardware resources and navigational needs. Adding infrastructure components such as cableway or track along which the network elements may move has been proposed. Such components drastically reduce the overheads that are incurred by mobility.

Controlled mobility in a sensor network mitigates many issues:

a. Network deployment: Self deployment becomes easier if sensors are able to move to desired locations to improve coverage [56].
b. Fault tolerance: Sensor nodes could move to locations to ensure increased connectivity and avoid routing holes in the event of node deaths [57-61].

c. Energy harvesting: Mobile elements could potentially physically supply energy to sensor nodes from areas where energy is available [58, 62].

d. Adaptive behavior: Controlled mobility enables the network to be reactive to events in the network or to events in the monitoring area in surveillance applications [63, 64].

e. Concurrent missions: Mobile elements make it possible to support concurrent disparate missions with a common set of sensors [61, 65, 66].

f. Localization: Beacon from a mobile element can be used for localizing the static nodes in a sensor network [67].

g. Data delivery in disconnected networks: Mobile elements can help to deliver data between disconnected or intermittently connected network partitions [68-70] [8, 59].

h. Network longevity: Longevity of sensor networks can be increased by incorporating mobile elements in the network – mobile sinks, mobile relays or mobile sensor nodes [71-80].

2.2 Problem Definition: Lifetime Limitation in Static Sensor Networks

In sensor networks deployed for monitoring applications, nodes generate data periodically at a constant rate and send the data to one or multiple base stations. Typically, multi-hop routing is used to route the data. Thus, the amount of energy dissipated by the nodes in the network depends on their distance from the base station. The closer the node to the base station, higher the energy dissipation rate [81]. Figure 2-1:a shows an example monitoring network which is periodically routing data to two base stations located on the periphery of the field. Figure 2-1:b shows the energy dissipation rate of the sensor nodes.

It is clearly evident that the nodes in the immediate neighborhood of the base stations drain energy at a significantly higher rate compared to the rest of the network. This is due to the fact that first-hop nodes or gateway nodes have to forward messages originating from many other nodes in addition to their own messages. Thus, when the gateway nodes run out of energy, they cause a partition of the network, making the sensor network non-functional, limiting the lifetime of the network. Consequently, the lifetime of the network is directly related to the longevity of the first-hop nodes. This is a fundamental limitation for sensor routing solutions involving multipoint-to-point data models.

The routing of the sensed data to the base station is one of the most energy intensive operations in a sensor network. In the interest of network longevity, therefore, it is important to ensure that periodic data collection is performed in the most energy efficient manner. If the base station is static, then the most energy efficient solution is obtained by placing the base station in the most optimal position. While this results in lifetime improvement over an arbitrarily placed base station scenario, it still suffers from high energy drainage rates around the base station, as shown in Section 2.2. To address this issue, researchers have proposed the use of mobile elements in the network whose mobility can be controlled for the purpose of enhancing performance objectives such as energy efficiency and network longevity.

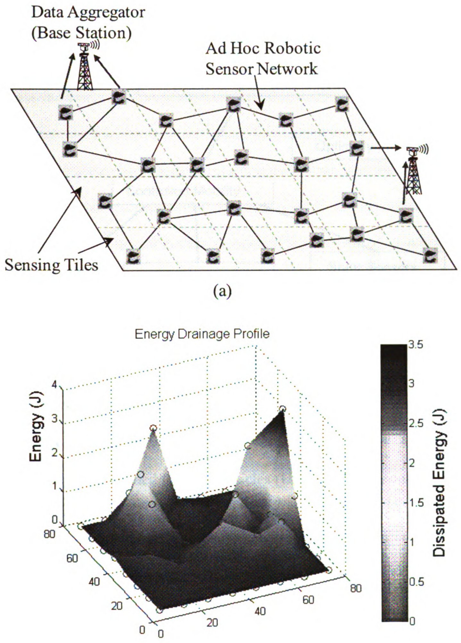


Figure 2-1: a) Sensor network with multi-hop data collection b) Energy drainage profile

2.3 Controlled Mobility for Data Collection

Figure 2-2 shows the possible solutions for data collection in sensor networks.

Controlled mobility solutions that have been proposed fall into one of the following three application categories depending on the elements which are mobile.

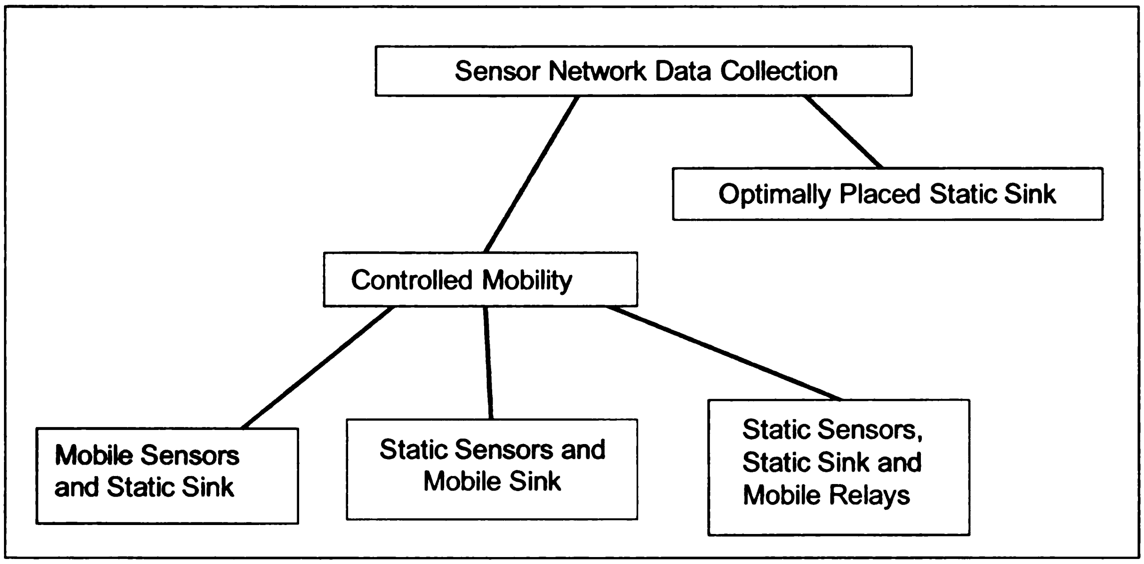

Sensor Network Data Collection
Optimally Placed Static Sink
Controlled Mobility
Mobile Sensors and Static Sink
Static Sensors and Mobile Sink
Static Sensors, Static Sink and Mobile Relays

Figure 2-2: Design options for data collection in sensor networks

First are networks where sink or base station is static, but the sensors are capable of mobility. Mobile robotic networks [82, 83] comprising teams of mobile robots, underwater vehicles, Unmanned Ground Vehicles (UGVs) or Unmanned Aerial Vehicles (UAVs) are examples. The mobility of these devices can be controlled by the network. Second are networks where sensors are static and the base station or sink is mobile. In this case, the trajectory of the base station can be planned to meet network objectives. The last type is networks where sink and sensors are static, however mobile elements called *relays* move such that they visit nodes and temporarily take on the responsibilities of the host node for a certain period of time.

2.4 Related Work

Existing literature pertaining to controlled mobility in each of the three categories discussed in Section 2.3 and shown in Figure 2-2 is presented in this section.

2.4.1 Mobile Sensors and Static Sink

The proposals in this category are aimed at networks which consist of mobile sensors such as networked robotic sensors capable of at least limited mobility. In this solution, the sensor nodes execute a controlled and concerted mobility sequence for extending the network life by distributing routing energy overhead across the network. The only solution in the literature that proposes such a mobile-sensor-based approach reduces energy consumption by adjusting [84] the positions of the intermediate nodes on a per-route basis from each sensor to the base station. The key idea is to move intermediate nodes on a route such that they are evenly spaced on the line between the source and the destination. Such a node placement has been proven to minimize transmission costs incurred by nodes when power control is used. The main emphasis of this mechanism is to extend network lifetime by reducing the energy spent per delivered data packet. The cost and benefit of mobility is taken into account when a mobility decision is made. This idea was further developed in [85] with the development of protocols and algorithms required to locally collect and share the cost and benefit information of movement to allow nodes to make local mobility decisions.

2.4.2 Static Sensors and Mobile Sink

The key idea behind mobile sink is that the nodes around the sink change over time and therefore the energy drainage due to routing occurs more uniformly, thereby extending the network life. Applications involving unmanned aerial (UAV) and ground vehicles (UGV) as mobile sinks have gained popularity particularly for networks deployed in adversarial and hard to reach terrains. In addition to the network life and collection delay, the time for which a UAV or UGV is exposed to threats during data collection also acts as a critical design constraint.

The research work pertaining to mobile-sink-based data collection can be classified into two categories: *real-time* and *non-real-time* collection. In the *real-time* case, the mobile sink moves along a trajectory visiting various nodes in the network in a pre-determined order. While the sink is visiting a node, all other sensor nodes route their data to the sink at its current location. The research challenge here is to jointly optimize the sink trajectory and the packet routing paths from each node to the mobile sink. Note that the delay that is incurred by data packets is in the time scale of packet routing similar to the static sink scenario. Therefore, this data collection scenario is referred to as *real-time*. Since the nodes need to route data to the sink at its current location, higher control overhead is required to maintain route information to the sink.

In *non-real-time* collection, the mobile sink collects data from a subset of nodes referred to as Designated Gateways (DGs). The rest of the sensor nodes route their data to one of the DGs where it is buffered until the mobile sink comes into the vicinity of the DG. The key research problem here is to jointly optimize the trajectory (includes the choice of *designated gateways* (DGs) and the order in which they are visited) and the packet routes from the sensor nodes to the DGs. Thus, the delay for the data packets is in the time scale of mobility and hence the scenario is referred to as *non-real-time*.

2.4.2.1 Real-time

In the existing literature that addresses the problem of data collection by a mobile sink to meet real-time application requirements, there are three distinct categories: the first addresses the problem of routing to the mobile sink [86-90] for a given trajectory, the second focuses on optimizing the trajectory for a given routing scheme [91-93] and the third attempts to jointly optimize mobility and routing [71].

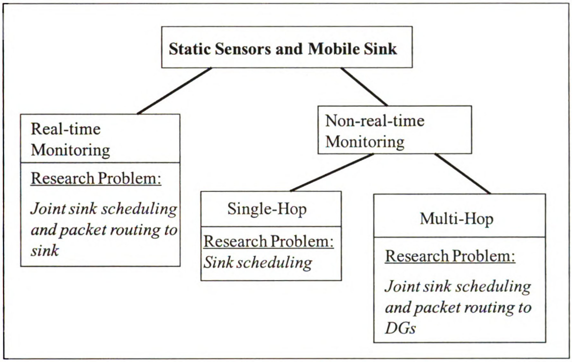


Figure 2-3: Solution classification for mobile sink data collection

Routing to a mobile sink is a challenge because of the cost of maintaining paths to the sink from every sensor node. There is a tradeoff between the need for frequent re-routing to ensure optimal network operation and minimizing the overhead of topology management. Two approaches have been taken to route to the mobile sink. As the sink moves, the nodes with first-hop connectivity with the sink change. Therefore, the sensor nodes could continuously establish new paths to the sink [87]. The authors [87] propose a distance-vector-approach-based routing protocol called MobiRoute. Route messages are exchanged periodically among neighbor nodes, and the next hop nodes are chosen by evaluating the costs of routing data through different neighbors. This results in high control overhead and high data loss as paths are torn down and re-established, especially if the sink is highly mobile.

To avoid excessive control overhead that results from rerouting paths, alternative approaches have been explored. The first approach entails extending the paths of existing routes by adding hops locally to route packets to the sink at its new location. Paths to the sink are occasionally rerouted [88] when they become overly inefficient. The second approach is to designate a gateway to relay data to the mobile sink from the sensor nodes. When the sink moves to a new location, the designated gateway establishes a path to the mobile sink and continues to route packets to the sink node using triangular routes [86]. These routes are gradually optimized over time. The incremental route optimization scheme presented in [86] is called *Net Cast Routing* (NCR). The third approach is a learning-based approach to route data efficiently to a mobile sink [89]. In the scheme presented in [89], first-hop nodes learn the sink's movement pattern over time and statistically characterize it as a probability distribution function. So, the sensor nodes make a decision about the next hop based on the time at which the data packet is generated. In the first and second approaches, as the sink mobility increases, data latency goes up due to longer paths to the sink or due to control overhead in addition to data loss that could occur when the route is re-established. Obviously these approaches do not scale well when the sink is very highly mobile. The third approach does not work well in networks where the mobile sink changes its trajectory frequently.

A few papers deal with the problem of trajectory optimization for a certain routing scheme. In this approach the sink moves from one location to another from amongst a set of feasible locations and stays there for an extended period of time. Nodes that are not in direct contact with the sink use multi-hop paths to route data to the sink. The trajectory computation problem involves computing the duration of time that the mobile sink needs to stay at each location, also referred to as sojourn time at each location. A linear programming formulation of the problem for obtaining the sojourn times is provided in [91] [94]. In this formulation the set of feasible locations is the same as the locations of the sensor nodes themselves. Also, it assumes that the movement from one sink location to another is instantaneous and that the data generation rate and initial energy are uniform across all nodes. An improved linear programming formulation that takes into account the problem of finding energy-efficient paths apart from the sojourn times has been presented in [93]. However, the feasible sink locations are restricted to a few hand-picked locations in the sensor field, unlike [91] where all node locations are potential candidates.

In [71], authors use heuristics to solve the joint routing and sink mobility optimization. They conclude that the best trajectory for a mobile sink is the periphery of a circular sensor field. The suggested routing strategy is to route to the edge of the network using shortest path routing and then to follow the edge to the current location of the sink, referred to as *round routing*. This paper does not explore the protocol aspects of this scheme. Also, it assumes a uniform data generation rate across time and nodes and assumes a circular sensor field. The best trajectory suggested by this solution may not necessarily apply to any arbitrary topology. In practice, implementing round routing requires the sink and the sensor nodes to have location-awareness.

The above-mentioned trajectory computation approaches assume a fixed routing scheme or limit the number of feasible locations. Also, they assume that the sink is likely to have long sojourn times at the feasible locations. Therefore, they do have mechanisms to alter sink trajectories based on any dynamic parameters such as changing data generation rates of nodes in the network. A distributed implementation of [91] has been presented in [92]. [95] also presents a similar solution. The protocol presented, called GMRE (Greedy Maximum Residual Energy) [96], greedily selects the site within a certain $d_{\max}$ surrounded by nodes that have the most energy left. After spending a time $t_{\min}$, the sink evaluates whether to move toward one of its adjacent sites. In order to decide if it should move, the sink gathers residual energy information of nodes around the potential future sites. If there are sites with higher residual energy than at the current site, the sink moves. During the duration of the sink movement from the current site to the next, the packets are buffered at the first-hop nodes. This increases the data latency. In the proposed solution framework, the handoff mechanism ensures that the latency incurred by data packets is lower as there is no buffering of packets involved.

2.4.2.2 Non-real-time

Single-Hop:

This long-lifetime and high-delay data collection is particularly suitable for delay non-critical applications in networks with long unattended operating life. The primary research problem is to compute the trajectory with minimum possible *round trip time* while collecting data in a single hop. For applications with heterogeneous data generation rates and limited buffers at the sensor nodes, the sink has to visit the nodes before their buffers overflow in order to minimize data losses [97, 98].

The existing literature in this area primarily addresses the continuous non-real-time applications using centralized trajectory computation and random-walk based mechanisms. The centralized trajectory formulations pose the sink scheduling problem either as a special case of a Vehicle Routing Problem (VRP) [97], a Traveling Salesman Problem (TSP) [98-101] or a Label Covering Tour problem in [102] or a cluster-formation problem [103, 104]. While providing efficient trajectories, all these solutions assume the availability of sensors' geographical location information, and therefore, would not apply to networks without localization services. A number of random-walk-based single-hop collection mechanisms without sensor localization are reported in [105, 106]. The main drawback of these random-walk-based approaches is their inability to deterministically collect data from all sensor nodes. In addition to coverage problems, this also leads to unbounded packet delivery delays.

Multi-Hop:

If the sensor network application is delay tolerant, then data from the nodes does not necessarily have to be routed to the current location of the sink. Instead the data can be buffered at the *Designated Gateway* (DG) nodes until they come in contact with the sink. This obviates the need to reroute paths constantly to the current location of the sink. In this paradigm, a node routes packets to the closest gateway node. Data collection using a mobile sink that moves through a sensor field along a fixed track is presented in [7, 73]. Although the path is fixed, the mobile sink has the ability to vary its speed. The paper presents different motion control algorithms for the sink including fixed speed movement, variable speed based on data rate and data priority. The extent of multi-hop routing here is dependent on the location of the fixed track with respect to the topology. Although this achieves better lifetime than the static-sink approach, due to the static nature of sink trajectory the energy depletion is eventually experienced by the DGs or nodes along the trajectory. The paper also describes the modifications that can be made to Directed Diffusion to route packets to the hop-wise closest gateway nodes. If the sink moves out of the transmission range of a gateway node before all the buffered data is transferred to the sink, the data may be dropped. In order to overcome the scalability and data loss problems that can occur when using a single mobile sink moving along a fixed path in a sensor field, [107] proposes using multiple mobile sinks. It also provides a load balancing algorithm that the multiple mobile elements can use.

In [108], a joint trajectory and route optimization problem has been formulated in which for a given MDH speed and a target delay deadline, "rendezvous points" are chosen such that the time required to visit them meets the target delay while minimizing the energy required to route data to the rendezvous nodes. This formulation assumes that the time required to upload data from the *rendezvous points* is negligible. In other words, the wireless capacity constraints are not taken into consideration. A random-walk based scheme with limited multi-hop routing was proposed in [106]. Although it performs better than the single-hop version, the general concern about coverage still remains. In contrast, the distributed framework presented in this paper can collect data with complete node coverage while minimizing the *delivery delay* as indicated by the results in Chapter 9.

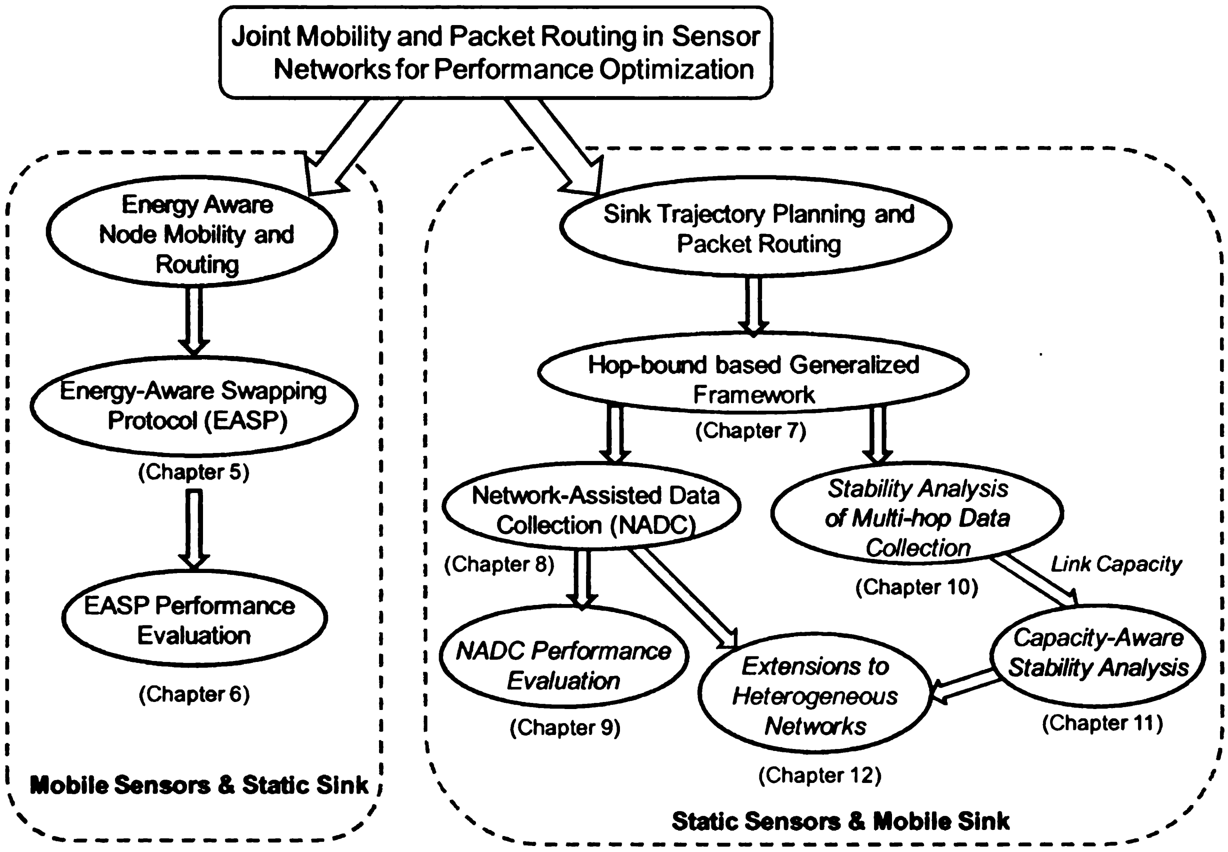
Test-bed experimental results showing that employing mobile sinks for data collection can help energy and lifetime performance are presented in [109]. In addition, researchers have studied the use of mobile devices for ferrying data amongst sensor nodes in a disconnected network [110] [68]. The idea is to use mobile elements whose movement is controlled such that the network throughput is enhanced. Use of multiple mobile elements and designing their trajectory has also attracted some attention [111] [112]. In this thesis, we restrict our attention to connected networks with one mobile sink.

2.4.3 Static Sensors, Static Sink and Mobile Relays

Mobile sensor based and mobile sink based solutions do not apply in networks where both the sensors and the base station are static. Or, in hostile terrain, it may not be possible for a base station to be mobile. In such situations, hybrid networks have been considered where base stations and most sensors are static. One or more energy-rich mobile nodes capable of controlled mobility called 'mobile relays' are added to the network to act as relays for the data [72, 74]. Mobile relays are different from the mobile sink in that they are not data collection devices. They merely take over the responsibilities (sensing and communication) of the nodes on their trail and enable the nodes to rest for a certain duration. This could help bottleneck nodes to conserve energy and thereby increase the network lifetime. The research issue in this paradigm is to determine the amount of time that the mobile relay will spend at each sensor location in the network. Then, while the mobile relay is at a node's location (say node *i*) for a duration say *d*, it inherits the responsibilities of node *i* and allows it to rest. The routes used by the sensor to send data to the sink may remain unchanged (static routing) or could change to utilize the mobile relay as much as possible (dynamic routing). Dynamic routing leads to higher improvement in lifetime as more packets pass through the mobile relay and hence more energy savings is achieved by the sensor nodes in the network.

Chapter 3. Thesis Scope and Contributions

The focus of this thesis is to investigate frameworks that can help to leverage mobile elements in sensor networks to improve performance, especially lifetime and various energy metrics. The basic goal is to explore joint mobility and packet routing optimization and in the process understand the tradeoffs that are involved to achieve the better performance. Based on the insights gained about the tradeoffs, system design guidelines are developed for given application and network parameters. Solution frameworks for two categories - namely, networks with mobile sensors and static sink and static sensors and mobile sink - have been considered in this thesis. Figure 3-1 shows a visual summary of the scope of the work presented in this thesis.



Joint Mobility and Packet Routing in Sensor Networks for Performance Optimization
Energy Aware Node Mobility and Routing
Energy-Aware Swapping Protocol (EASP)
(Chapter 5)
EASP Performance Evaluation
(Chapter 6)
Mobile Sensors & Static Sink
Sink Trajectory Planning and Packet Routing
Hop-bound based Generalized Framework
(Chapter 7)
Network-Assisted Data Collection (NADC)
(Chapter 8)
NADC Performance Evaluation
(Chapter 9)
Stability Analysis of Multi-hop Data Collection
(Chapter 10)
Link Capacity
Capacity-Aware Stability Analysis
(Chapter 11)
Extensions to Heterogeneous Networks
(Chapter 12)
Static Sensors & Mobile Sink

Figure 3-1: Pictorial view of the scope of the thesis

3.1 Collaborative Node Mobility for Energy Conservation

The motivation for this work is to leverage sensor mobility to achieve better network lifetime. In traditional static networks, network lifetime is limited by the energy reserves of the gateway nodes as pointed out in Section 2.2. When the network becomes partitioned, the network collectively still possesses a large amount of energy. The existing literature [84, 85] in this area attempts to increase the network lifetime by reducing energy spent per packet. The goal of this thesis is to investigate a different approach, namely to employ controlled mobility to evenly spread the routing burden and thereby increase the lifetime. The main emphasis of our mobility control framework is to achieve energy load distribution without significantly altering the packet transmission cost. In this new approach, energy consumption across the sensor nodes is distributed by executing a cooperative mobility mechanism based on localized energy information at the nodes, while the base-stations remain static. The idea is to extend network life without incurring additional delivery latencies (unlike the energy constrained routing mechanisms, which give rise to longer paths), especially in networks with limited path diversity.

In this thesis, the uneven energy drainage problem of sensor networks is mapped to a biological problem in penguin colonies. Inspired by the principles of the biological solution, a framework is proposed within the constraints of the sensor network resources. The proposed solution recognizes that energy cost of physical mobility is several orders of magnitude higher than communication cost. Thus, it attempts to minimize the amount of physical movement required to achieve longer lifetimes. The mobility pattern required to achieve the maximum lifetime is analytically derived. Also, a distributed framework is proposed that allows nodes to make a decision about movement based on local energy information and to implement the physical movement. The protocols developed resolve contention with respect to chosen movement and take care of handoff requirements to ensure that node mobility does not affect network layers significantly. The solution framework was implemented in simulation and the performance of the distributed algorithm is compared with a centralized version.

3.2 Generalized Framework for Mobile Sink Trajectory Planning and Routing

Data collection using energy-rich mobile elements has been proven to increase lifetime of sensor networks. In the literature, researchers have studied single hop and multi-hop data collection separately. In this thesis, the goal is to provide a generalized framework for non-real-time data collection using mobile sinks and to design distributed algorithms and protocols required to implement the generalized framework in practice.

3.2.1 Generalized Problem Formulation

Figure 3-2 shows the collection delay and lifetime performance of the solution space for mobile sink data collection. Note that since mobile sinks are considered energy-rich, the cost of their mobility is not considered in the problem formulation. The lowest-delay solution involves data collection using an optimally placed static sink [113-116]. The sink needs to be placed such that the average delay from all sensors to the sink is minimized. This solution, however, suffers from the worst possible network life due to the expedited energy depletion of the gateway nodes around the static sink as mentioned in Section 2.2. The opposite scenario - best lifetime with worst delay - can be achieved when the sink is programmed to periodically visit each node in the network for single-hop data collection. Maximum lifetime is ensured because of zero routing energy expenditure, and the maximum delay is caused due to the fact that the sink has to physically move to individual sensor nodes, and therefore the trip time will be the maximum.

Real-time event monitoring using a single dynamically chosen Designated Gateway (DG) represents a space that corresponds to higher delay, but better lifetime than the optimally placed static sink scenario. Better lifetime in this approach will be expected because the dynamic selection of DGs helps distribute the gateway energy load among all the sensor nodes. Slightly higher delay compared to the static sink case can be expected, because a majority of the time, the location of the mobile sink will not correspond to the delay-optimal sink location.

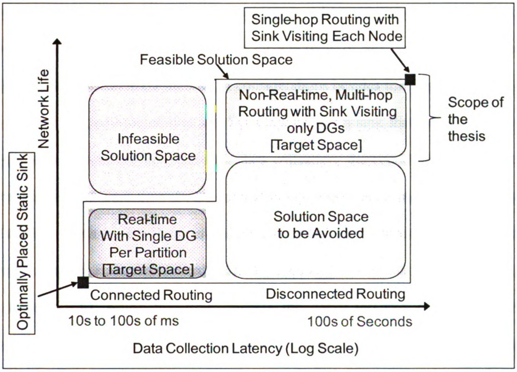

Single-hop Routing with Sink Visiting Each Node
Feasible Solution Space
Network Life
Infeasible Solution Space
Non-Real-time, Multi-hop Routing with Sink Visiting only DGs [Target Space]
Scope of the thesis
Optimally Placed Static Sink
Real-time With Single DG Per Partition [Target Space]
Solution Space to be Avoided
Connected Routing
Disconnected Routing
10s to 100s of ms
100s of Seconds
Data Collection Latency (Log Scale)

Figure 3-2: Solution mapping to latency and lifetime performance

Note that the space above the real-time monitoring solution is fundamentally infeasible. This is because any connected routing would require constant sensor-to-sink physical connectivity and multi-hop routing, and the latter causes enough energy drainage to prevent the lifetime from being close to the maximum. However, a better lifetime can be achieved by the non-real-time solution in which the mobile sink collects data from multiple designated gateways. Data from each sensor is uploaded to one of those gateway nodes. Due to the disconnected nature of the routing here, the delay is expected to be several orders of magnitude larger than the real-time case. Finally, note that the space below the non-real-time solution should be avoided because in spite of larger delays, the lifetimes offered by solutions in this area are too short.

Thus, there is a clear collection delay and lifetime tradeoff associated with the data collection solution space as observed from Figure 3-2. This thesis focuses on the non-real-time solution with the sink visiting a subset of DGs for collecting data. Non-real-time data delivery is appropriate for a number of monitoring type applications [117]. In such applications, a number of considerations besides lifetime and energy come into play for sink path determination. For instance, for a sink that periodically visits a network deployed in hostile terrain for data collection, the round trip time essentially determines the exposure time. In the case of energy constrained mobile devices such as UAVs [118] which have a limit on their uninterrupted flying time, the trajectory round trip time has to be planned accordingly. Alternatively, for a sink that collects making back-to-back trips, round trip time impacts the delay incurred by data packets as explained in Chapter 6. Thus, given the requirements of a monitoring application, constraints imposed by the mobile sink (including speed, energy resources and so on), the data generation rate of the network and such other parameters, an appropriate data collection solution has to be chosen; single-hop data collection or multi-hop data collection using mobile sink or a static sink. Thus, it is important to devise a generalized framework that encompasses the static sink scenario, single-hop data collection at the extremes and all multi-hop solutions in between them. In this thesis, the non-real-time mobile sink data collection problem has been formulated in a generalized way. In Chapter 6, a simple model has been developed based on key network, mobile sink and application parameters and analyzed to show the energy-delay balance achieved by the framework.

3.2.2 Distributed Algorithms/Protocols for Generalized Framework

Based on the survey of the literature presented in Section 2.4.2, it can be seen that mobile-sink-based solutions can be realized with one of the following trajectory computation mechanisms:

1. **Centralized:** A centralized entity that maintains sensors' location information computes the trajectory and programs the sink accordingly. In addition to poor fault tolerance, the primary disadvantage of this approach is that the trajectory adaptations due to topological changes can be very slow.
2. **Random-Walk based:** Sink trajectory consists of a series of random steps and at every step, the sink randomly picks a direction and distance to travel. While traveling, the sink collects data from the nodes that come in direct contact. Random-walk-based mechanisms can also be designed without localization services.
3. **Area Scanning:** Given the dimensions of the sensor field, the sink computes a trajectory to scan the entire sensor field area. The trajectory depends on the dimensions of the sensor field and the radio communication range. This approach does not necessarily require any sensor localization services, either.

In this thesis, the sink trajectory computation mechanism developed to realize the proposed generalized collection framework is based on a different approach. Sensor nodes cooperatively and dynamically determine the sink trajectory based on the physical network topology. The nodes themselves guide a mobile sink along the computed trajectory in a fully distributed manner. Trajectory adaptation in this approach can be more reactive than the centralized case. The proposed distributed framework called Network Assisted Data Collection (NADC), is described in Chapter 7. Network-guided navigation mechanisms are designed without relying on localization. The developed distributed algorithms/protocols are implemented in simulation and evaluated. The performance evaluation is presented in Chapter 8.

In Chapter 9, the model first developed in Chapter 6 is further analyzed from the perspective of stability of data collection. In addition, simulation results that validate the model results are also presented. In Chapter 10, the model is extended to take the capacity of the MDH-DG links into account and the impact on stability is explored. The insights gained in Chapter 9 and Chapter 10 are applied to a heterogeneous data rate sensor network and the distributed protocol framework (NADC) presented in Chapter 7 is enhanced. The enhancements designed for NADC and its experimental evaluation are presented in Chapter 11. The contributions of this thesis are summarized and ideas for future work are outlined in Chapter 12.

Chapter 4. Collaborative Node Mobility for Energy Conservation

4.1 Introduction and Related Work

The node mobility framework presented in this chapter is aimed at networks which consist of mobile sensors sending data to a static sink as introduced in Section 3.1. In this case, the sensor nodes themselves (e.g. robotic sensors) can be mobile to some extent. The idea is that the sensor nodes execute a controlled and concerted mobility sequence for extending network life by distributing routing energy overhead across the network. The only other solution in the literature that proposes such a mobile-sensor-based approach reduces energy consumption by adjusting [84, 85] the positions of the intermediate nodes on a per-route basis from each sensor to the base station as discussed in Section 2.4.1. Their key idea is to move intermediate nodes on a route such that they are evenly spaced on the line between the source and the destination. Such a node placement has been proven to minimize transmission costs incurred by nodes when power control is used. While the main emphasis of this mechanism is to extend network lifetime by reducing the energy spent per delivered data packet, we employ a different approach. The main emphasis of our mobility control framework is to achieve energy load distribution without altering the packet transmission cost. In this new approach, energy consumption across the sensor nodes is distributed by executing a cooperative mobility mechanism based on localized energy information at the nodes, while the base-stations remain static. The goal is to extend network life without incurring additional delivery latencies (unlike the energy constrained routing mechanisms, which give rise to longer paths), especially in networks with limited path diversity.

4.2 Biologically Inspired Problem Formulation

The proposed controlled node mobility in this thesis has been inspired by a natural cooperative behavior that was observed in Emperor Penguins [119] living in the Antarctica. During the winters, a group of emperor penguins form a crèche, which is an aggregation of penguins huddled together in close proximity to improve the collective heat insulation of the group. A crèche consists of an inner portion and an outer periphery. The penguins in the periphery are exposed to the brutal winds and frigid temperatures and hence, lose more heat than the birds near the center. During the course of the winter, penguins on the periphery move into the central regions and the crèche appears to move en-masse from its formation spot. As a result of this, the total heat loss of the group is thought to be evenly distributed across all the individuals. The crèche is a way of providing warmth and ensuring that the group survives the winter. The penguins appear to take turns to be at the periphery and as a result no individual penguin is exposed to the frigid temperatures for too long at a stretch. This behavior conserves body heat for the entire colony and enables most penguins to make it through the winter.

The idea of our work is to develop a parallel between the penguin crèche system and mobile sensor networks in which data is sourced from the nodes and is destined to one or multiple base stations, as shown in Figure 2-1:a. For such network models, nodes near the base stations are exposed to higher energy loss due to aggregation and higher routing overhead. Energy drainage rates for the nodes in a multi-hop network are shown in Figure 2-1:b. Observe that the nodes close to the base-station demonstrate significantly higher energy consumption rates compared to the nodes that are away from the base station. The nodes close to base stations can be compared with the penguins that have to withstand higher body heat loss at the periphery of their crèche. Similarly, nodes far away from the base station enjoy low energy drainage rates similar to the penguins in the inner portion of the crèche.

The goal of this work is to develop cooperative node mobility algorithms with similar objectives of the mobility behavior of Emperor penguins for evenly distributing the routing energy load across sensor nodes for extending network life. This requires that sensor nodes have mobility, and the resulting load distribution should be able to extend lifetime even after compensating for the energy for the physical node movement itself. This can be particularly challenging when considering the fact that energy cost for physical movement of a sensor node can be several orders of magnitude higher than the packet communication cost. Using simulation with the energy budgets of a realistic sensor and the movement energy of a micro-robot system prototype [120] we demonstrate that the proposed controlled mobility can work exceptionally well in a number of application scenarios. The localized mobility mechanism developed in this thesis is called Energy Aware Swapping Protocol (EASP) which is executed by the mobile sensors for determining their mobility pattern to extend the overall network life.

The proposed mechanism is general in not assuming that the base stations are necessarily located along the periphery of a network, as shown in Figure 2-1. For any arbitrary base station placement, there will be routing hot-spots around the base stations. The goal of the derived movement in EASP is to ensure that nodes spend comparable amounts of time in the hot-spots for even energy loss distribution. By distributing the routing energy consumption evenly, EASP is able to extend the network life.

The main contributions of this work are as follows. First, we show that there is a parallel between the heat insulation problem in the penguin colony and uneven energy drainage problem in sensor networks. Second, inspired by the goals of the penguin survival strategy, we propose a mobility algorithm to elongate lifetime in networks where nodes are capable of limited mobility. Penguin mobility and the proposed EASP algorithm share a common goal – namely, enhancing the system lifetime by ensuring that all entities are exposed to severe conditions for equal durations. However, the actual mobility algorithm in EASP is developed in accordance with the constraints of sensor networks -namely, the relatively high cost of mobility, routing, and MAC overhead incurred due to movements. Third, we have developed a localized distributed protocol to support EASP mobility. Finally, we have evaluated the performance of EASP on linear and two-dimensional networks and compared the performance with existing solutions. In the simulation experiments, the cost of mobility has also been taken into account.

This chapter is organized as follows. Section 4.3 uses an example to motivate the role of controlled mobility in sensor networks. Sections 4.6 thru 4.9 describe the proposed EASP algorithm in detail and present analysis for linear sensor networks. Chapter 5 presents the performance evaluation of the EASP framework.

4.3 Controlled Mobility for Energy Load Distribution

4.3.1 Example Scenario

A. *Shortest Path Routing (SPR)*

We consider sensor-to-base station data model that holds in most networks with monitoring style applications [117, 121]. For such a data model, the simplest way to minimize per-packet energy consumption is to use shortest path routing from each node to its destination base station. For example, in Figure 4-1:a, the shortest path routes from nodes *A* and *B* to the base-station are through nodes *B* and *E* respectively. Assuming nodes *A* and *B* as packet generators, and equal initial energy across all nodes, *E* will be the first node to be energy-exhausted, in which case the network will get disconnected from the base station. Additionally, while node *B* is burdened with routing *A*'s data, nodes *C* and *D* remain unused. This example illustrates as to how while minimizing the cost per packet, the shortest path routing can create hot-spots (node *B* in this case) due to the lack of load distribution.

B. *Energy-aware Routing (EAR)*

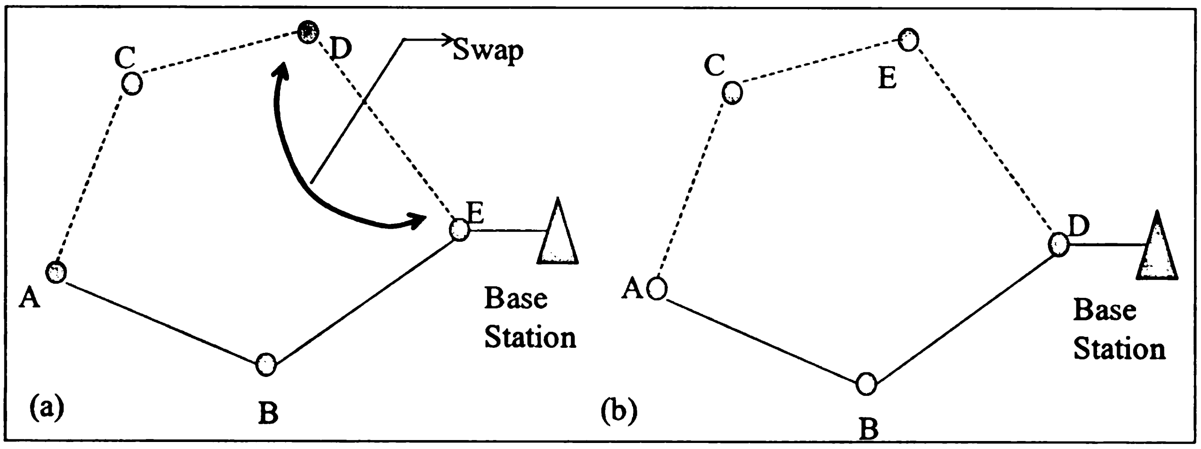


Figure 4-1: Benefits of location swap for extending network longevity

Now consider the situation when an energy-aware routing scheme [44] is used in the above network. If the *B* to base station route is established first, it is likely to be routed on the shortest path through *E*. But then, for node *A* to base station, the route *ACDE* will be chosen. Note that although this will lessen the energy burden on node *B* (compared to the shortest path case), due to the increased hop-count it will increase the total amount of energy that is spent to route packets generated by node *A*. Also, since node *E* continues to be the bottleneck (as it is the only node connected to the base-station), it will still be the first node to be energy-exhausted, and to cause a total disconnection from the base station.

This example demonstrates that for certain topologies working with a many-to-one data aggregation model, energy aware routing does not always help to extend the network life.

C. *Controlled Mobility*

Now consider if the nodes were capable of moving, then as node *E* is running out of energy, nodes *D* and *E* could swap their positions. After such a swap, if nodes *A* and *B* both continue to use the shortest path route, then the route becomes *ABD* as shown in Figure 4-1:b. This ensures that the energy expended per packet from node *A* to the base station remains the minimum. Also, since the hop count remains the same as before, the swap does not affect the delivery latency. Most importantly, the position swap ensures that the lifetime of network connectivity with the base station is extended compared to the shortest path routing and the energy-aware routing cases. This is the basic strategy adopted in EASP.

4.4 Network and Application Model

We assume a sensor field to be configured into equal size tiles as shown in Figure 4-5. One mobile sensor is deployed per tile for sensing desired parameters from the area of that tile. The deployed node can be located anywhere in the tile, and moves around within the tile for monitoring purposes. We assume monitoring style applications [117, 122] that require each node to generate data at a constant rate and to send it to a base station that is located within or at the periphery of the field. There could be multiple base stations, but data from a node is always sent to only one base station. The tiled network model assumed for this work is an application abstraction and is not a requirement for the EASP protocol framework itself. EASP can be extended to arbitrary network topologies with appropriate protocol extensions.

The wireless transmission range with respect to tile dimension is assumed to be large enough so that the nodes in two adjacent tiles can always communicate with each other irrespective of their individual locations within the respective tiles. This requires two nodes, placed in the diagonally opposite corners of two adjacent tiles, to be able to communicate. The number of neighbor nodes depends on the shape and size of the tiles. For example, in the network in Figure 2-1, each node has a maximum of eight neighbors, one in each of its adjoining tiles.

Nodes are mobile, and according to our EASP protocol a node can swap position with any of its neighbors. The goal is to develop a conflict-free algorithm for an energy-aware node swap pattern that can extend network life by evenly distributing the routing load across the network.

4.5 Problem Formulation

Consider a network with *n* tiles with a node in each tile routing data to the closest of the available base stations. Each node has an initial energy E_{init} and generates data at a constant rate and routes the data to the base station using shortest path routes. The energy aware node mobility problem can be formulated as finding the sequence of locations (tile) or a series of tuples $(time_i, tile_i)$ for each node such that the time until the death of the first node (*T*) is maximized subject to the constraint that each tile should have a node at all times.

4.6 Energy Aware Swap Protocol (EASP)

4.6.1 EASP Architecture and Protocol Outline

The architectural components of the EASP framework are depicted in Figure 4-2.

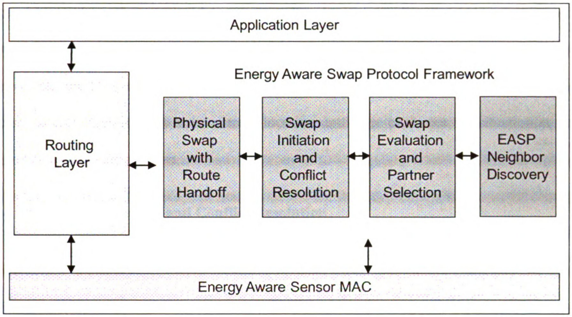

Application Layer
Energy Aware Swap Protocol Framework
Routing Layer
Physical Swap with Route Handoff
Swap Initiation and Conflict Resolution
Swap Evaluation and Partner Selection
EASP Neighbor Discovery
Energy Aware Sensor MAC

Figure 4-2: EASP protocol modules

4.6.1.1 Neighbor Discovery

Using a standard *Hello* protocol, each node gathers and maintains the a) Residual Energy, b) Energy Drainage Rate (EDR), and c) Neighborhood Energy Drainage Rate (NEDR) information about each of its 1-hop neighbors. EDR is defined as the combined communication and swap energy spent per unit time, which is measured by each node during its sojourn time at a given location. NEDR for a node is computed by averaging the gathered EDR values from all its 1-hop neighbors. In other words, NEDR represents the energy drainage rate information about the neighborhood of the 1-hop nodes of the node receiving the *Hello* packets. Therefore, NEDR contains information about nodes in the 2-hop neighborhood of the node under consideration.

4.6.1.2 Swap Evaluation and Partner Selection

Based on the above information gathered about its 1-hop and the 2-hop neighbors, each node periodically and asynchronously executes an algorithm that evaluates its own and its neighbors' residual energy and the energy drainage rates. Based on its relative state with respect to its neighbors, the node decides whether it is beneficial to swap, and if yes, it selects one of its neighbors as the swap partner. If a node decides not to swap at the current time, then it stays at its current location until it re-evaluates the situation at the next swap evaluation phase.

4.6.1.3 Swap Initiation and Conflict Resolution

If a node decides to swap with a selected partner, then it initiates a physical swap operation with the partner through a conflict-free swap process. A swap conflict can occur if two nodes select the same neighbor to swap with at the same time. To avoid such conflicts, EASP introduces a mechanism that ensures that only one node can swap with a given node at any given time. A physical swap occurs only after a node initiates a swap with a selected partner and the partner agrees to do so.

4.6.1.4 Physical Swap with Route Handoff

Once a swap partner is selected and any conflicts are resolved, the node and its swap partner physically move to each other's locations, referred to as the *target-destinations*. When two nodes swap, they are also required to mutually handoff their routing table information in order to make sure that the connections that were previously routed through them continue to function after the swap. The physical swap operation involves the following steps:

a. The swapping nodes compute a handoff location and move towards it.

b. Upon reaching the handoff location, they execute a handoff process

c. The nodes then move from the handoff location to their target-destination locations.

4.6.2 Interaction with the Core Protocols

MAC Layer: The EASP architecture does not make any specific assumptions about the underlying MAC layer. However, a collision-free and energy-efficient MAC is expected to leverage the benefits of EASP to its fullest extent. Scheduled TDMA access control protocols [123, 124] are known to provide the best energy performance due to their near-zero collisions and zero *idling energy* expenditure [6, 37] achieved through synchronous interface on-off operations [125]. However, as used for our experiments in Chapter 5, random access protocols [126] such as ALOHA, CSMA or CSMA/CA with reduced *idling energy* expenditure can also be used with the EASP protocol framework.

Routing Layer: The operation of EASP is also de-coupled from the specifics of the routing layer. Usual shortest path, geographic [127-129], or energy-aware routing algorithms [130, 131] can be used. Data-centric routing protocols such Directed Diffusion [132, 133] can also be used.

4.7 Protocol Details

The algorithmic details about the EASP protocol module are described in this section.

4.7.1 Swap Evaluation and Partner Selection

The goal of this component is to periodically evaluate if the node would benefit from swapping with one of its neighbors. In addition, the objective is to select the most beneficial swap partner from among the neighbors during the evaluation process. The key idea used here is to achieve the global objective of network life elongation by making each node to prolong its own life by local cooperation with its neighbors. We attempt to achieve this using a cooperative strategy inspired by the Emperor Penguins. A desirable cooperative strategy for the penguins is that every penguin in the crèche periodically attempts to swap its position with a neighbor penguin based on the available localized information such as the temperature and the penguin's own energy state. The collective goal of survival of the entire population could then be achieved as a result of this highly localized swap process. In the proposed EASP framework, the mobile sensors use a very similar localized and cooperative swap approach that is based on the sensors' own energy and the energy status of their neighbors.

The *primary criterion* used by a sensor in EASP is to move to a position that has a lower energy drainage rate than its current position. Therefore, each node tries to swap its position with a neighbor that is in a better energy draining position than its own. However, the physical mobility of a swap operation can be fairly energy-intensive, and therefore, the number of swap operations needs to be minimized. In order to control the number of swaps, it is essential that a *secondary criterion* should also be a part of the swap decision. Based on two different secondary criteria for controlling the number of swaps, we propose two flavors of EASP: EASP-Residual Energy (EASP-RE) and EASP-Lifetime (EASP-LT). EASP-RE and EASP_LT both require neighbors' energy drainage rate (EDR) and the residual energy information that is periodically gathered using the neighbor discovery protocol as outlined in Section 4.6.1.1. A sensor node is assumed to be able to estimate its residual energy by tracking its battery voltage drop with time as outlined in [134]. Based on this overall logic, the basic swap evaluation process is designed as follows:

1. During periodic evaluation of its neighbors' EDRs, a node identifies the neighbors with EDR values lower than itself as potential swap partners.

2. The node then evaluates the secondary condition which is different for EASP-RE and EASP-LT.

    a. In EASP-RE, the node verifies to see if any of the potential swap partners have a residual energy that is greater than its own by a threshold value referred to as Th_{swap}.

    b. In EASP-LT, the node assesses lifetime gain that can be achieved by swapping with each of the potential partners. The lifetime gain computation takes into account the energy cost of the swap operation itself. First, the expected lifetime of the node and its potential swap partners at their current locations is computed as the current residual energy divided by the EDR of their respective locations. Since we define the time until the first node death as lifetime, the minimum of the expected lifetimes of the node and its potential swap partners is considered to be the local lifetime, and referred to as the $Current\_life$. Next, for each of the potential swap partners, assuming that the two nodes will swap with each other, the expected lifetime of the node and its potential swap partner is calculated after subtracting the energy cost of the swap from their residual energy values. The minimum of the two expected lifetimes is the post-swap lifetime, and is referred to as $lifetime\_post\_swap$. Now, the secondary criterion is satisfied if $lifetime\_post\_swap$ is greater than the $Current\_life$ by the threshold value Th_{swap}.

3. Finally, of all the potential swap partners that satisfy the primary and the secondary criteria, the node chooses the lowest EDR neighbor. If none of the neighbors

satisfy both conditions, then the node decides not to initiate a swap at the current time. Note that this does not preclude the node from performing a swap if the swap operation is initiated by one of its neighbors.

The swap evaluation and partner selection logic is summarized in the pseudo code shown in Figure 4-3. Two extended versions of the swap algorithm EASP-RE-2H and EASP-LT-2H are also proposed in order to study the performance improvement of EASP by extending its EDR and other information scope from only 1-hop neighborhood to up to 2-hop neighborhood. In EASP-RE-2H, for example, each node not only maintains the EDR values of its 1-hop neighbors, but also the average EDR of all its 1-hop neighbors' neighbors. This quantity is called the Neighborhood EDR (NEDR), which is computed by each node and exchanged as a part of the Neighbor Discovery process. Apart from the primary and secondary criteria described above, a third criterion needs to be satisfied for the swap decision in these 2-hop versions of the EASP protocols. This third condition is that the EDR of the evaluating node must be greater than the NEDR of the evaluated neighbor as a potential swapping partner.

The idea behind this third condition is that before initiating a swap with a neighbor, a node should make sure that there are no nodes around that neighbor that might benefit more from swapping with this neighbor. It ensures that all nodes allow more beneficial swaps to happen, whenever possible. Experimental results described in Section 5.2 indicate that adding the 2-hop EDR information to the swap evaluation process indeed improves the performance of the EASP framework.

```
Swap Evaluation and Partner Selection at Node i:
    //initialization
    edr_diff_seen_so_far  ← 0
    swap_partner  ← NULL  // no swap partner selected yet

    for (each neighbor in Neighbor_Table) {
        edr_diff ← EDR(i) – EDR(neighbor)   //compute difference in EDR values

        // check if this is the lowest EDR neighbor seen so far
        if ( EDR(neighbor) < EDR(i) and (edr_diff_seen_so_far < edr_diff) ) {
            if (secondary_criterion is satisfied) {   //see below
                If (EDR(i) > NEDR(neighbor))   // required only for 2Hop version of EASP
                    swap_partner ← neighbor
            }
        }
    }
    If (swap_partner was selected)
        Initiate swap operation

Secondary Criterion Check:
EASP-RE: if (remaining_energy(neighbor) > remaining_energy(i) + $Th_{swap}$)
            Secondary criterion satisfied

EASP-LT:   //compute current lifetime of node i and neighbor
    $Lifetime_i$  ←  remaining_energy(i) / edr(i)
    $Lifetime_{neighbor}$  ← remaining_energy(neighbor) / edr(neighbor)
    Current_life  ←  min($lifetime_i$, $lifetime_{neighbor}$)

    // compute estimated lifetime gain post-swap
    $lifetime\_ps_i$  ←  (remaining_energy(i) – swap_cost) / edr(neighbor)
    $lifetime\_ps_{neighbor}$  ← (remaining_energy(neighbor) – swap_cost) / edr(i)
    lifetime_post_swap  ← min($lifetime\_ps_i$, $lifetime\_ps_{neighbor}$)

    If (lifetime_post_swap > current_life + $Th_{swap}$)
        Secondary criterion satisfied

```

Figure 4-3: Pseudo code for swap evaluation and partner selection

4.7.2 Swap Initiation and Conflict Resolution

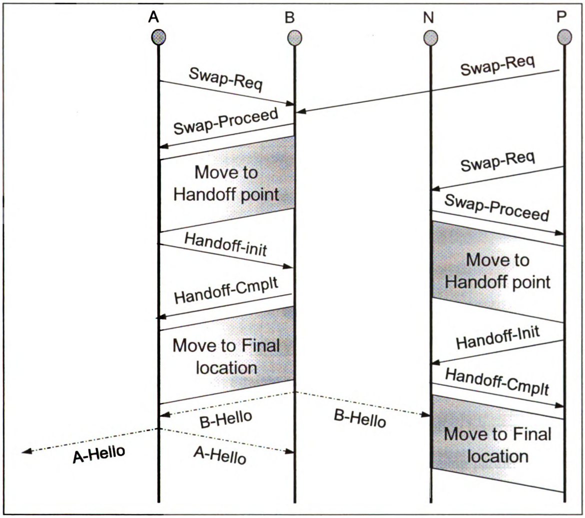


Figure 4-4: Timing diagram for swap initiation and conflict resolution

Since the sensors perform autonomous swap evaluation asynchronously, it is possible that a node is simultaneously selected as a target swap partner by more than one of its neighbors. To deal with these conflicts, EASP introduces a conflict resolution technique similar to the RTS/CTS mechanism used in the IEEE 802.11 MAC protocol. Two messages, namely, *Swap-Req* and *Swap-Proceed* are used for beginning the swap operation. Upon deciding on a swap, a node sends a *Swap-Req* packet to the selected partner. If the partner is available for a swap, then it responds with a *Swap-Proceed* packet. This way, the neighbors of the two swapping nodes become aware of the swap in progress and hence any further contentions are avoided.

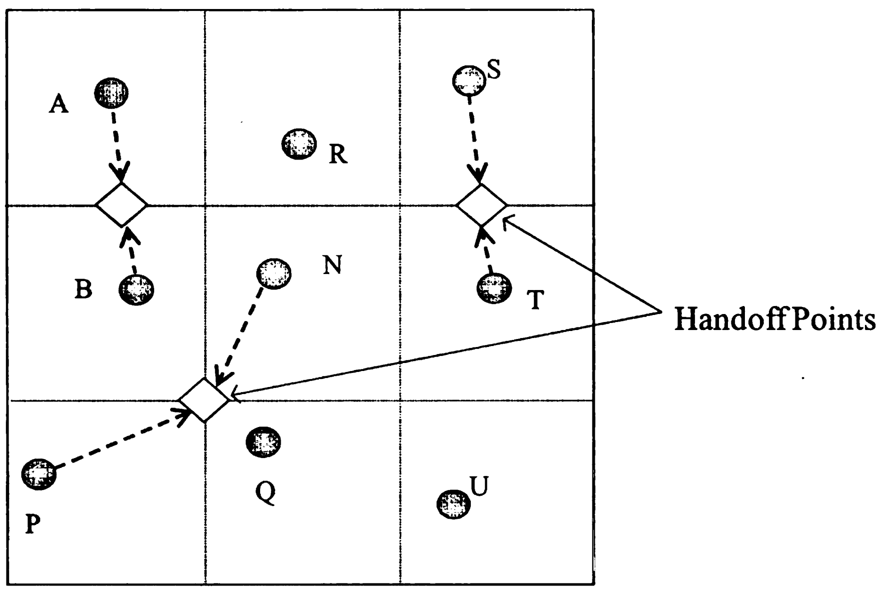


Figure 4-5: Straight and diagonal handoffs

Figure 4-4 shows the timing diagram for an example swap operation between nodes *A* and *B* (in Figure 4-5), initiated by *A* after it selects *B* as its swapping partner. Node *A* initiates a swap by sending a *Swap-Req* message to Node *B*. In the mean time, node *P* which is not aware of the *Swap-Request* from *A*, also sends a *Swap-Request* to sensor *B*. Since the request from *A* was received first, node *B* ignores the *Swap-Request* from *P*. Then node *B* responds to the *Swap-Req* from node *A* with a *Swap-Proceed* message. Sensor *P* waits for the *Swap-Proceed* and eventually times out. It checks to see if it can swap with one of its other neighbors and finds that node *N* satisfies the necessary swap conditions. So, it initiates a swap with node *N*. This demonstrates that multiple simultaneous swaps in the same neighborhood are possible as long as each node is involved in at most one swap operation.

After the handoff phase is completed, both *A* and *B* continue broadcasting *Hello* messages to establish new neighborhood relationship with the nodes around. The new neighbors of A namely *P*, *Q*, *N* and *R* add node *A* to their neighbor list. Note that nodes *P* and *Q* have to add a new entry for node *A* while nodes *N*, *R* and *B* have to only change the status information. Similarly, when node *B* sends out a *Hello* message, nodes *R* and *N* change the status of *B*. After the swap, *B* is no longer a neighbor of *P* and *Q*. Hence, nodes *P* and *Q* eventually time out and remove entry for node *B* from their neighbor tables. This concludes the localized swap initiation and conflict resolution algorithm.

4.7.3 Physical Swap with Handoff

The physical swaps operates through the following three distinct phases.

Phase 1: Move towards a computed handoff point

Depending on the relative locations of the swapping sensors the swaps could be '*Straight*' or '*Diagonal*'. For example, swaps between nodes *A* and *R* or between *A* and *B* in Figure 4-5 will result in a '*Straight*' swap, whereas a swap between *A* and *N* will be '*Diagonal*'. This swap type determines the location of the *handoff point*, at which the routing table and any other relevant information are mutually handed off between the swapping partners. In the case of a straight swap, the handoff location is computed to be the intersection point of the line joining the current locations of the two sensors and the tile boundary line between them. For a diagonal swap, the handoff location is the corner point at the intersection of the two tiles in which the nodes are located. The handoff points for each kind of swap operation are shown in Figure 4-5. It is important that a swapping node does not abandon the routing responsibilities of its pre-swap tile before its swap partner is ready to take over the routing responsibilities of the tile. To ensure this, the handoff is performed when both the swapping partners reach the computed handoff point. In a non-tiled network with arbitrary topology, the handoff point would be selected midway between the locations of the two swapping partners.

Phase 2: Perform route handoff

Upon reaching the handoff point, the node that had initiated the swap operation, initiates route handoff by sending a *Handoff-Init* message. The swap partner responds with a *Handoff-Cmplt* message. The nodes exchange information about their old locations with each other such as tile identifier, next hop and the EDR values. After this point, the swapping nodes take over the routing responsibilities of the new tile.

Phase 3: Move to the target destination

During this phase, the swap partners move to the desired position within the target tile, and continue to send out periodic neighbor discovery messages indicating their ownership of the new tile. Nodes that were routing packets through the swapping sensors update their next hop information to reflect the swap operation when they receive these messages from the swapped nodes during this phase. Consider the example scenario in Figure 4-5, in which node Q was routing packets through node B before a swap between A and B took place. After the handoff between A and B is completed, based on A's *Hello* packets, node Q updates its next hop to A and starts routing packets through it.

4.7.3.1 State Machine for the Physical Swap

Figure 4-6 and Figure 4-7 show the state machines for the swapper (initiating node) and the swappee during a physical swap. The five states and their associated events and actions are described as follows:

IDLE: This is the default state of a sensor node. It can transition to the PRE-MOVE state as a *swapper* after it sends a *Swap-Req* message to initiate a swap. As a *swappee*, a node can enter in the PRE-MOVE state when it receives a *Swap-Req* message.

PRE-MOVE: Upon receiving the *Swap-Proceed*, a swapper moves to the MOVE state. If it does not receive the Swap-Proceed within a certain duration, it goes back to IDLE. A swappee enters the MOVE state after it sends a *Swap-Proceed* message.

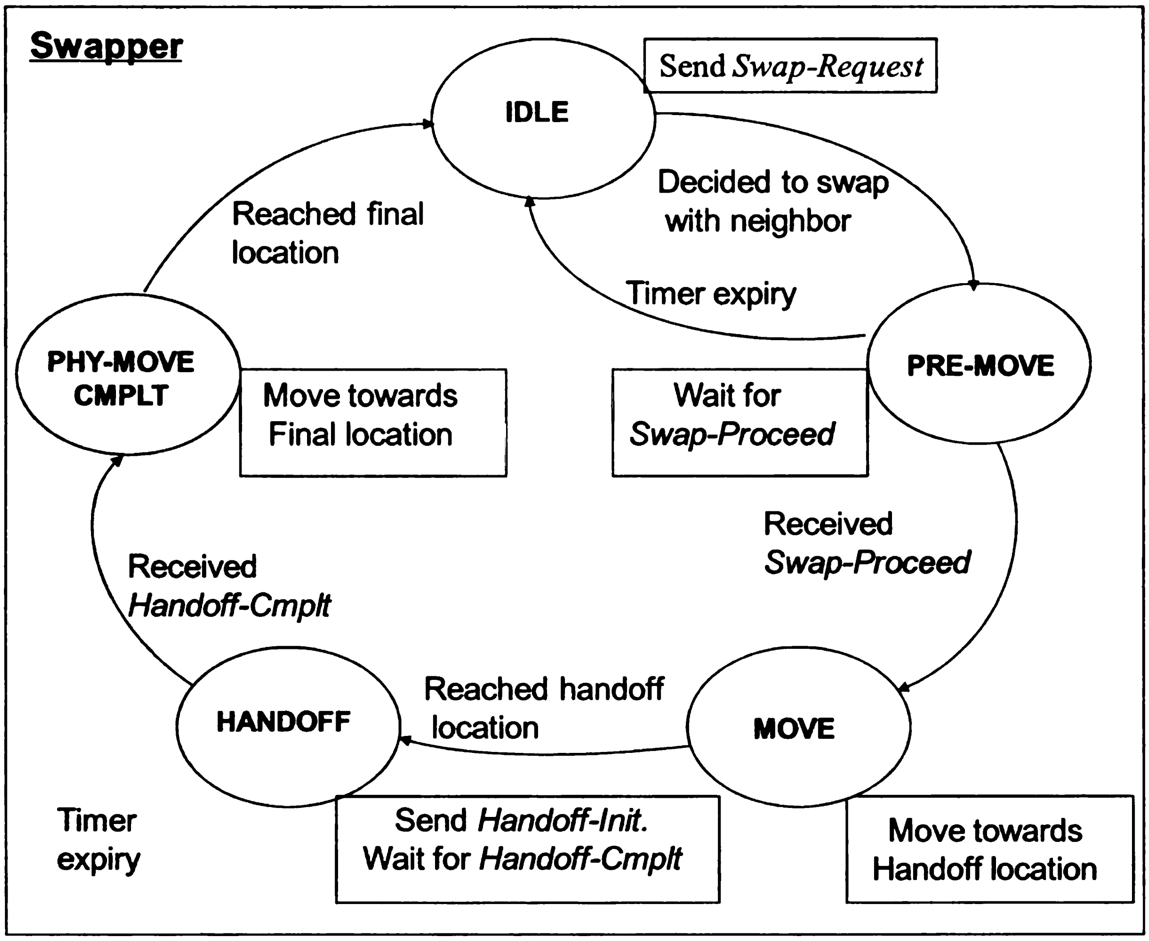


Figure 4-6: State machine for the swap operation at Swapper

MOVE: Both the *swapper* and the *swappee* remain in this state while moving towards the handoff location. Upon receiving the handoff location, the nodes enter the HANDOFF state.

HANDOFF: The nodes remain in this state while executing the routing information handoff, and exit to the PHY-MOVE-CMPLT state only after the swappee sends the *Handoff-Cmplt* message and the swapper receives the *Handoff-Cmplt* message from the swappee.

PHY-MOVE-CMPLT: In this state, both the *swapper* and the *swappee* move towards their desired location in the new tile, after which nodes reenter the default IDLE state.

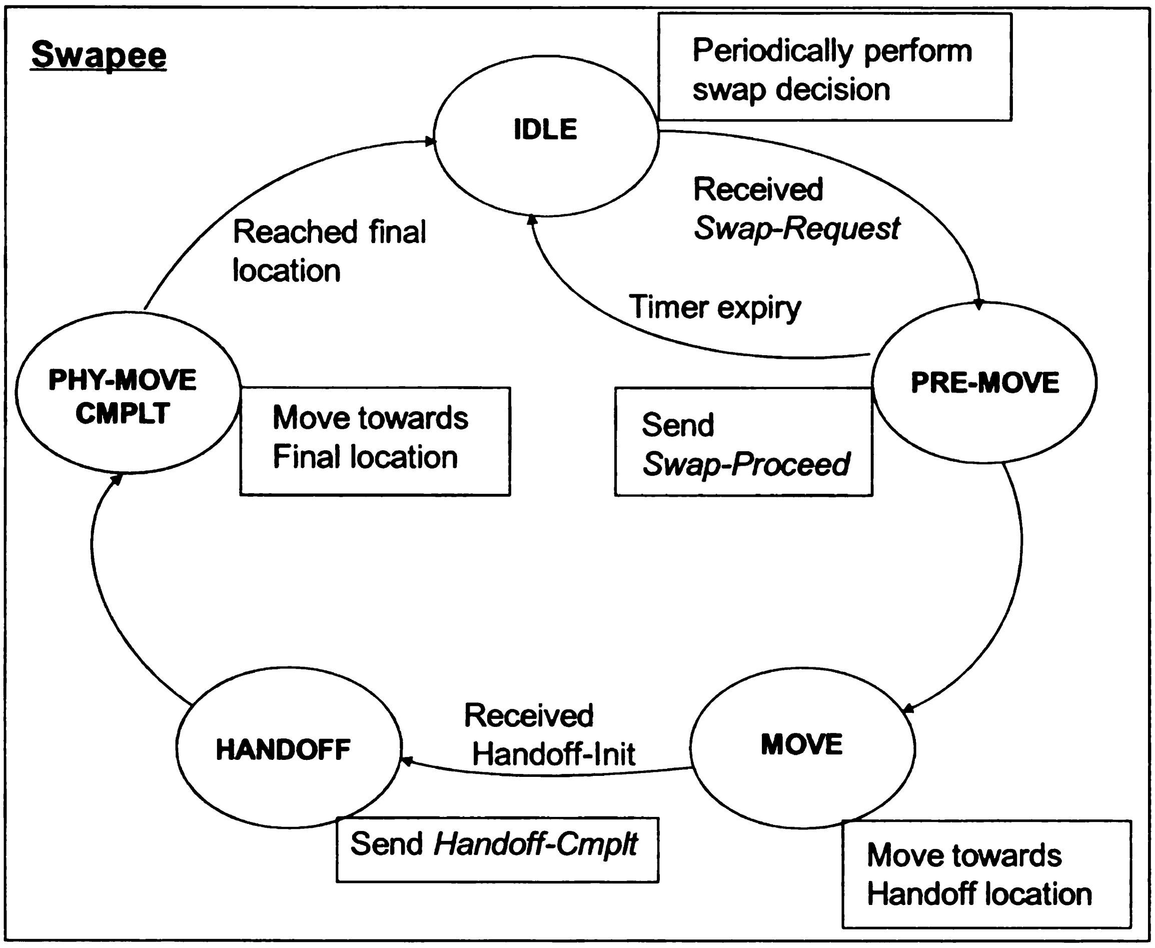


Figure 4-7: State machine for the swap operation at Swappee

4.8 Design Generalization for Arbitrary Topologies

To be applicable for arbitrary non-tiled topologies, the EASP framework will need to be extended for accommodating the following aspects. First, the distance between neighbors can be larger than the tiled model, since the neighborhood will no longer be restricted by the tile boundaries. Second, when a swap occurs, the swapping nodes may not be within the communication range of all the neighbors of the swapping partner immediately after the handoff. The temporary lack of communication between a node and its new neighbors can lead to a temporary disruption of the route and cause some data loss. To deal with the data loss, the swap evaluation and partner selection algorithm should be enhanced to take the distance between the swapping nodes into account. Finally, there will not be any restrictions on the number of neighbors a node can have. This means that a node can have more than eight swap partner choices. Note that most of these extensions involve implementation issues, while the basic concept of EASP remains the same.

4.9 Analysis of Swaps in Linear Networks

4.9.1 Performance Upper Bound

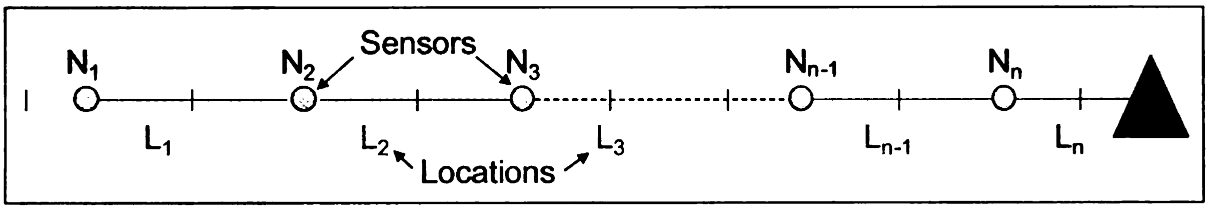


Figure 4-8: A linear network with *n* nodes and one base station

For analytical tractability, we do this analysis for a linear network of *n* nodes as shown in Figure 4-8. Also, we assume a synchronous version of EASP in which swap evaluations happen at the same time periodically on all nodes. Since the amount of time that a node stays at a location (also referred to as the sojourn time) is expected to be much larger than the physical time for a swap, physical swaps are assumed to occur instantaneously. Each node periodically generates a certain amount of data destined to the base-station. Data packets are forwarded in the direction of the base station by each node. Thus, each location has an EDR value proportional to the number of packets that are generated and forwarded by it. All nodes have the same initial energy and the swap cost is assumed to be zero (only for this analysis). Lifetime of the network is defined as the time at which the first node in the network runs out of energy.

The maximum lifetime of the network can be achieved when all nodes get energy exhausted exactly at the same time. In other words, the target will be to attain a situation where the sum of the energy left on all nodes in the network is zero. With zero swap cost, the maximum achievable lifetime can be written as:

$$T_{\max} = \left(E_{init} \times n\right) \Big/ \sum_{i=1}^{n} EDR_i \qquad (1)$$

where n is the number of nodes, E_{init} is the initial node energy, and EDR_i is the energy drainage rate at location i. EDR_i is higher for locations nearer to the base station because of their higher routing burden. Now, to achieve maximum lifetime, all nodes have to equally share the routing burden. Therefore, since each location has a unique energy drainage rate, every node must spend equal amount of time in each location in the network. Assuming negligible swap durations compared to the sensor sojourn time, the total time that a node should spend at each location (sojourn time) can be expressed as:

$$T_{sojourn\_time} = T_{\max} / n$$

This implies that every node has to swap in such a way that it visits every location in the network at least once. Because of the unavailability of any circular path in the linear network, each node has to traverse to one end and then back to the other end to ensure that it visits every location. This implies that a node actually needs to visit each location twice. As a result, the modified sojourn time of a node at each location (this is also the interval between two consecutive swap cycles) can be written as:

$$Swap\_Interval = T_{\max}/2n = E_{init} \Big/ 2 \times \sum_{i=1}^{n} EDR_i \qquad (2)$$

where $T_{\max}$ is the maximum lifetime as computed in equation (1). Now, let the minimum number of swaps required to achieve the maximum lifetime be Min_{swaps} .

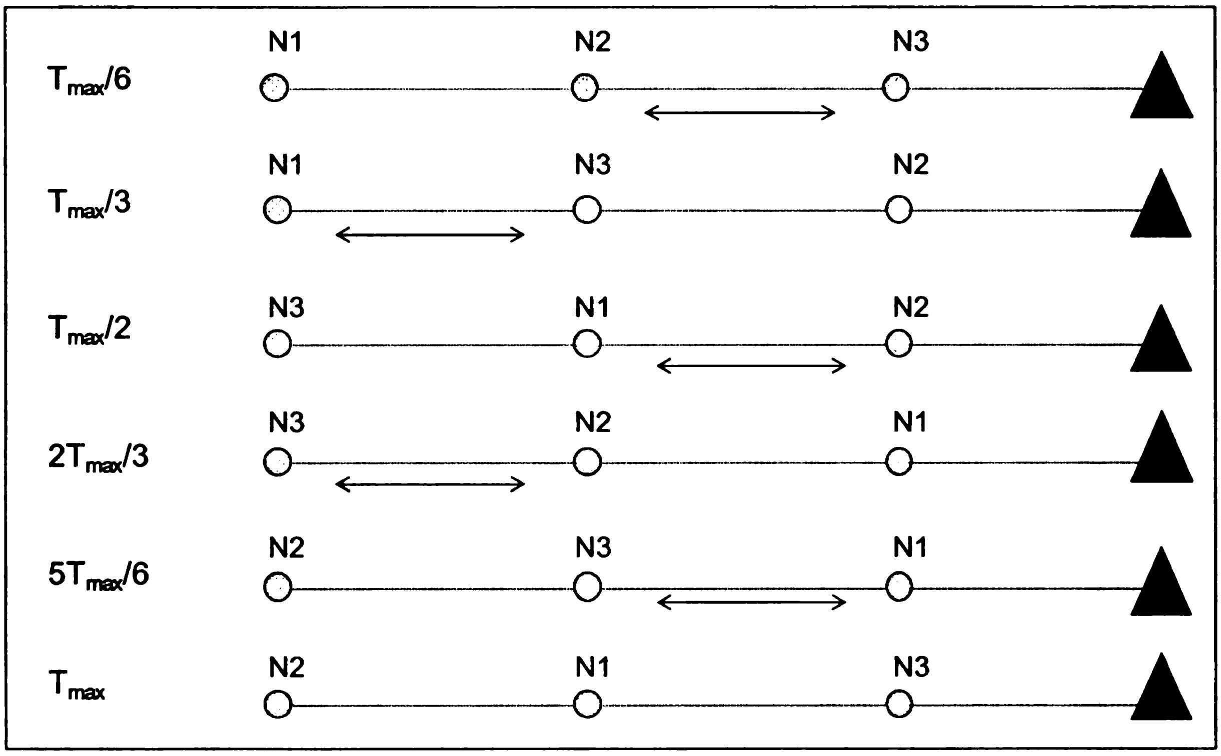


Figure 4-9: Swap sequence for maximum network life in a 3 node linear network

Case 1: If n is odd, the number of nodes involved in a swap during a swap cycle is $(n-1)$. With two nodes per swap, the number of swaps is $(n-1)/2$. If the interval between swap cycles is $Swap\_Interval$, the number of swap cycles that can occur within $T_{\max}$ is one less than $T_{\max}/Swap\_Interval$, which is $(2n-1)$. Therefore, the minimum number of swaps required to achieve the maximum lifetime:

$$Min_{swaps} = (2n-1)(n-1)/2.$$

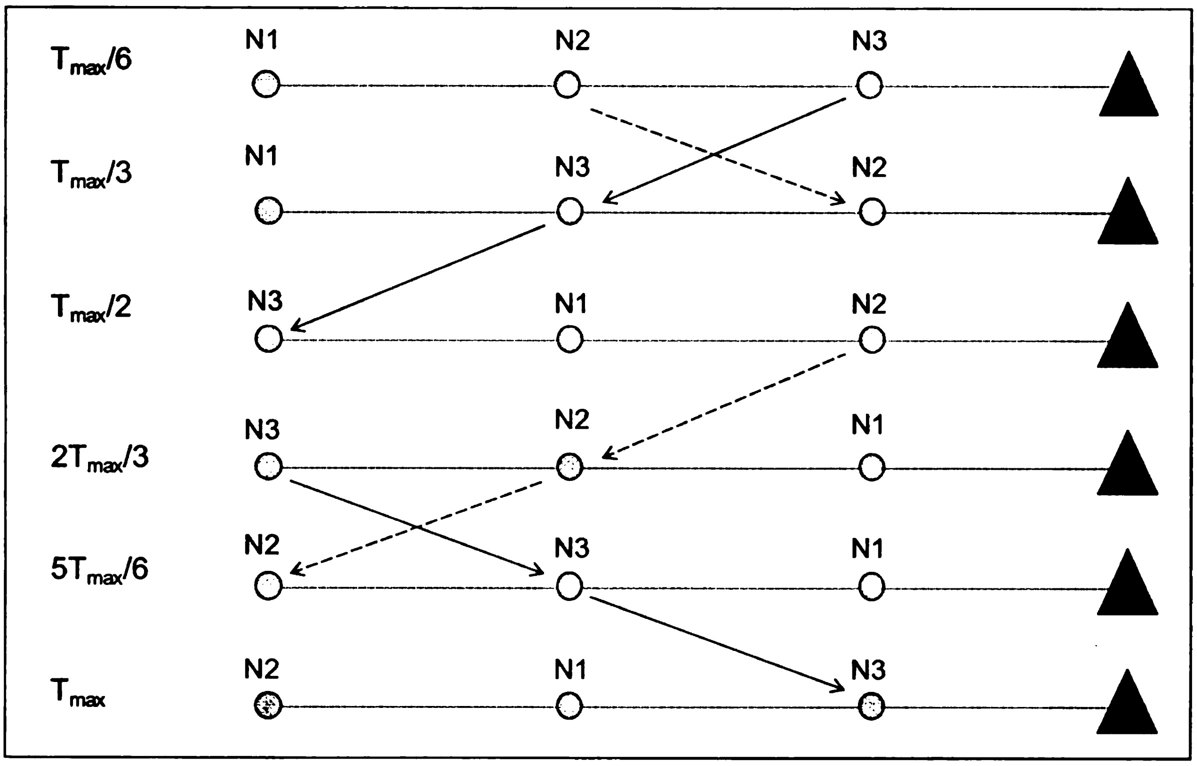


Figure 4-10: Trajectory of N2 and N3 during the maximum life swap sequence

Figure 4-9 shows the swaps that are required and the intervals at which these swaps must occur in a 3 node network in order to achieve the maximum lifetime. For this network, 5 swap cycles are required at intervals of $T_{\max}/6$. Figure 4-10 shows the movement pattern of the center node and an end node in the 3-node linear network. Note that, being in the center, node N2 has to move to one end of the network and then to the other in order to visit all the locations.

<u>Case2:</u> If n is even ($n > 2$), using the similar logic as above, the minimum required swaps for maximum life can be derived as: $Min_{swaps} = n(2n-3)/2$.

<u>Lifetime with non-zero swap cost</u>: When the zero swap cost assumption is relaxed, the maximum lifetime is expected to be relatively lower. However, for a given *n*, the minimum number of swaps for the maximum life is expected to remain the same. This is because the required swap pattern still needs to ensure that the nodes spend equal amount of time at each location. The pattern does not depend on the initial energy or the swap cost itself. Therefore, the maximum lifetime (best case) with non-zero swap cost can be obtained by deducting the cost of the required number of swaps from the initial energy in equation (1) resulting in:

$$T_{\max} = \left\{ \left(E_{init} \times n \right) - \left(Min_{swaps} \times S_{\text{cost}} \times 2 \right) \right\} \Big/ \sum_{i=1}^{n} EDR_i$$

where S_{cost} is the cost incurred by each node involved in a physical swap.

4.10 Summary

In this chapter, a cooperative node mobility framework called EASP was developed to evenly distribute the routing energy burden experienced by nodes in networks with many-to-one data model. Inspired by principles of penguin colony strategy for surviving frigid winters, a swapping mobility algorithm for nodes was proposed. The protocol components and distributed algorithms required for the swapping decision and the handoff mechanisms based on localized information were developed. The performance upper bound for EASP was derived for linear networks. Performance evaluation of EASP is presented in Chapter 5.

Chapter 5. Energy Aware Swapping Protocol - Performance Evaluation

In Chapter 4, Energy Aware Swapping Protocol (EASP) was developed and the various components were presented. In this chapter, the performance results obtained from simulation experiments of EASP are presented. Two simulators were used for the evaluation: a C-based simulator and NS2 based detailed packet level simulation. The run time required to run lifetime simulations using the detailed NS2 implementation was impractical. Therefore, we used the detailed simulation to run EASP using lower initial energy for nodes and analyzed the viability of the EASP protocol despite the control overhead and packet losses during handoff and MAC effects. The C simulator was used to run the simulations with large nodal energy to evaluate the long-term performance of EASP and to compare it with a centrally computed best solution.

While the results from both implementations are very useful for performance evaluation, the results obtained from these two implementations for similar scenarios are not directly comparable. This is because the C simulator does not model the MAC and the physical layers. Thus, the MAC layer retransmissions and losses are not taken into account. Also, it cannot run the swap conflict resolution and handoff functionality associated with state machines specified in Figure 4-6 and Figure 4-7.

Lifetime of a network is defined [87] as the duration after which at least one node in the network becomes energy exhausted. In most scenarios, including in this linear network, it has been observed that the energy exhaustion first happens to a node that is visiting a location 1-hop away from the base station, thus rendering the network disconnected from the base station. Due to this, observing the first instance of energy exhaustion does provide a practical perspective of the network life. Also, in several of our experiments, the CPU time taken for simulating the energy exhaustion of the entire network, especially with all the EASP protocol components, proved to be prohibitively long. Because of these two reasons we have chosen to adopt the first node exhaustion definition of network life.

5.1 Results from C Based Simulator

The preliminary performance evaluation of EASP was conducted using a simulator written in C. Simulation experiments were run on a tiled sensor field with one sensor node in each tile as shown in Figure 2-1. Based on a sensor transmission range of 30m, the size of the tile was chosen to be $1lm \times 1lm$. For a simulated data monitoring application, each sensor node generates a 512 byte packet every second, destined for the base-station. We modeled the mobile sensor nodes as a MICA MOTE [34] sensors attached to LEGO Mindstorm robotic vehicles [120]. The Mindstorm robots are powered by two Lego Technic 43362 motors [135]. Specifications of the sensor MOTE and the vehicle motors are shown in Table 1.

Sensor Mote Related Parameters	
Transmission range	100ft or 30.5m
Transmit Current	12mA
Receive Current	1.8mA
Data rate	40kbps
Motor related parameters	
Rotation speed	219 rpm
Torque	2.25N.cm
Electrical power	1.1W

Table 1. Sensor node and vehicle motor specifications

The transmission cost in MICA for a 512 byte packet (based on the values shown in Table 1) is 3.8 mJoules and the cost of receiving a packet is about 0.6 mJoules.

Communication cost, the energy consumed to transmit and receive one 512 byte packet, is therefore 4.4 mJoules.

Swap cost is defined as the amount of energy consumed by a node to physically move across tiles. Assuming a swap distance of 11m (tile size), and considering the rpm and power of the LEGO motors as shown in Table 1, the energy required to drive the LEGO motors for a swap is 40.8 Joules. Since two nodes are involved in a swap, the total cost of each swap is 81.6 Joules.

Performance of the distributed EASP, as proposed in Section 4.6, has been evaluated and compared with the following protocols:

SPR: Shortest path routing is used to route packets to the base-station. No node swapping is executed.

Centralized Swapping Algorithm (CSA): An algorithm that relies on global network topology and initial energy information performs swaps that result in maximum operating lifetime for the linear network as described in Section 4.9. The performance of this algorithm is considered to be the upper bound for the performance of EASP.

Energy-Aware-Routing (EAR): We have implemented a constrained routing algorithm [44] which maximizes the minimum remaining energy along any route from a sensor node to the base station.

Note that the behavior of EAR is expected to be not much better than the SPR case for networks with insufficient path diversities. Since in our linear network model all packets have to pass through a single gateway node (see Figure 4-8), there is no path diversity for EAR to gain over the SPR case. That is why we have compared EASP with EAR only for two-dimensional networks, and not for the linear network.

5.1.1 Linear Topology

Figure 5-1 shows the network lifetime for the topology in Figure 4-8. Lifetime of a network is defined up to the time at which the first node becomes energy exhausted. The first observation in Figure 5-1 is that for all protocols, the network life reduces with increasing network size. This is simply because with a given initial energy, with larger number of nodes more traffic is generated. With this increased energy demand for the routing shortens the network life.

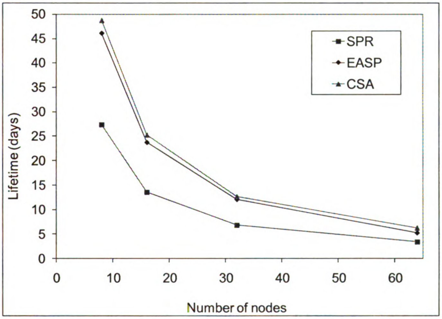


Figure 5-1: Network life versus network size

The second observation is that our proposed distributed EASP can significantly increase the network life over the SPR case. The gain in network life ranges from 55% for a network size of 64 nodes to 78% for a network size of 32.

The last observation is that EASP performs nearly as well as the centralized protocol with global knowledge of network topology and initial energy information. This indicates that the distributed swap algorithm with local information, as executed by the Emperor penguins [119] can deliver near best case performance as outlined in Section 4.9.

To understand the protocol dynamics better, we plot (in Figure 5-2) the standard deviation of nodal energy across the network at different points in time. For the SPR case, this quantity increases monotonically simply because each node, depending on its location (and routing burden), drains energy at a different rate.

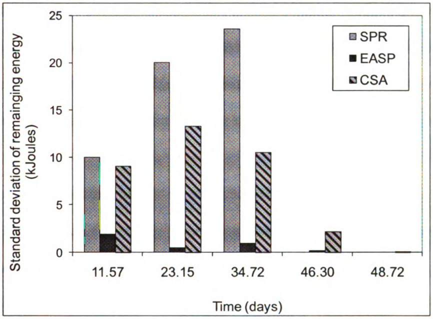


Figure 5-2: Spread of nodal energy with time

Number of nodes	SPR	EASP	CSA
8	0.01908	0.0203	0.0192
16	0.0365	0.0395	0.0371
32	0.0715	0.0773	0.0738
64	0.141	0.148	0.1509

Table 2: Energy per packet (Joule)

An interesting observation in Figure 5-2 is that although CSA provides the longest network life, its remaining energy variation is more than that of EASP at intermediate points in time. This can be explained by the fact that CSA performs swapping based on the global knowledge of the topology and the initial energy distribution. The goal for is it to derive a swap pattern (as explained for a three node network in Section 4.9.1) for lifetime maximization. It does not take intermediate energy conditions in to account and therefore no part of the protocol attempts to reduce the variance of energy across nodes during the intermediate points of operation. EASP, on the other hand, works based on the temporal energy distribution and explicitly attempts to minimize the variance by executing swaps. This accounts for smaller standard deviation numbers of EASP in Figure 5-2.

Average energy required to deliver each packet for all three protocols is reported in Table 2. This includes the energy required to transmit (3.8 mJoules), receive (0.57 mJoules), and perform node swap, which is 40.8 Joules.

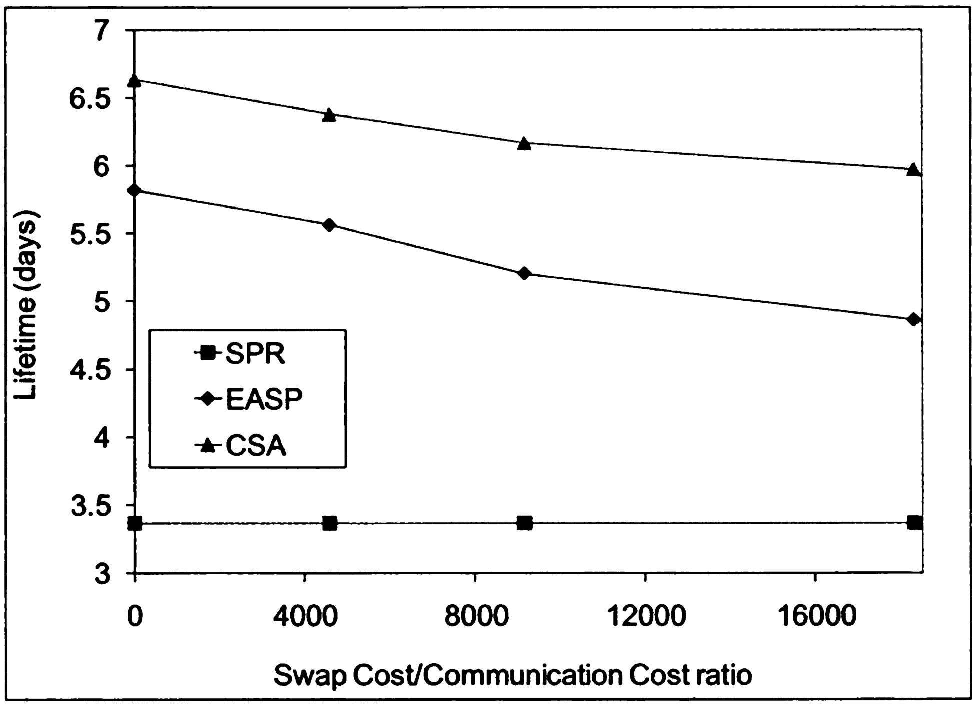


Figure 5-3: Impact of swap cost on network life

Note that the protocols that consume lower energy per packet do not necessarily provide longest network life. SPR, for example, has the lowest energy-per-packet because it does not have any swap cost. The network life for this protocol, however, is the shortest (see Figure 5-1).

Both EASP and the centralized algorithm have higher energy-per-packet costs because of their swap energy expenditure. In spite of this small increase in the energy-per-packet figures, both these protocols can deliver longer network life. Results in Table 2 also demonstrate that although each swap operation is four orders of magnitude more energy expensive compared to the packet communication cost, EASP, just like the centralized algorithm, is able to limit the number of swaps, so that there is an overall energy gain that extends the network life.

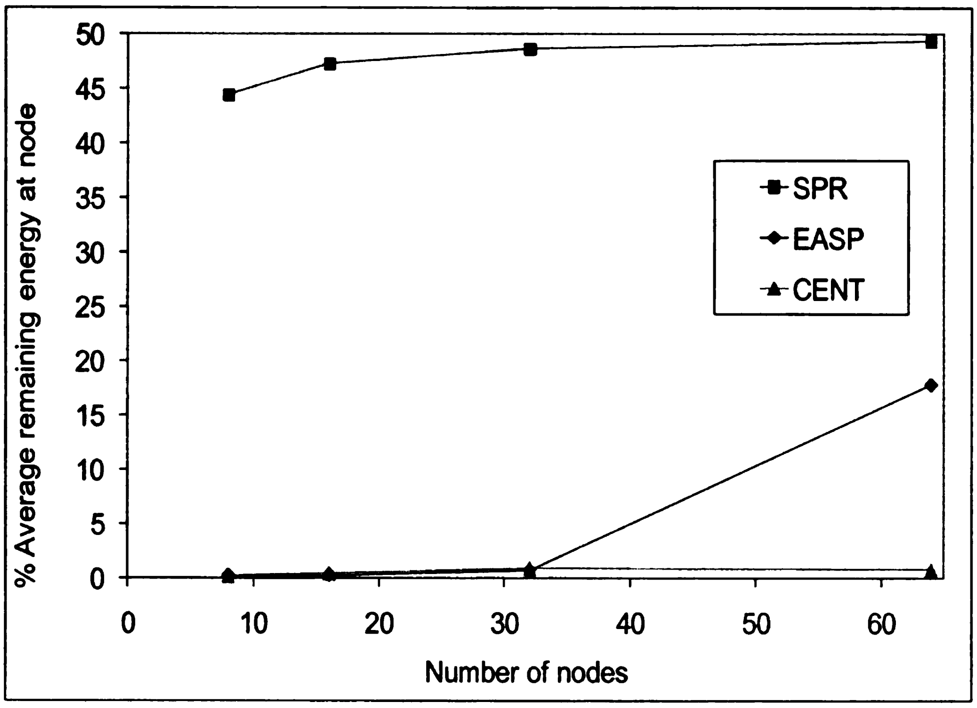


Figure 5-4: Average remaining energy at the end of network life

The impact of swap energy cost on the longevity of a network is shown in Figure 5-3. For a given communication cost, as the swap cost increases, the gain in lifetime gets offset by the swap energy consumption, and the network lifetime drops. It is notable that despite the relatively large cost of the physical swaps, our proposed EASP algorithm is able to significantly extend network life by successfully distributing the routing load.

Figure 5-4 shows the average amount of energy left in the nodes as a percentage of initial energy, with increasing network size. Large values of remaining energy for the SPR protocol indicate that when the network life ended because of energy exhaustion of at least one node, plenty of energy is still available in the network because of the poor energy distribution achieved by this protocol. Observe that for smaller networks, the remaining energy numbers are very close to zero for both EASP and the CSA protocol. With increasing network size, although the remaining energy remains more or less constant for the CSA protocol, it increases for EASP. This indicates that there is still room for improvement in the distributed EASP mechanism.

5.1.2 Two-dimensional Topology

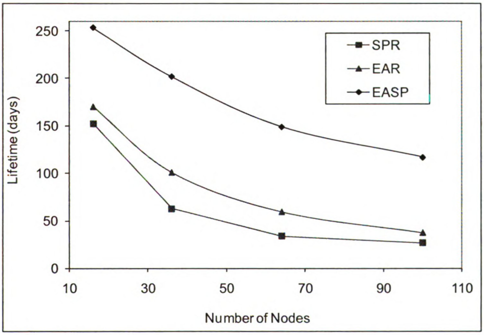


Figure 5-5: Network life versus network size

The distributed EASP algorithm, proposed in Section 4.6, has been also implemented for the two-dimensional networks as shown in Figure 2-1. Experiments reported in this section were carried out on networks with one base station that is connected to three sensor nodes which act as the gateway nodes. As for the data model, each sensor node periodically generates fixed size packet and sends it to the base station.

Performance of EASP (with shortest path routing) has been compared with SPR (with shortest path routing), and the constrained Energy Aware Routing (EAR), as described in Section 5.1. From the results in Figure 5-5, it is evident that like in linear networks, EASP can deliver better network life compared to both EAR and SPR protocols. Network life reduces with network size due to increased overall traffic load, as observed for the linear case in Figure 5-1.

Note that the EAR routing did not have any significant lifetime extension over the SPR protocol. The reason for this is two-fold. First, since EAR is not constrained to shortest paths, it was found to be taking longer routes compared to the SPR protocol. As a result, although the energy load was somewhat distributed by EAR, its energy-per-packet numbers looked worse than those for SPR. Higher energy per packet offsets part of the gain due to energy load distribution. The second and more important reason is that for nodes near the base station, not much energy distribution can be achieved using EAR. For example, the three gateway nodes will have to carry the entire routing burden irrespective of the load distribution achieved by EAR in the rest of the network. When these gateway nodes become energy-exhausted, the network becomes useless. For EASP, on the other hand, these three gateway nodes are not fixed. In other words, when a gateway node becomes somewhat energy depleted it simply swaps with a neighbor with better energy condition. In fact, a large number of nodes serve the role of energy-demanding gateway nodes at different points in time. This provides excellent energy load distribution across nodes, and the result is network life extension by a factor of 2 to 3.5.

Figure 5-6 shows the number of nodes alive at any given time for all the protocols under consideration. So far, we have defined lifetime as the time until the death of the first node. Another reasonable definition of network life would be the duration for which a certain percentage of the nodes remain alive, or the network remains connected with the base stations. To understand such behavior better, we conducted experiments on an 8x8 network with two base-stations and five gateways until all the gateway nodes to the base-station were energy-exhausted.

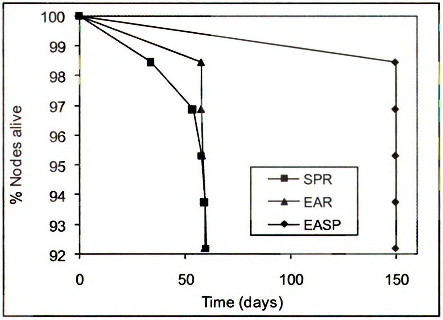


Figure 5-6: Remaining alive nodes with time

Results show that with SPR protocol, the first gateway node dies after approximately 20 days, which is earlier than EAR (40 days) and much earlier than EASP (150 days). After all 5 gateway nodes are energy-exhausted, the sensor network is considered to be disconnected from the base station. Therefore, in Figure 5-6, network disconnection time for a given protocol can be indicated by the time up to which 92% (59 non-gateway nodes out of 64 nodes) of the nodes are alive. For SPR, EAR and EASP, such disconnection times are found to be 60.2 days, 59.8 days and 150 days. With network disconnection time as the metric, the performances of SPR and EAR are almost identical, but EASP outperforms both of them by approximately two and a half folds.

5.2 Detailed Packet Level NS2 Simulation

Using the NS2 network simulator, EASP experiments were run on a tiled sensor field with one mobile sensor per tile as shown in Figure 4-5. All EASP modules (described in Section 4.6 and Figure 4-2) including the EASP Neighbor Discovery, Swap Evaluation and Partner Selection, Swap Initiation and Conflict Resolution, and the Physical Swap with Route Handoff were implemented within NS2. Sensor transmission range and the tile dimension were chosen to be $30m$ and $10m \times 10m$ respectively. These dimensions ensure that the nodes in two adjacent tiles can always communicate irrespective of their individual locations within their respective tiles. As for the data model in the baseline case, each sensor generates a 512-byte packet every second, destined for the base-station. The effects of packet size variation on EASP performance are reported later in this section.

IEEE 802.11 [136, 137] is used as the medium access control (MAC) protocol. In order to emulate energy-aware sensor MAC protocols [123, 124] with zero idling, carrier sensing and overhearing energy expenditure, we set those expenditures to be zero for our employed version of the 802.11. As the communication cost, we account for the energy drainage incurred only by the transmissions and receptions.

As for the mobile sensors, we have modeled the experimental parameters based on a Swarm Capable Autonomous Vehicle (SCAV) [138] prototype that was designed around a MICA2 MOTE [34] sensor and a LEGO Mindstorm robotic vehicle [120]. The Mindstorm robots are powered by two Lego Technic 43362 motors [135]. The motors rotate with a speed of 219 rpm while producing a torque of 2.25N.cm. Assuming an average swap distance of 10m (tile size), the energy required to drive the motors for a swap operation is 37.1 Joules. Since two mobile sensors are involved in a swap, the total cost of each swap operation is 74.2 Joules. An initial energy of 81 KJoules (corresponds to two AA batteries used by the MICA2) was assigned for each sensor node.

5.2.1 Evaluated Protocols

The following protocols have been evaluated for performance comparisons.

Shortest Path Routing (SPR): During the network initialization, two minimum spanning trees, each rooted at each base-station, are created. Each node forwards its packets through the tree that provides a shorter hop-count to the tree root. No swaps are executed in this protocol.

Energy-Aware-Routing (EAR): We have implemented a constrained-routing [44] algorithm which maximizes the minimum remaining energy along any route from a sensor to its destination base station. Like in the SPR protocol, spanning trees are created such that the tree for each base station keeps track of the minimum energy node that has been encountered along a branch of the tree. Each sensor picks a next hop node such that it maximizes the minimum residual energy along the path to the base-station.

The EAR protocol is generally able to perform better than the SPR when there is sufficient route diversity in the network. Since no such diversity is present in linear networks with single base station (see Figure 4-8), we have not evaluated EAR for those networks; it has been evaluated only for two-dimensional networks.

Energy Aware Swap Protocol (EASP): The EASP protocol, as described in Section 4.6, was implemented with the parameterization as outlined in Section 5.2. Data routing for these experiments are always performed using the shortest paths. As explained in Section 4.7.1, a node executing the EASP-RE version of the protocol requires the residual energy of its swap partner to be higher than its own by a percentage threshold value (Th_{swap}). This implies that as the nodes approach the end of life stage, they satisfy the swap threshold requirement more easily and tend to swap more frequently. This can nullify the gain in lifetime achieved during the earlier stages.

In order to prevent frequent swaps at the final stages, we ensure that nodes only swap when there is enough life left to recover the energy cost of the swap operations. We refer to this *End-of-Life-Limit*. Note that such a limit is not required for the EASP-LT version of the implementation as it already takes energy cost of swap into account in its secondary condition as mentioned in Section 4.7.1.

5.2.2 Performance Results

5.2.2.1 Benefits of Swap in Linear Topologies

Figure 5-7 shows the lifetime for the linear network in Figure 4-8 with varying number of nodes.

The first observation in is that for all protocols, the Figure 5-7 network life reduces with increasing network size. This is simply because with a given initial energy per node, with larger number of nodes more traffic is generated. With increased energy demand for routing, the network life is shortened. The second observation is that the distributed

EASP protocols can significantly increase the network life over the SPR protocol. The gain in network life ranges from 74% for a network size of 8 nodes to almost 80% for a network size of 32.

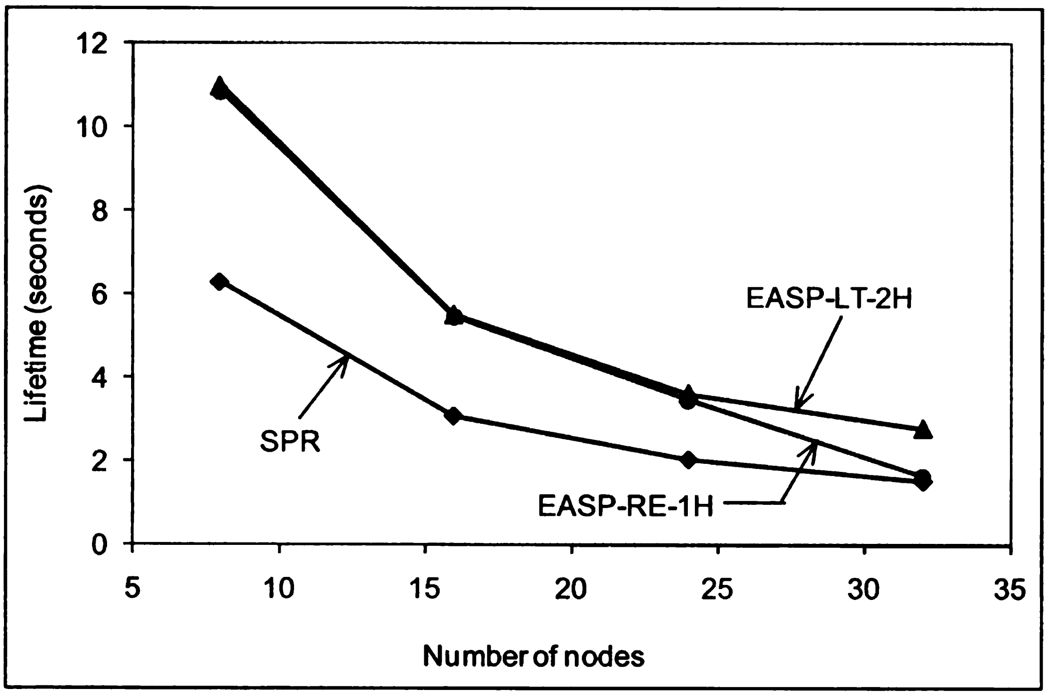


Figure 5-7: Lifetime for different protocols in linear topology

We have reported the performance of EASP-RE-1H and EASP-LT-2H, which are the best performing EASP protocols in their respective categories with 1-hop and 2-hop energy and EDR information. The lifetime gain achieved by EASP-RE-1H and EASP-LT-2H are very similar in small networks. However in larger networks, while EASP-LT-2H is able to sustain the gain, the EASP-RE-1H protocol offers only marginally better network life compared to that of SPR. With EASP, elongation of network life fundamentally depends on swaps for periodically replacing energy-drained nodes close to the base station by energy rich nodes away from the base station. As the network size increases, the absolute life of the network decreases, thus for a given initial energy the number of swap opportunities also decreases. With fewer swap opportunities available, the quality of the swap becomes more crucial for large networks. With 2-hop energy and EDR information and taking the swap cost into account while performing swap evaluations, the EASP-LT-2H protocol is able to perform better quality swaps compared to its EASP-RE-1H counterpart with lesser amount of information. This explains why EASP-LT-2H outperforms EASP-RE-1H in large networks.

To summarize, the results in Figure 5-7 indicate that the proposed biologically inspired energy aware swap strategies do elongate the network life for one-dimensional sensor networks.

5.2.2.2 Swap in Two-dimensional Topologies

In addition to SPR and EAR, we compare the performance of EASP with a centralized implementation of MAPC (Mobility Assisted Power Control) solution [84, 85, 139], in which the core idea is to move the intermediate nodes on a route in such a way that they are evenly placed along the line from source to destination as mentioned earlier in Section 2.4.1. With transmission power control, such a configuration has been shown to minimize communication cost [84]. In essence, MAPC attempts to reduce the EDR values associated with nodes using controlled mobility. EASP, on the other hand distributes the energy drainage more evenly across nodes. Note that the MAPC solution is independent of the underlying routing strategy. Therefore, MAPC with shortest path routing implies data is routed to the base-stations along routes with similar hop counts as in SPR. In other words, the packet delivery ratio, average hop count, and delay should remain the same as in SPR. The metrics that are affected by the intermediate node alignment and the resulting transmission power optimization are: 1) energy cost incurred per packet, and thereby 2) network lifetime. In this sub-section, we show MAPC results where they are different from SPR. It is important to note that the presented results of MAPC reflect its best performance by using load-balanced shortest hop routes through a centralized implementation. Also note that transmission power control is used only in MAPC protocol, whereas for SPR, EAR and EASP protocols the maximum transmission power is used.

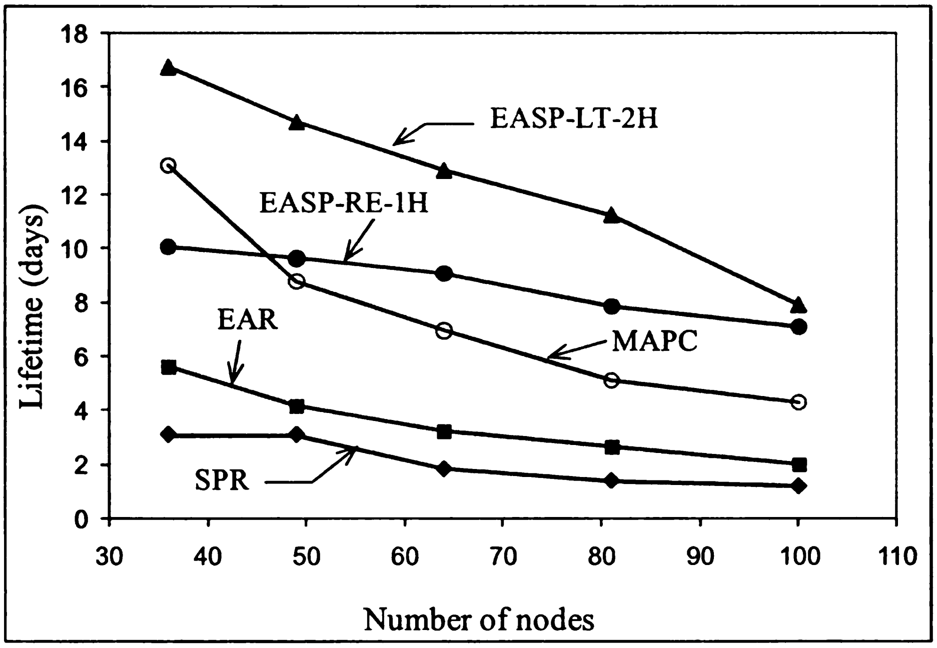


Figure 5-8: Network lifetime of 2-dimensional networks

Network Life: Experiments on two-dimensional tiled networks (see Figure 4-5) were carried out with two base stations, where each base station is within the transmission range of two neighboring tiles. As a result, four mobile sensors act as gateways between the sensor field and the base stations. As for the data model, each sensor periodically generates fixed size packets destined to the closest (shortest hop) base station.

Performance comparison between SPR, Energy Aware Routing (EAR) [44], MAPC[84, 85], and EASP protocols is presented in Figure 5-8. Note that EASP, MAPC, and SPR work with sensor-to-base station shortest path routing, whereas EAR chooses energy constrained routes based on the mechanism described in Section 5.2.1. In Figure 5-8, it is evident that like in linear networks, EASP can deliver better network life compared to both EAR and SPR protocols. Network life reduces with increase in network size due to increased overall energy load as observed for the linear case in Figure 5-7.

Observe that the EAR routing extends network life by a small amount over the SPR protocol as it balances the load across the gateway nodes. However, the relative gain is much smaller than that of the EASP protocols. There are two reasons for this. First, since EAR is not constrained to shortest paths, it was found to be taking longer routes compared to the SPR protocol. As a result, although the energy load was somewhat distributed by EAR, its energy-per-packet numbers (shown in Table 3) are worse than those for the SPR case. Higher energy-per-packet expenditure offsets part of its gain due to the achieved energy load distribution. Note that the longer routes also increase the packet delivery latency as discussed later in this section.

The second and more important reason is that for nodes near the base stations, not much energy distribution can be achieved using EAR. The four gateway nodes for two base stations will have to carry the entire routing burden irrespective of the load distribution achieved by EAR in the rest of the network. When these gateway nodes become energy-exhausted, the network becomes disconnected. For EASP, on the other hand, when a gateway node becomes somewhat energy depleted it simply swaps with a neighbor with better energy condition. In fact, a large number of nodes serve the role of the energy-demanding gateway nodes at different points in time. This provides excellent energy distribution across nodes, and that results in network life extension by a factor of

3.25 to 8. Due to the same set of reasons outlined in Section 5.1.1 for the linear networks, the EASP-LT-2H protocol outperforms EASP-RE-1H.

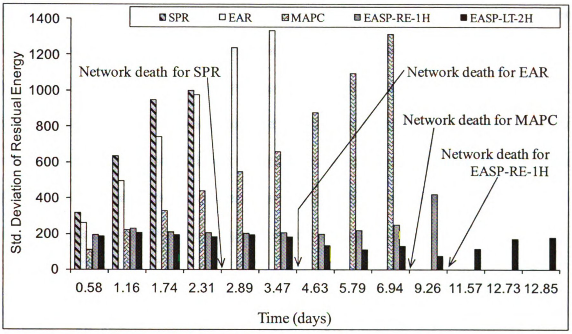


Figure 5-9: Residual energy variance in a 8x8 network

In Figure 5-8, it can be observed that MAPC performs better than SPR by a factor of 2.8 to 4.2. By reducing the packet transmission energy on the nodes along a route, and balancing the load across the gateway nodes, the gateway nodes in MAPC are able to survive longer compared to SPR. The lifetime improvement seen by MAPC over SPR is higher than what has been reported in [84] and [85] primarily due to our centralized implementation. Even so, EASP consistently performs better than MAPC suggesting that EASP achieves better energy load distribution which is substantiated by the residual energy variation across nodes, as shown in Figure 5-9.

Residual Energy: In Figure 5-9, we report the standard deviation of the remaining nodal energy with time. For the SPR protocol, this quantity increases monotonically because the nodes on the routing hot-spots becomes increasingly more energy depleted compared to the other nodes, and thus the variance in the residual nodal energy increases. For the EAR protocol, similar trend of increasing standard deviation can be also observed. This is due to the fact that the energy depletion rates are variable over nodes depending on their routing burden, and constant over time due to the absence of any swaps. As a result the relative difference in residual energy across the nodes monotonically increases over time. MAPC also shows the same variance trend as SPR and EAR. However, the variance is at a smaller scale because of the lower energy consumed per unit time by nodes. Also, note that the final standard deviation values reached by EAR and MAPC around network death are the same. MAPC takes longer to reach that value.

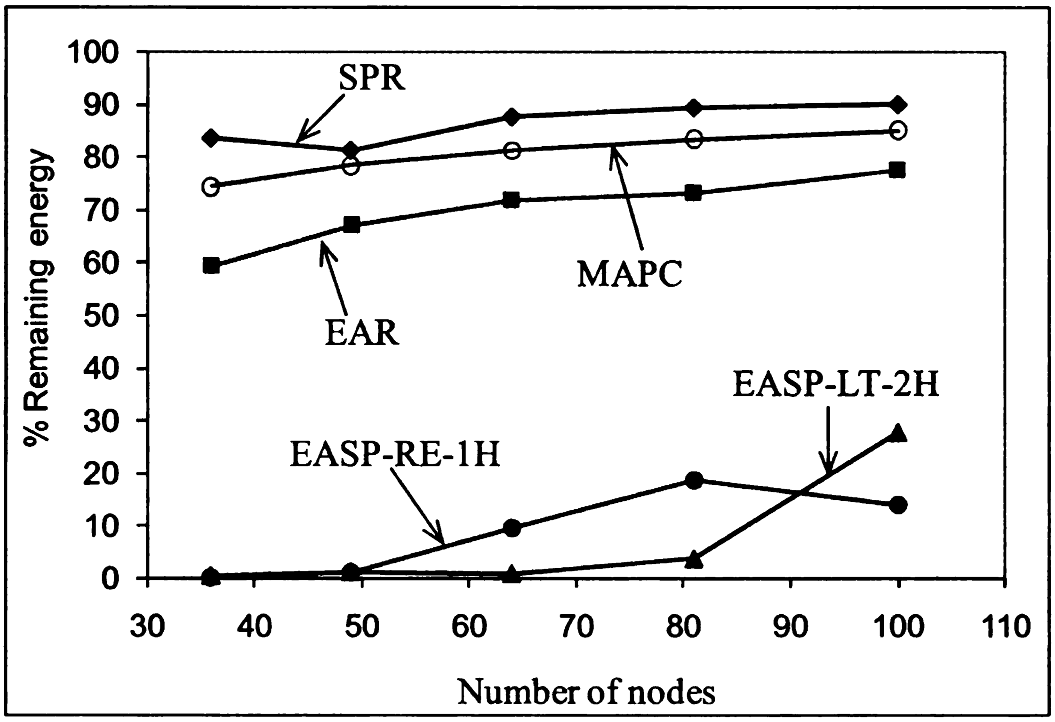


Figure 5-10: Average remaining energy at the end of network life

The EASPs, on the other hand, are able to maintain relatively smaller standard deviation due to their swap operations. Towards the end of the network life, both the EASP protocols experience slightly increasing standard deviation due to reducing swap rates caused by the *end-of-life-limit* effect as explained in Section 4.7.1. This decrease in the swap rate is caused by the fact that there is not enough lifetime left in the network to recover the swap energy cost following each swap.

Figure 5-10 reports the average remaining energy after the network life expiry, as a percentage of the initial energy. Large remaining energy for the SPR, MAPC, and EAR protocols indicate that plenty of energy remains available in the network when at least one node becomes energy exhausted. These undesirable effects are due to the lack of energy distribution in SPR and MAPC, and long routes in EAR. Although, SPR and MAPC use shortest path routes, MAPC is able to use more energy in the network because routes are more load-balanced. Also, although MAPC outlives EAR, remaining energy is lower for EAR primarily because EAR uses longer routes and hence incurs higher energy cost for delivering packets. For the EASP protocols, however, the remaining energy numbers are relatively much smaller. For example, in a 6×6 (36 nodes) network, only 3.7% of the nodal energy remains after the event of the first nodal energy exhaustion. With increasing network size, the remaining energy for EASP-LT-2H slightly increases. This slight performance degradation is caused due to fewer swap opportunities as a result of shortened network life for larger networks.

Swap Counts: Figure 5-11 shows the swap count per node during the defined network life. As anticipated, the protocol EASP-LT-2H, equipped with up to 2-hop energy and EDR information and taking the estimated life time information for the swap decisions, swaps much less aggressively, but more judiciously, compared to its 1-hop counterpart. This trend is valid for all network sizes, although the number of swaps turns out to be insensitive to the network size itself. Very little additional room for swap reduction for this protocol appears to be the reason for this.

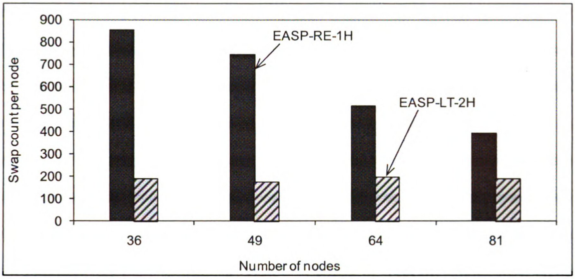


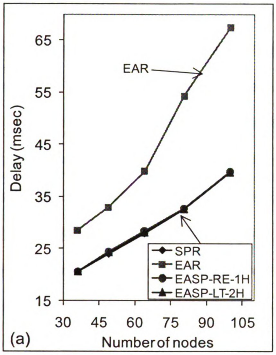
Figure 5-11: Per node swap count for EASP protocols

For the 1-hop EASP-RE-1H protocol, however, the swap count reduces monotonically with increasing network size. Due to higher network traffic and energy load, and the subsequent reduction in network life accounts for this trend of swap frequency reduction.

Network size	SPR	EAR	MAPC	EASP-RE-1H	EASP-LT-2H
36	16.9	23.2	6.24	31.55	18.9
49	19.3	25.75	7.82	32.6	21.3
64	21.7	28.8	8.64	31.6	24.4
81	24	34.9	10.44	33	27.25
100	27	36.3	11.15	38.5	28.95

Table 3: Energy per packet (in milliJoules)

Energy per Packet: The average energy required to deliver each data packet is reported in Table 3. The energy per packet includes expenditures for transmission (3.4 mJoules for maximum power), receptions (2.06 mJoules for maximum power), and for the physical node swaps, which is 37.1 Joules for the swapping nodes. Note that the protocols with lower energy-per-packet do not necessarily provide longest network life. The MAPC and SRP protocols, for example, have relatively lower energy per packet values because of min-hop routes and zero swap costs. MAPC has lowest energy per packet value on account of its node placement strategy and power control for reducing communication cost. For the EASP protocols, the huge energy expenditure incurred by the physical swap operations and the conflict resolution protocol as outlined in Section 4.7.2 increases the per packet energy expenditures. However, their longer network life allows much higher data delivery, thus the energy cost incurred for swaps is spread over a larger number of packets. This offsets the swap energy costs and brings the energy per packet costs down to the vicinity of that for the SRP protocol. With smaller number of swaps (see Figure 5-11) compared to the EASP-RE-1H, the energy-per-packet figures for the EASP-LT-2H are very close to those for the SRP protocol. But interestingly, in spite of their higher energy per packet numbers, both the swap based protocols deliver longer network life as reported in Figure 5-8. Finally, the energy aware routing protocol (EAR) has the highest energy per packet cost because of its longer routes. Additionally, because the number of packets delivered with EAR is lower than the EASP protocols, and therefore, the cost of maintaining energy aware paths is spread over a smaller number of packets which leads to higher energy cost per packet. As pointed out earlier, MAPC extends lifetime by using controlled mobility to achieve lower EDRs, while EASP reduces the variance in nodal energy consumption. Note that it is possible for EASP and MAPC to be employed simultaneously in which case higher lifetime can be expected than the individual strategies alone.



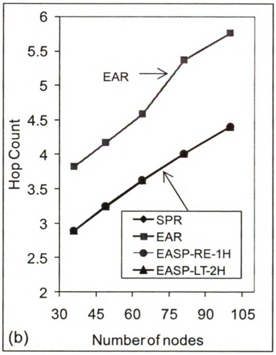
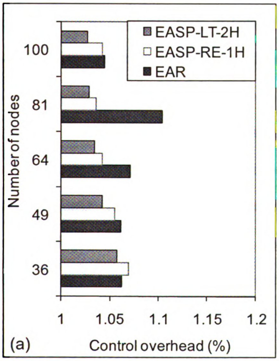


Figure 5-12: a) End-to-end delay and b) route length



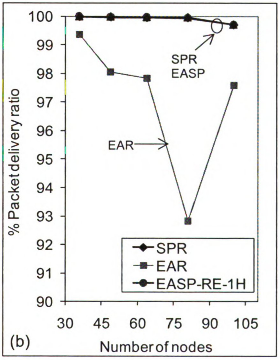


Figure 5-13: a) Control overhead, and b) Packet delivery ratio

Packet Delivery Delay: The end-to-end delay performance of the EASP protocols is shown in Figure 5-12:a. EAR demonstrates the highest delay because of its longest routes as evidenced in Figure 5-12:b. Since with both EASP-RE-1H and EASP-LT-2H packets are always delivered along the shortest path, their delay and route lengths are same as those for the best case SPR protocol. Note that the route handoff part of the EASP framework, as described in Section 4.6.1.4, ensures that in spite of the physical swap operations, the data routes to base stations never change, and therefore the shortest paths are maintained throughout the network life.

Protocol Control Overhead: Figure 5-13:a shows the percentage control overhead which is defined as the number of generated control packets normalized by the total network traffic including the data packets. For EAR, the overhead is contributed mainly by the neighbor discovery packets with residual energy information used for constrained route discovery [44]. The EASPs have two sources of control overhead. First, similar to EAR, they also incur neighbor discovery with EDR and residual energy information. Second, they use the swap and handoff related control packets as described in Section 4.7.3.

Results in Figure 5-13:a indicates that compared to the EAR protocol, the EASPs have smaller control overhead, which is in the vicinity of on 1% or so, and that is in spite of the additional control overhead of the swap-related protocols. This is because the additional swap control overhead gets distributed due to the much higher number of delivered packets with the EASPs, and that keeps the percentage overhead low. Also, the control overhead of EASP-LT-2H is lower than EASP-RE-1H for a) its longer life and b) fewer swaps as observed in Figure 5-11.

Results in Figure 5-13:b demonstrates that compared to the EAR protocol, the EASPs offer better packet delivery rates similar to that of the shortest path routing protocols.

As the network becomes larger, the delivery rates for SPR and EASP show a decline due to mild congestion-related packet drops. In EAR, longer routes incur more transmissions which give rise to congestions earlier than the SPR and EASP protocols. This leads to the lower packet deliveries for EAR even for smaller networks.

Based on the observed results so far, it can be concluded that the 2-hop EASP-LT-2H protocol consistently outperforms its 1-hop EASP-RE-1H counterpart, and therefore the next set of results will include only EASP-LT-2H as the best case EASP performance.

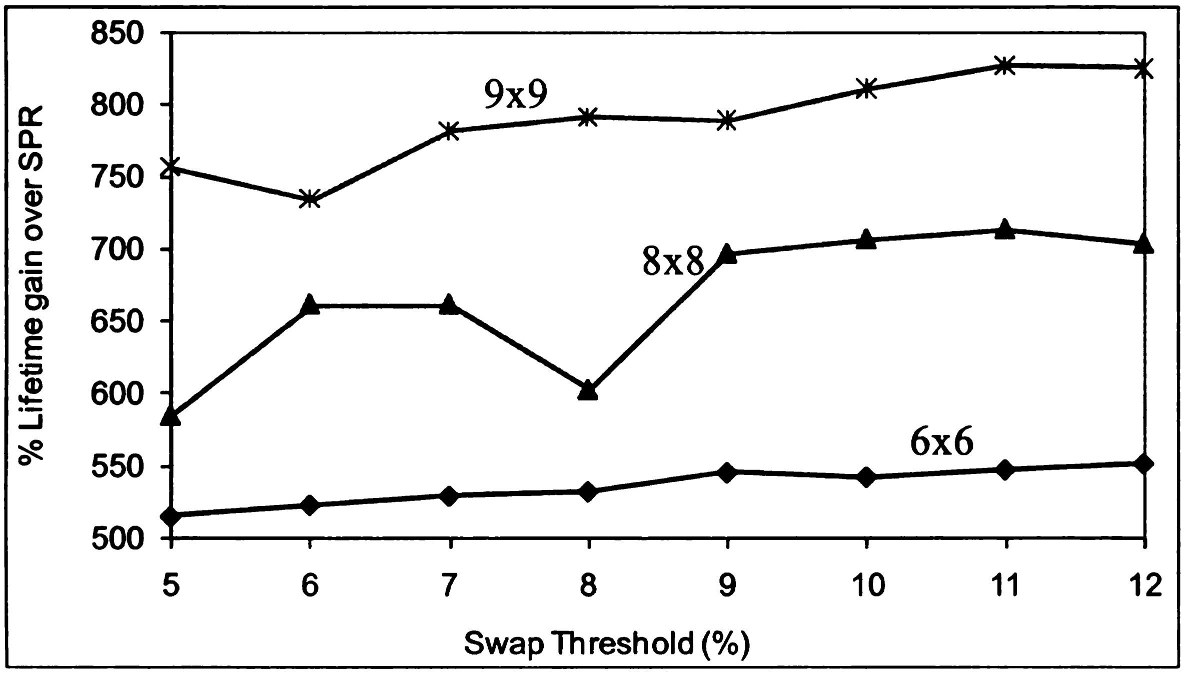


Figure 5-14: Impact of swap threshold on EASP performance

Sensitivity to Swap Thresholds: Figure 5-14 shows the threshold (Th_{swap}) sensitivity of the EASP-LT-2H protocol. With very small Th_{swap} values, the protocol was observed to swap very aggressively, and the lifetime benefits of the swaps were seen to have been offset by the hefty energy costs of the physical sensor moves. With very large Th_{swap}, on the other hand, sluggish swaps were observed to defeat the basic goals of the EASP framework. It was experimentally found that the EASP-LT-2H protocol is relatively insensitive to Th_{swap} for a reasonably large window of the threshold – that is approximately from 5% to 12% for a wide range of network sizes. This result can be used as a guideline for swap threshold selection for the EASP process.

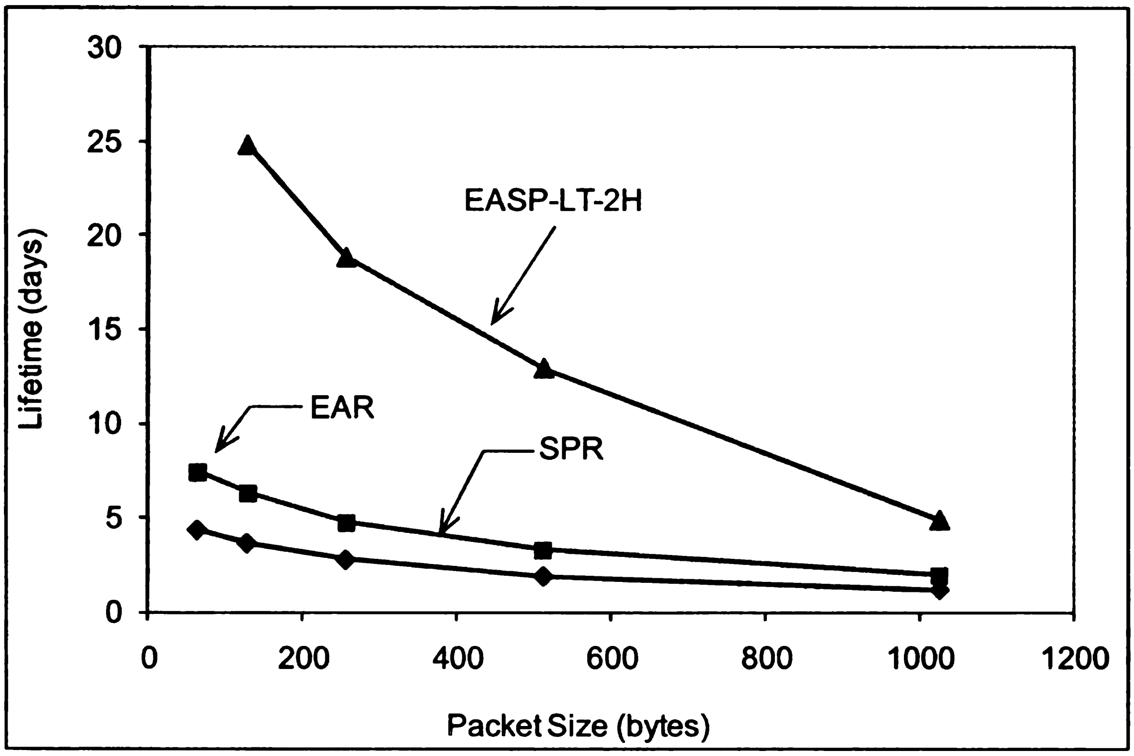


Figure 5-15: Impact of data packet size on performance

Impacts of Packet Size: Figure 5-15 shows the impacts of varying packet size on the lifetime in an 8×8 (64 sensors) network. With a fixed packet rate, increasing packet size increases the overall energy load and, as demonstrated in Figure 5-15, reduces the absolute overall network life. The relative gain of EASP-LT-2H over SPR, however, remains quite high across the entire range of experimented packet sizes. The relative gain ranges from approximately 700% for the 128 byte packets to 450% for 1024 byte packets. To summarize, although the absolute network life depends on the packet size, the lifetime advantage of EASP varies from about 7-fold improvement for smaller packets to over

4.5-fold improvement for larger packets. These results indicate the suitability of the EASP swap framework for a wide range of target sensing and similar applications with different packet size requirements.

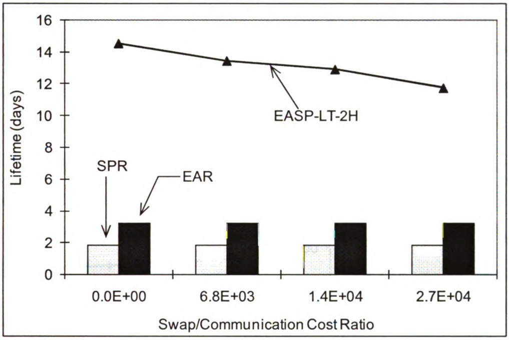
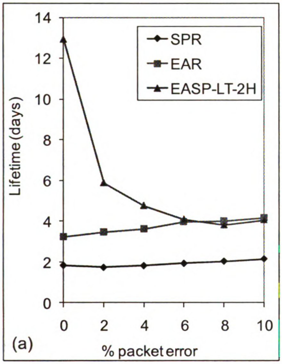


Figure 5-16: Impact of swap cost to communication cost ratio

Sensitivity to Relative Swap and Communication Costs: Swap experiments were conducted on the 8×8 topology while varying the energy cost of a physical swap operation from zero to larger values up to four orders of magnitude higher than the communication energy cost. The communication cost was kept constant during these experiments. Results in Figure 5-16 indicate that increasing the swap cost with respect to the communication cost leads to lower lifetime gain. For instance, the lifetime shrinks from 14 days with zero swap cost to around 12 days with the highest cost used. However, the notable result is that even with up to four orders of magnitude relative swap costs, the EASP system is able to deliver significant amount of lifetime elongation compared to the plain SPR and EAR routing mechanisms. It turned out that the consideration of the swap cost in EASP-LT-2H's expected lifetime computation during the swap decision process is responsible for its desirable insensitivity to such large swap costs.



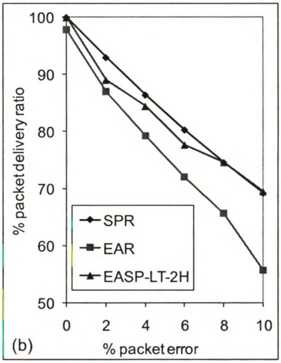


Figure 5-17: Error Sensitivity a) Lifetime, and b) Delivery ratio

Sensitivity to Packet Errors: To study the impact of channel errors on the performance of EASP, we conducted experiments on the 8×8 topology with packet error rates in the range of 0 to 10%. The error in these experiments represents the error seen above the MAC layer after the re-transmission attempts failed. Results in Figure 5-17:a indicate that the lifetime for SPR and EAR increase marginally as the error rate increases. This is mainly due to the fact that the lost data packets, caused by channel errors, do not have to be routed all the way to the base station. This results in routing energy savings - resulting in higher lifetime, though at a cost of reduced packet deliveries as seen in Figure 5-17:b. However, in case of EASP-LT-2H, the presence of error adversely impacts the swap protocol (see Section 4.7) - resulting in fewer swaps. The higher the error, the fewer the swaps and hence the lower the lifetime. These results underscore the need for an error-free channel or sufficient protection at the link layer to ensure the delivery of swap control packets to get significant lifetime gain with the EASP paradigm.

5.3 Summary and Conclusions

In Chapter 4 and Chapter 5, a mechanism for extending network life by introducing a biologically inspired energy-aware mobility in wireless sensor networks has been presented. The concept has been inspired by a natural grouping behavior of Emperor penguins for conserving body heat during the winters. Based on this concept, we provide a distributed sensor mobility framework that works based on localized energy information and makes sure that the routing load across sensor nodes in a network is evenly distributed to collectively extend the network life. A performance upper bound for linear networks was derived and compared with the performance of the distributed EASP protocol. Proof of concept and long term lifetime gain of EASP was established using a simple C-based simulator. Experimentally, using an extensively built simulation model in NS2, we have evaluated two versions of the proposed framework.

Experimental results demonstrate that using the distributed algorithms for biologically inspired controlled node mobility, it is possible to significantly extend the network life. We also show that this holds in situations where the energy cost of node mobility is modeled as up to four orders of magnitude larger than the communication cost for each packet.

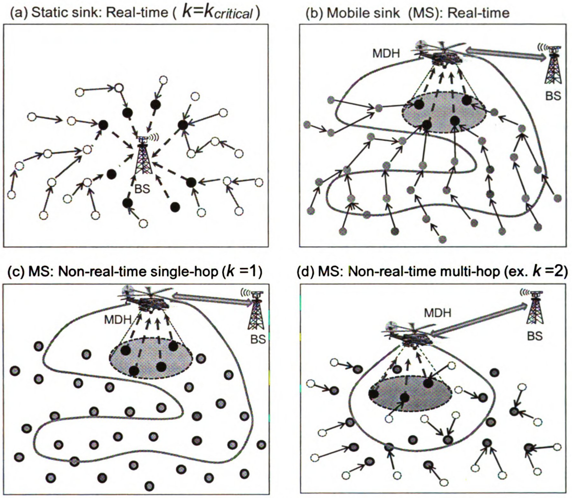
Chapter 6. Generalized Framework for Joint Trajectory and Routing Optimization

The approach proposed in Chapter 4 is applicable to networks composed of nodes that are capable of limited mobility. However, a large percentage of sensor network deployments are expected to be static in which case node mobility based lifetime elongation techniques do not apply. In this chapter, we consider such static networks and address the data collection problem using the mobile sink approach as outlined in the contribution summary presented in Section 3.2.

6.1 Sensor Network Data Collection

Data collection delay and network life are often considered the two most important issues while designing an embedded sensor network as discussed in Chapter 1. As shown in Figure 6-1:a, the best delay can be obtained when data from the sensors is uploaded to a static sink in real-time using multi-hop routes. This approach, however, can suffer from limited network life due to the traffic hotspots at the nodes around the sink.

The key idea behind mobile sink, also referred to as Mobile Data Harvester (MDH), is that the nodes around the sink change over time and therefore the energy drainage due to routing occurs more uniformly, thereby extending the network life. With mobile sinks, the protocols for packet routing and sink navigation need to be designed based on a clear tradeoff between the data delivery delay, network lifetime, and the sink exposure time. As shown in Figure 6-1:c, for applications with large delay tolerance, a mobile sink can collect data from individual sensors by following a trajectory spanning all the network nodes. Locally sensed data is buffered at each node until the sink comes in contact. Therefore, the delay experienced by the data packets is in the same timescale as the round trip time of the mobile sink. While incurring a large delay, this mode of data collection provides the most energy efficient solution since the transmission energy expenditure is incurred only for a single hop. Due to its best energy and the worst delay performance, this single-hop collection can be considered to be the other extreme of the traditional static sink scenario as illustrated in Figure 6-1:a, which offers the best delay and the worst energy solution.


(a) Static sink: Real-time ($k=k_{critical}$)
BS
(b) Mobile sink (MS): Real-time
MDH
BS
(c) MS: Non-real-time single-hop (k =1)
MDH
BS
(d) MS: Non-real-time multi-hop (ex. k =2)
MDH
BS

○	Sensor node never in direct contact with MS.	●	Node in direct contact with BS/MS currently
•	Node in direct contact with MS at some point, forwards data immediately to the MS at its current location	◉	Designated Gateway (DG) – Comes in direct contact with MS at some point in the trajectory, buffers generated or routed data till contact

Figure 6-1: Different routing scenarios in static and mobile-sink-based data collection

A controlled multi-hop (with a hop bound of *k*) data collection strategy, as shown in Figure 6-1:c, can be used for striking a desirable balance somewhere between those two solution extremes. In this approach, the sink collects data only from a subset of nodes (referred to as *designated gateways or DGs*) which aggregate and route data from other sensors that do not make direct contact with the mobile sink. The aggregated data is buffered at the DGs until they come in contact with the mobile sink. For a given network, higher the number of Designated Gateways, lower the overhead of multi-hop routing (small *k*) and hence higher the delay and lower the routing energy expenditure. In other words, by controlling the hop bound *k*, it is possible to strike a balance between the packet delivery delay and the energy expenditure. For any given topology, $k=1$ is equivalent to single hop data collection and $k = k_{critical}$ results in the static sink scenario. The values of *k* between 1 and $k_{critical}$ can be used for achieving the desired delay or lifetime performance.

6.2 Application Model and Problem Statement

Sink availability in the sensor field and data collection duration can generally be represented using a duty cycle model as shown in Figure 6-2. Consider the Mobile Data Harvester (MDH) in Figure 6-1:d, that traversing a sensor field to collect data through a number of pre-selected Designated Gateways. The time required for collecting all data from the field during each trip is referred to as the *trip collection-time* t_{coll}. The periodicity of the MDH or the duration of the collection cycle is τ. The objective is to minimize t_{coll}. For stable operation, t_{coll} should be less than or equal to τ. The quantity t_{coll} also represents the *flying time* or *exposure time* that a UAV-style mobile sink has to withstand during each collection cycle. With constraints on *flying time* (or *exposure time*) of a UAV (or UGV [96]) sink, the collection framework will have to incorporate controlled multi-hop routing for shortening the quantity t_{coll}.

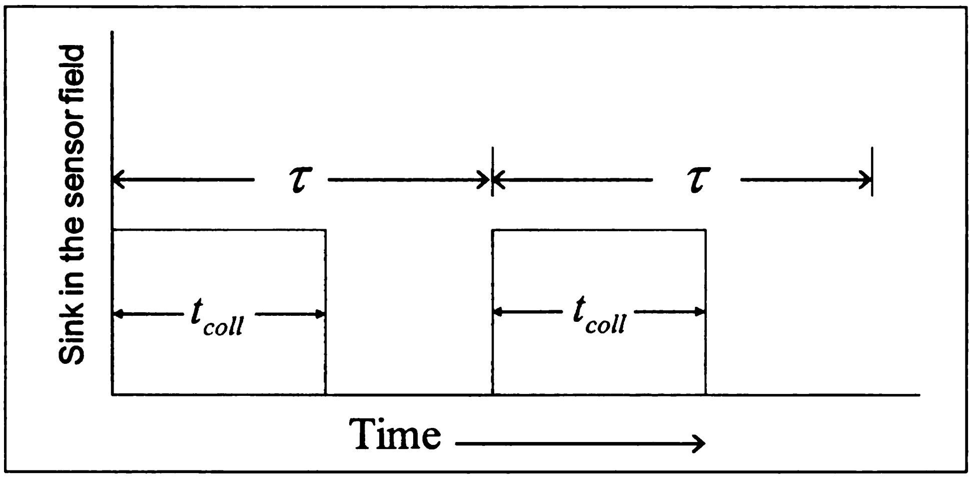


Figure 6-2: Sink periodicity and Data Collection Delay

Delivery delay for a packet is specified as the interval between when it is generated at a sensor and when it is collected by a sink. Depending on when a packet is generated within a *collection cycle*, it can face a *delivery delay* anywhere between 0 and τ, resulting in an average of $\tau/2$. The *delivery delay* does not depend on *trip collection-time* t_{coll} except in applications where $\tau = t_{coll}$ which represents a mobile sink that makes back-to-back trips to collect data. Note that the collection cycle τ represents the upper bound of the collection-time t_{coll} for stable operation. For a given MDH technology such as UAV or UGV, the collection cycle τ is assumed to be constant, and it represents the maximum duration between refueling.

Data collection can be real-time when the incurred delay is primarily in the time-scale of packet forwarding for applications such as catastrophic event and intrusion detections. Non-real-time collection, on the other hand, is categorized by delay in the same time-scale of sink mobility, and is applicable for monitoring style applications such as environmental data monitoring.

While the static sink solution in Figure 6-1:a caters only to real-time applications, both real-time (Figure 6-1:b) and non-real-time scenarios (Figure 6-1:c and Figure 6-1:d) can be accommodated by the mobile sink approach. From a sink mobility standpoint, the applications are categorized in Figure 6-3. For all data collection applications, maximizing the network lifetime is the primary objective. However, depending on the nature of the application, there might be secondary objectives as shown in Figure 6-3.

Continuous Data Collection (CDC) ($\tau = t_{coll}$):

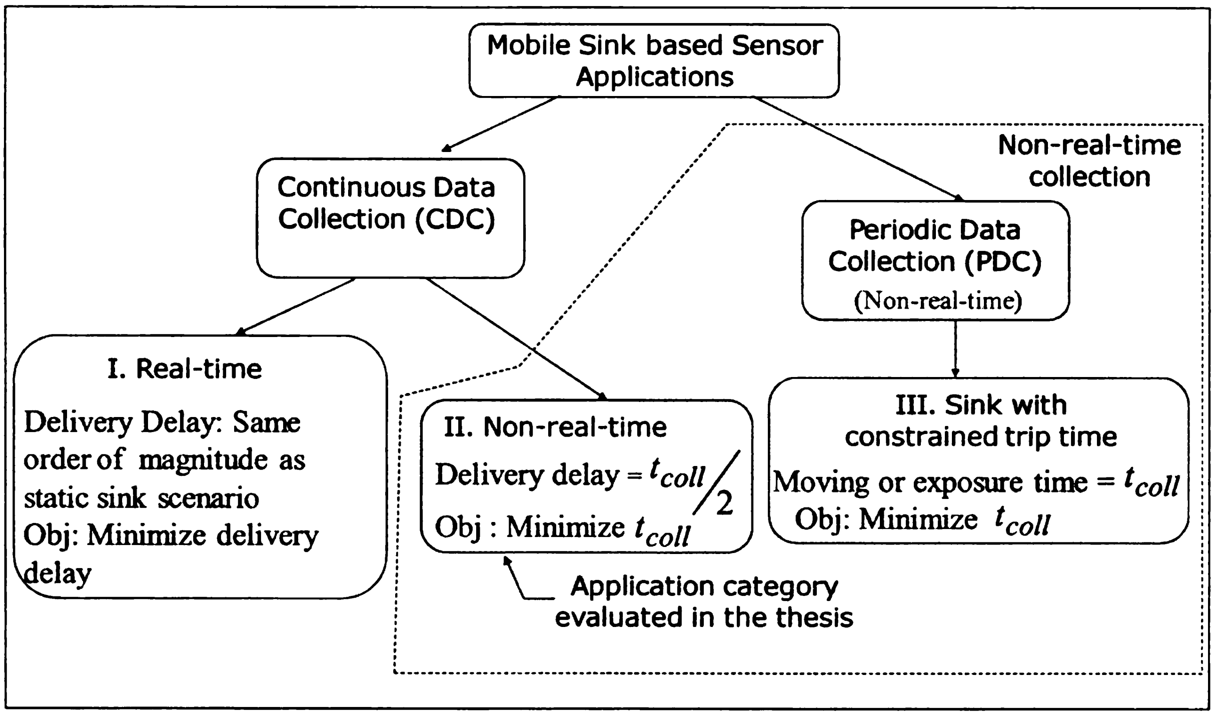

Mobile Sink based Sensor Applications
Non-real-time collection
Continuous Data Collection (CDC)
Periodic Data Collection (PDC) (Non-real-time)
I. Real-time
Delivery Delay: Same order of magnitude as static sink scenario
Obj: Minimize delivery delay
II. Non-real-time
Delivery delay = $t_{coll}/2$
Obj : Minimize t_{coll}
III. Sink with constrained trip time
Moving or exposure time = t_{coll}
Obj: Minimize t_{coll}
Application category evaluated in the thesis

Figure 6-3: Characterization of the applications with mobile sinks

The mobile sink is continuously present in the sensor field ($\tau = t_{coll}$) collecting data either in real-time or in non-real-time mode. In the real-time case, as the mobile sink moves along its trajectory, data from the entire network is routed to the sink at its current location. Conceptually, this situation incurs delivery delay in the same order of magnitude as the static-sink scenario in Figure 6-1:a. The objective is to achieve the average delay as close to that of the static sink scenario.

In the non-real-time case, data is first sent to a set of designated gateway (DG) nodes using multi-hop routes, and then collected by the mobile sink when it comes in contact with the DGs as shown in Figure 6-1:b and Figure 6-1:c. The parameter t_{coll} influences the per-packet data collection delay (between generation at a sensor and upload to the MDH) and the actual travel time of the MDH in each collection cycle. The goal is to design sink trajectories and data routes to minimize the average packet delivery delay $t_{coll}/2$ (or $\tau/2$).

<u>Periodic Data Collection (PDC) (</u>$\tau > t_{coll}$):

This corresponds to the generalized scenario for $t_{coll} > \tau$, in which after each *trip collection-time* t_{coll} the sink exits the sensor field and reenters at the beginning of the next *collection cycle.* Since the average *delivery delay* in this case is $\tau/2$, once the parameter τ is chosen (possibly based on the sink availability), reduction of the quantity t_{coll} does not help from a *delivery delay* perspective. However, minimizing t_{coll} by smart trajectory computation and route selection can offer reduced flying time (or *exposure time*) for UAVs and UGVs. For MDH is deployed in a hostile terrain, the collection time t_{coll} also represents the duration for which the mobile sink is exposed. Therefore, the objective is again to minimize t_{coll}. All discussed application categories are marked as I through III in Figure 6-3.

6.3 Data Collection Modes

Data collection can take place in two modes namely *Collect-on-Move* mode or in *Stop-on-Demand* mode.

a. *Collect-On-Move (COM)*: In this mode, the MDH moves at a constant speed throughout its trajectory and collects data from the DGs while it is moving. This mode of collection is suitable when all data buffered at the DGs can be uploaded to the MDH within the contact duration. A number of parameters can influence the suitability of this mode including the data generation rate of the nodes, the hop-bound *k*, the speed of the MDH, the transmission range of the nodes. Sink devices such as fixed winged UAVs [140] [141] that are unable to stop or easily change their speed should collect data in *Collect-on-Move* mode.
b. *Stop-On-Demand (SOD)*: In this mode, the MDH moves at a constant speed along its trajectory and collects data from the DGs as it is moving. However, if it is not able collecting all the data buffered at the DGs, it stops for the duration required to collect the remainder of the data. This mode of collection is suitable when the contact duration is not sufficient to collect buffered data. Sink devices such as rotary winged UAVs [118]or UGVs [96] that are able to hover or stop can collect data in this mode.

6.4 Parameterization of Multi-hop Routing

The static, single-hop and various multi-hop data collection scenarios described at the beginning of this chapter can be parameterized using the unified hop-bound factor *k* which represents the maximum number of hops that a packet is allowed to traverse from a sensor to the MDH. This hop-bound factor essentially determines the extent of multi-hop routing

in the network. As k increases, average hop count incurred by packets increases. At the same time, the number of DGs and the hence the trajectory length decreases. As a result, the amount of data that needs to be uploaded at each DG goes up. When shortest paths are used for routing, the maximum amount of forwarding that can be experienced by a packet is the same as the diameter of the network. Therefore, the maximum value for k is the network diameter D. The minimum value of k that allows static data gathering operation shown in Figure 6-1:a is termed as $k_{critical}$. In other words, $k_{critical}$ represents the optimal placement of the static sink with respect to packet hop count. Static gathering is feasible with multi-hop routing in the range $k_{critical} \leq k \leq D$.

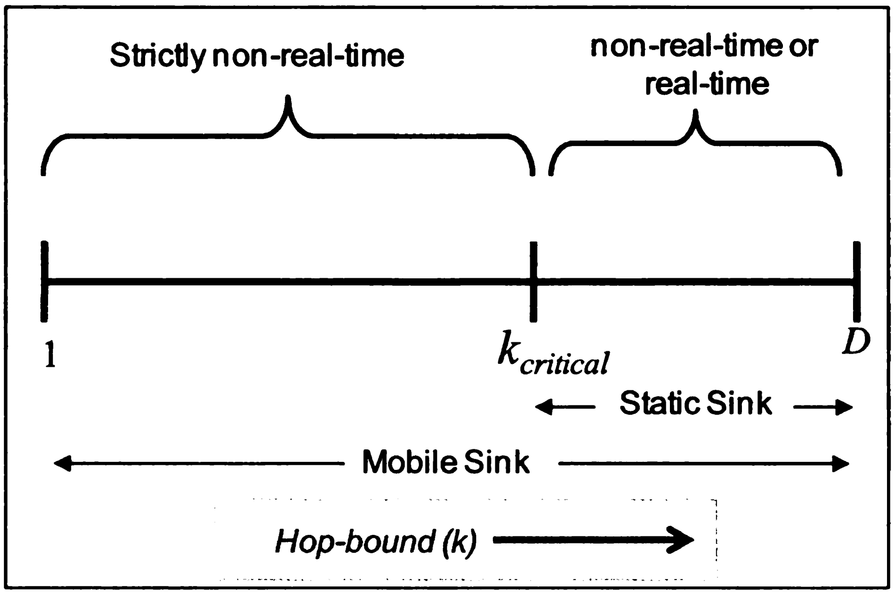


Figure 6-4: Feasible region for k with mobile sinks

As shown in Figure 6-4, with a mobile sink, the extent of multi-hop routing can vary over the entire range of $1 \leq k \leq D$. For the sub-range $1 \leq k < k_{critical}$, the data collection is strictly non-real-time since the MDH is not reachable by all nodes at all points in its trajectory. For the routing range corresponding to $k_{critical} \leq k \leq D$, both real-time and non-real-time collections are feasible by the mobile sink. In this thesis, we specifically focus on the region of k ($1 \leq k < k_{critical}$) in which a mobile sink can only operate in non-real-time mode.

The objective is to develop joint MDH navigation and data routing mechanisms such that sensor data can be collected from a connected sensor network using up to k-hop transmissions, where k is a tunable parameter. In general, the impacts of increasing k are:

1) Decreasing number of Designated Gateways (see Figure 6-1:c and Figure 6-1:d)
2) Increase in multi-hop routing and consequently the energy cost for delivering each packet
3) Increase in data that is collected from each DG which also increases the time needed for collecting data from each DG

6.5 Problem Formulation

For multi-hop routing (see Figure 6-1:d), the hop-bound factor k represents the maximum number of hops that a packet is allowed to travel while being routed from its originating sensor node to the mobile sink. For a given topology, $k = 1$ indicates single hop data collection with mobile sink, and $k = k_{critical}$ indicates the minimum value of k that allows static data gathering operation shown in Figure 6-1:a. The values of k between 1 and $k_{critical}$ (as shown in Figure 6-1:d) can be used to achieve the desired delay or lifetime performance. The MDH trajectory computation with the hop-bound factor k is equivalent to finding a k-constrained *minimum spanning trajectory* (MST). A k-constrained MST is defined such that the trajectory is within up to k hop reach of all network nodes, while providing the minimum possible round-trip distance.

We formulate the k-constrained MST discovery problem as a k-hop dominating set

[11] search or a k-hop cluster formation [18] problem. The dominating nodes or the cluster-heads represent nodes that the MDH has to visit to ensure that data is collected from all network nodes using up to k hop routes. Those dominating or cluster-head nodes define the mobile sink trajectory and are henceforth referred to as *Navigation Agents*. In our formulation, the first hop neighbors of the *Navigation Agents* act as the *Designated Gateways*. Thus, nodes send their data packets to one of the DGs using up to $k-1$ hop routes. One hop is incurred when the data is uploaded from a DG to the MDH, resulting in a maximum of k hops for any packet. In this thesis, we explore and analyze the above k-constrained MST discovery mechanism and characterize the impacts of the tunable hop-bound factor k on collection time t_{coll}.

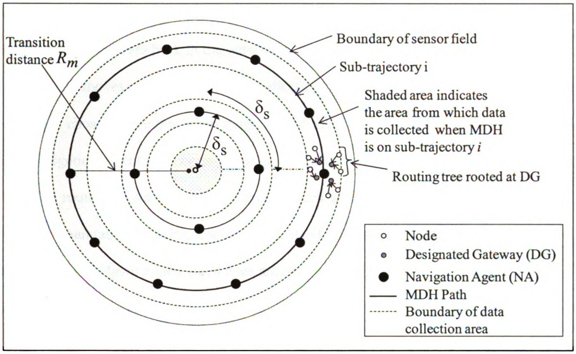
Symbol		**Representation**
Network parameters	n	Total number of nodes in the network
	d	Node density per unit area (per sq. meter)
	R	Radius of the sensor field (meters)
	C	Capacity of the MDH-sensor link (bits per second)
Radio	r	Transmission range of the sensors and the mobile sink (meters)
Traffic	λ	Data generation rate of each node in the network (bits per second)
Mobile Data Harvester (MDH)	v	Speed of the mobile sink (meter/sec)
Policy	k	The hop-bound factor

Table 4: Symbols definition

The assumptions made are as follows. First, we assume that no sensor localization services are available to the sensors and the sink. Second, the mobile sinks move at a constant speed. Third, the link capacity of the sensor-sensor links is sufficiently high so that the aggregated traffic never exceeds the wireless link capacity. The impact of relaxing the capacity assumption is considered in Chapter 10. Hence, only the limited contact time between the DGs and the MDH restricts the amount of data that can be collected.

6.6 Model for the Hop Bound Factor *k*

In this section, we develop a model for analyzing the impacts of the hop-bound *k* on trip collection-time and energy performance. As mentioned in Section 6.5, the MDH has to traverse the network such that all nodes are reachable within *k* hops from the MDH at some point on its trajectory. Such a problem can be formulated as a *k*-hop minimum dominating set problem or *k*-hop clusterhead determination problem [142] on the graph induced by the network topology. Such problems are known to be NP-hard and the determination of the optimal trajectory is beyond the scope of the thesis. Since our focus is specifically on the impacts of multi-hop routing in the data collection process, for the purpose of analysis, we consider a circular network terrain as shown in Figure 6-5 with uniform and dense node distribution. This implies that a MDH trajectory that covers the network area completely while minimizing the travel distance is required. In the literature, survey trajectories for mobile robots have been studied to achieve area coverage for remote sensing. Researchers have found that concentric circular trajectory [143] and spirals [143, 144] provide full area coverage with minimum travel distance. In this chapter, for detailed analysis, we adopt the concentric circular trajectory which is considered one of the best to cover a region. Specific parameters used for the following analysis are defined in Table 4.



Transition distance R_m
Boundary of sensor field
Sub-trajectory i
Shaded area indicates the area from which data is collected when MDH is on sub-trajectory *i*
Routing tree rooted at DG
δ_s
δ_s
○ Node
● Designated Gateway (DG)
● Navigation Agent (NA)
—— MDH Path
------ Boundary of data collection area

Figure 6-5: Sub-trajectories and collection boundaries

In the concentric circular trajectory, the overall MDH trajectory is split into multiple sub-trajectories of different radii as shown in Figure 6-5. The separation between the circular sub-trajectories is δ_S . Each sub-trajectory is defined by the *Navigation Agents* which represent the nodes that have to be visited to achieve area coverage. The distance separation between *Navigation Agents* is also set to δ_S . The parameter δ_S is chosen to achieve the degree of coverage that is required. If the value is chosen too low, coverage is not complete and when chosen too high, the amount of redundant coverage is high. For our analysis we have chosen to set $\delta_s = 2kr$. The separation distance was chosen to ensure that the routing regions associated with DGs around each Navigation Agent are non-overlapping to ensure tractability of the energy analysis presented in Section 6.6.3. The MDH in this case first completes the outer-most sub-trajectory and then moves inward until it reaches the inner-most one. Between two successive sub-trajectories, the MDH transitions using the shortest distance path. After reaching the inner-most sub-trajectory, it moves back to the outer-most one and starts the next collection cycle.

The MDH moves along its trajectory collecting data from Designated Gateways (DG) which are chosen to be the 1-hop neighbors of the *Navigation Agents*. All non-DG and non-Navigation Agent nodes route their data to one of the DGs using up to *k*-hop routes. After the routes are set up based on criteria such as shortest-hop or minimized-energy, each DG maintains a routing tree rooted at itself and spanning across the non-DG and non-*Navigation Agent* nodes. Such trees are depicted in Figure 6-5.

6.6.1 Trajectory Length

For the concentric trajectory as shown in Figure 6-5, the overall trajectory is split into multiple circular sub-trajectories of different radii. The number of such sub-trajectories can be written as R/δ_S where δ_S is the separation between the consecutive sub-trajectories. In our analysis, we use $\delta_S = 2kr$. The number of sub-trajectories can be written as $R/_{2kr}$. The radius of the outermost sub-trajectory is $R - kr$, and adjacent sub-trajectories are mutually separated by a distance of $2kr$. Thus, the radius of the i^{th} sub-trajectory (starting from the periphery) is $R - [kr + (i-1)2kr]$ for $i=1$ to $R/_{2kr}$, and the circumference is $2\pi[R - (kr(2i-1))]$. The total trajectory length L can be written as:

$$L = 2\pi \sum_{i=1}^{R/2kr} [R - (kr(2i-1))] + 2R_m \qquad \text{(Eq. 6-1)}$$

where R_m represents the total transition distance (see Figure 6-5) from the outer-most to the inner-most sub-trajectory. The quantity $2R_m$ represents the cumulative distances traversed across all sub-trajectories during a complete trip.

6.6.2 Trip Collection-Time

For a given link capacity *C,* and MDH speed *v*, with small values of *d*, λ and *k* the contact time for each DG is likely to be sufficient to complete the data upload. However, for relatively large values of *d*, *k* or λ, the MDH may need to stop at all or few DGs. Thus, depending on the specific parameters, the trip collection-time t_{coll} is determined either by the travel time of the MDH, or by the total data upload time - whichever is larger. The travel time of the MDH can be computed as $L/_v$. From Eq. 6-1, the move time to complete all sub-trajectories can be written as:

$$T_m = \frac{2\pi \sum_{i=1}^{R/2kr}\left[R - kr(2i-1)\right] + 2R_m}{v} = \frac{\left(\pi R^2 /_{2kr}\right) + 2R_m}{v} \qquad \text{...... (Eq. 6-2)}$$

Since the DGs are 1-hop neighbors of the *Navigation Agents,* a set of DGs can be seen as a cluster around a *Navigation Agent.* As *k* becomes larger, fewer *Navigation Agents* are required and hence, they get placed farther apart. This also increases the physical distances between the DG clusters. This implies that on certain segments of its trajectory, the MDH may not be in contact with any DG. Therefore, only a fraction of the move time T_m can actually be used for data collection. Let T_p be the productive move time, which is the time during which the MDH is in contact with at least one DG. T_p can be computed by multiplying the number of *Navigation Agents* N_w with the duration for which data can be collected when the MDH is in contact with at least one DG in the vicinity of each *Navigation Agent.* The contact distance with at least one DG in the vicinity of a *Navigation Agent* can range from *2r* (when a *Navigation Agent* and all its DGs are collocated) to *4r* (when at least two DGs of a *Navigation Agent* are located on two diametrically opposite points on the circle of radius *r* around the *Navigation Agent*) leading to an average of *3r*. Therefore, T_p can be expressed as $N_w \times 3r / v$. Given that the number of *Navigation Agents* N_w can be computed as $L/(2k+1)r$, T_p can be rewritten as: $T_p = \dfrac{3L}{(2k+1)v}$ ...... *(Eq. 6-3)*

Let T_u be the amount of time required to upload all the data accumulated at DGs around a *Navigation Agent.* If duration T_p is insufficient to collect data accumulated around a *Navigation Agent,* then the sink stops to collect the remaining data. So, t_{coll} is the sum of the time required to physically complete the trajectory and the stop times. Therefore, t_{coll} (see Figure 6-2) can be written as:

$$t_{coll} = \begin{cases} T_m & if\,(T_u \le T_p) \\ T_m + (T_u - T_p) & otherwise \end{cases} \qquad \text{.......... (Eq. 6-4)}$$

In Eq. 6-4, T_m represents the portion of the trip collection-time for which the sink is moving and $(T_u - T_p)$ represents the stop duration. When the amount of data to be collected from the DGs is small $(T_u \le T_p)$, the stop component is zero. In other words the collection time is same as the sink moving time T_m. This scenario represents the *Collect-on-Move* operating mode. When $(T_u > T_p)$, the stop duration is non-zero, representing the *Stop-on-Demand* collection mode.

representing the *Stop-on-Demand* collection mode.

As pointed out in Section 6.4, the value of the hop-bound factor *k* impacts the amount of data that is collected from the *Designated Gateways* once every collection cycle (τ). As the value of *k* increases, the amount of data to be collected also increases until stops are required to complete the data collection from DGs around a *Navigation Agent*. The lowest value of *k* for which stops are required is referred to as the stop threshold k_{stop}.

6.6.2.1 Numerical Evaluation of Trip Collection-time

Using Eq. 6-4, we obtained the trip collection-time for a circular network terrain of radius 300m and sensor transmission range of 30m. The minimum number of hops (i.e. $k_{critical}$) at which static sink operation is possible is 300/30 = 10. The collection cycle τ was set to 900sec and the MDH speed v was set to 3m/s.

Figure 6-6 and Figure 6-7 show the quantities T_m, T_p, T_u and the resultant trip collection-time t_{coll} with varying *k*, and for different values of λ. With increasing *k*, initially the quantity T_m decreases quite sharply, and then the rate of decrease reduces until *k* reaches a $k_{critical}$, at which point the sink becomes static (Figure 6-1:a). This is because with increasing *k*, the sink trajectory includes fewer and fewer *Navigation Agents* and hence it becomes shorter. The quantity T_p goes down at a much faster rate because as the *Navigation Agents* are spread farther apart, the MDH is not in contact with DG during an increasing duration while in its trajectory. T_u, on the other hand, depends on λ but does not vary with *k*.

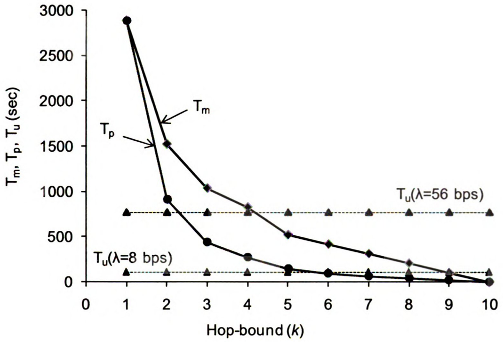


Figure 6-6: Impact on *k* on T_m, T_p and T_u

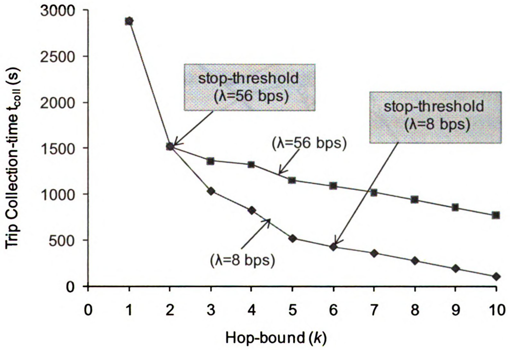


Figure 6-7: Impact of *k* on trip collection-time for different data rates

For all λ, as *k* increases, the collection-time goes down fast until the *stop threshold*

k_{stop} is reached, and then with a slower rate until $k_{critical}$ is reached. t_{coll} is initially dominated by T_m, but as T_p drops below T_u, the condition $t_{coll} > T_m$ as in Eq. 6-4 holds good. At $k = k_{critical}$, t_{coll} is same as T_u. For $\lambda = 8$ bps, the values of $k < 4$ result in trip collection-time greater than τ. For $k < 6$, the MDH can collect data without stopping because $T_p > T_u$ and therefore $k_{stop} = 6$. For $k > 6$, the MDH has to stop and collect data. In contrast, for higher data generation rate $\lambda = 56$ bps, the *stop threshold* $k_{stop} = 2$. Thus, in the higher data generation rate scenario, the stop threshold occurs for a relatively smaller value of *k*.

6.6.3 Energy Consumption

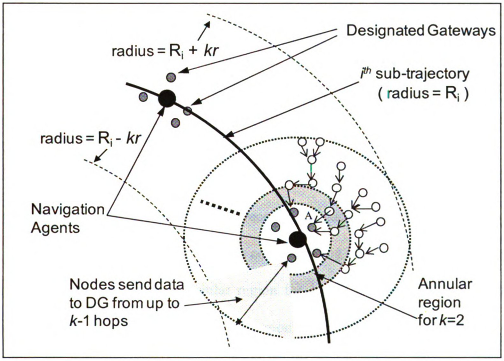


Figure 6-8: Data Aggregation at DGs (1-hop nodes of Navigation Agent)

Network life is closely related to two energy metrics, namely, the energy spent per packet and the standard deviation of energy drainage across all nodes. Note that the energy metrics are not dependent on the specific data collection mode, because energy consumption is only affected by the depth of the DG trees and the number of packets generated in the network. It does not depend on whether the sink had to stop or not for uploading data from the DGs.

In our analysis, the DGs are chosen to be all the 1-hop neighbors of *Navigation Agents*. An alternative approach would be to assign all 1-hop reachable nodes from the MDH to be the DGs along the full trajectory. An advantage of this approach is that it results in higher productive time T_p with $T_p = T_m$ in the best case. We have used the first approach to DG selection for easier analytical tractability of the energy model developed in this section. Regardless, the analysis performed in this section holds for any DG assignment for the reasons outlined in Chapter 9 and Chapter 10.

Figure 6-8 shows a zoomed-in segment of the i^{th} sub-trajectory with radius R_i. Depending on its origin, a data packet has to travel between 1 to *k* hops before it is collected by the MDH from a DG. Energy required to collect a packet that was routed *l*-hops can be written as $E_{tx} + (l-1)(E_{tx} + E_{rx})$, where E_{tx} and E_{rx} indicate the energy expenditures for transmitting and receiving a packet respectively. Around each *Navigation Agent*, there are annular regions corresponding to the number of hops to reach the MDH. An example of such annular region for *k* = *2* is shown in Figure 6-8. The number of packets generated in an annular region defined by radii $(l-1, l)$ is a function of the number of nodes in that region and their data generation rate, and can be written as $\pi\left[(rl)^2 - (r(l-1))^2\right]d \times \lambda$, where *d* is the node density. Considering *l*-hop routing for those packets, the total energy expenditure is:

$$\pi\left[(rl)^2-(r(l-1))^2\right]\times d\times\lambda\times\{E_{tx}+(l-1)(E_{tx}+E_{rx})\} \quad \text{...... (Eq. 6-5)}$$

Average Energy Per Packet (AEPP) within a *Navigation Agent's* jurisdiction (the area from which packets are collected at the DGs of that *Navigation Agent*) can be computed by dividing the total energy expenditure by the total number of generated packets. It can be written as:

$$AEPP=\frac{\sum_{l=1}^{k}\left\{\pi\left[(rl)^2-(r(l-1))^2\right]\times d\times\lambda\times\left[E_{tx}+(l-1)(E_{tx}+E_{rx})\right]\right\}}{\pi(kr)^2\times d\times\lambda}$$

which can be simplified as:

$$AEPP=\frac{\sum_{l=1}^{k}\left\{\pi\left[(rl)^2-(r(l-1))^2\right]\left[lE_{tx}+lE_{rx}-E_{rx}\right]\right\}}{\pi(kr)^2} \quad \text{......... (Eq. 6-6)}$$

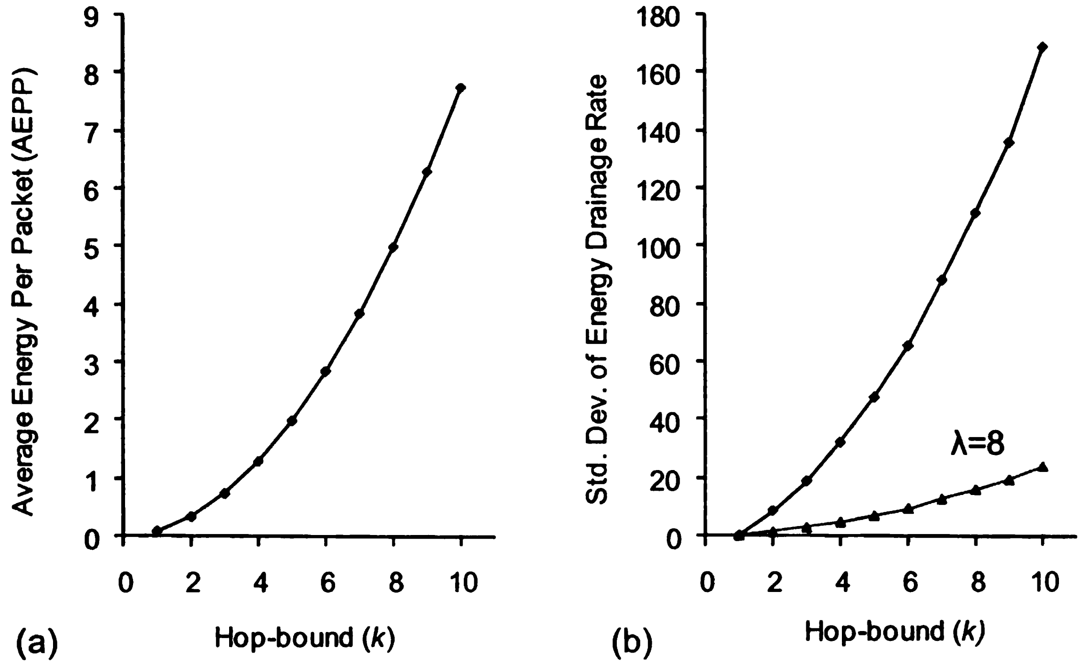


Figure 6-9: Impact of *k* on energy metrics

Assuming spatially uniform node distribution, which results in uniformly distributed *Navigation Agents*, the expression for AEPP in Eq. 6-6 also holds for the entire network. As shown in Figure 6-9:a, the energy required to deliver packets to MDH increases non-linearly with *k*. Eq. 6-6 also indicates that the data generation rate λ and node density *d* do not impact AEPP since the cost incurred by a packet is only decided by *k*, which indicates how far it has to be routed in the worst case.

Assuming that the tree under each DG is balanced, the number of nodes at each level of the tree is evenly distributed across all DGs. The energy drainage at each node is due to a) transmissions of its own originated packets, and b) routing packets from nodes that are farther away from the DGs. The variability in energy expenditure is mainly caused by the routing component. Sensors that are *k-1* hops away from a DG, i.e. *k* hops from the corresponding *Navigation Agent*, incur just the transmission cost for their own packets. Sensors in the annular region *A* defined by radii $(l-1, l)$ have to bear the routing cost of packets from nodes that are in the annular region defined by the radii (l, k). Thus, the routing burden on the nodes in the annular region *A* can be written as:

$$E_{routing} = \pi\left((kr)^2 - (lr)^2\right) \times d \times \lambda \times \left(E_{tx} + (l-1)(E_{tx} + E_{rx})\right)$$

Assuming that the routing is load-balanced and all nodes in region *A* experience equal routing burden, the routing energy drainage in region *A* can be written as:

$$E_{l,routing} = \frac{\pi\left((kr)^2 - (lr)^2\right) \times d \times \lambda \times \left(E_{tx} + (l-1)(E_{tx} + E_{rx})\right)}{\pi\left[(lr)^2 - ((l-1)r)^2\right] \times d}$$

which can be simplified as:

$$E_{l,routing} = \frac{\lambda(k^2 - l^2)(E_{tx} + (l-1)(E_{tx} + E_{rx}))}{(2l-1)} \quad \text{.... (Eq. 6-7)}$$

Eq. 6-7 implies that routing energy drain for any node depends only on its position in the tree rooted at a DG, the value of $k$which defines the depth of the tree, and the data generation rate λ. It does not depend on node density because with increasing density, the number of DGs increases proportionately, resulting in more DG trees, but the depth of the tree does not change. With increasing λ, the average routing load experienced by nodes increases. The mean routing energy drainage for a node is given by: $\hat{E}_{routing} = \frac{1}{k} \times \sum_{l=1}^{k} E_{l,routing}$. For a DG, the Standard Deviation of Energy Drainage (SDED) can be computed as:

$$SDED = \sqrt{1/k \times \sum_{l=1}^{k} \left[E_{l,routing} - \hat{E}_{routing}\right]^2}$$

which is plotted in Figure 6-9:b. Observe that the quantity SDED increases non-linearly with k. This increase pattern can be explained by the non-linear increase in multi-hop routing (i.e. a DG's tree size) with increasing k. The quantity t_{coll}, however, stops decreasing substantially once the multi-hop routing indicator k reaches the thresholdk_{stop}. This is apparent from Figure 6-7 in Section 6.6.2.1. These suggest that the high energy cost for large k is not quite offset by the reduction in the trip collection time. Therefore, values of k below k_{stop} should serve as a reasonable range from which k can be chosen for striking a reasonable balance between the collection delay and network life.

6.6.3.1 Impacts of Data Collection Modes on Energy

The above energy analysis has been performed for stable data collection when the MDH is operating within the feasible region of k as defined by the stop and the stability thresholds. Hence, the *Collect-on-Move* MDH is assumed to operate between $k_{stability}$ and k_{stop} and a *Stop-on-Demand* MDH operates beyond $k_{stability}$. Under such stable operations, all the data accumulated at the DGs during a collection cycle is collected by the sink during that cycle. As a result, the energy metrics developed in the previous sections hold regardless of the specific data collection modes.

6.7 Applicability of Model to Non-circular Covering Trajectories

Area coverage and minimization of the MDH travel distance are two primary factors that drive the choice of trajectory pattern. Concentric circular and spiral trajectories have been found to be best for achieving high coverage while minimizing the travel distance. However, there are a number of other practical considerations that can influence the choice such as minimizing the number of turns, energy required for special maneuvers and the like. Such considerations may dictate the choice of a different covering trajectory for specific mobile sink deployments. Thus, if a different covering trajectory is chosen, the analysis performed in this section still applies with minor modifications.

Figure 6-10 shows some example trajectories that can be used to achieve area coverage and δ_S represents the separation distance between sub-trajectories. The top graph (Figure 6-10:a) shows the concentric circular sink trajectory which consists of multiple circular sub-trajectories of different radii. In the middle graph (Figure 6-10:b), the sink trajectory traverses the network from side to side. In the bottom graph (Figure 6-10:c), the sink traverses the terrain following a spiral trajectory. If a different sink trajectory is used, Eq.

6-1 will need to be modified appropriately and the resulting trajectory length, L (reported in Appendix B) should be used for the rest of the derivations in Section 6.6.2.

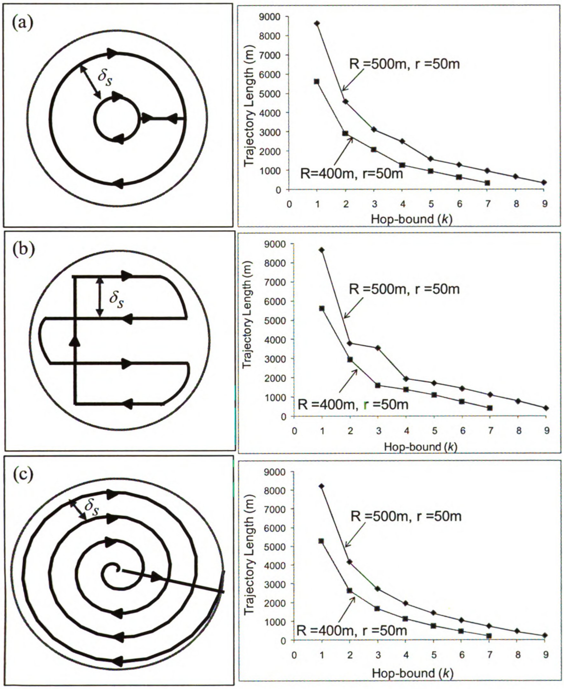


Figure 6-10: Impact of k on trajectory length, L

Figure 6-10 shows the length of the sink trajectories for different values of k for the chosen heuristic methods for different ratio of topology radius R and transmission range r and $\delta_s = 2kr$. In all cases, results show that as the extent of multi-hop routing increases, the trajectory length shrinks. This trend was found to be valid for trajectories other than those shown in Figure 6-10 and other values of δ_s .

6.8 Summary and Conclusions

In this chapter, a generalized formulation of the data collection problem has been developed that encompasses both static and mobile sink scenarios. The formulation uses a tunable hop-bound factor (k) which represents the amount of allowed multi-hop routing. The ranges of k appropriate for real-time and non-real-time data collection using static and mobile sinks were analyzed. A model was developed based on the proposed generalized framework and expressions were derived for trip collection-time and energy metrics. In addition, the threshold of k referred to as k_{stop} which would require the mobile sink to switch from *Collect-on-Move* to *Stop-on-Demand* mode was identified.

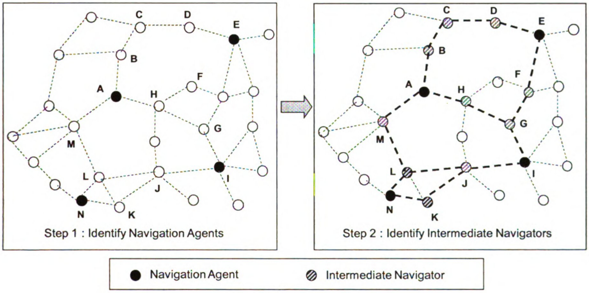
The generalized formulation for data collection proposed in this chapter can be used to assess relative merits of static and mobile sink data collection options for given application, network and MDH parameters. When mobile sink option is considered, the degree of multi-hop routing that can be sustained without incorporating stops in the MDH trajectory can be determined. This is an important consideration for sinks such as fixed-wing UAVs that cannot hover or stop as mentioned earlier. Numerical evaluation of the trip collection-time and energy metrics shows that as the hop-bound factor increases, the trip collection-time decreases at the expense of energy performance. This confirms that the hop-bound factor k can be used as a knob to achieve the desired energy-delay balance in the context of mobile sink data collection.

Chapter 7. Network-Assisted Data Collection (NADC)

In Chapter 6, we presented a generalized formulation for the data collection problem using a hop-bound factor *k* which can used to achieve the desired energy-delay balance. In this chapter, we develop a distributed network-assisted solution architecture for non-real-time data gathering to realize the generalized *k*-hop based data collection framework shown in Figure 6-1:d.

7.1 Concept Summary

Instead of sensors' geographical location information, the proposed navigation mechanism uses network connectivity information for MDH navigation. The key components of the navigation and data collection mechanism are as follows.



A
B
C
D
E
F
G
H
I
J
K
L
M
N
Step 1 : Identify Navigation Agents
Step 2 : Identify Intermediate Navigators
● Navigation Agent
▨ Intermediate Navigator

Figure 7-1: Logical overlay topology construction for *k*=2

Phase 1: A logical overlay graph is created (see Figure 7-1) which is composed of two kinds of nodes – Navigation Agents (NA) and Intermediate Navigators (IN). The MDH is navigated through the NA nodes, and the IN nodes act as the intermediate MDH routing entities between two NA nodes. The 1-hop neighbors of the NAs act as the Designated Gateways by accumulating and buffering data from the other nodes and uploading it to the MDH. The construction of the overlay graph is accomplished in following two steps.

Step 1: Navigation Agents are selected such that by passing through them, the MDH is guaranteed to traverse a trajectory that ensures that all nodes can be reached within a maximum of k hops. Navigation Agents ensure that data is collected from all the nodes in their vicinity and guide the navigation of the MDH. Thus, a spanning trajectory is accomplished by the MDH without having to pass through the close proximity of each individual sensor node. This is the key to reducing the *trip collection-time* t_{coll} in the proposed navigation mechanism. Figure 7-1 shows a network of sensors where Nodes A, E, I and N are identified by the network as the NA nodes. The value of k is set to 2 in this example. Note that by visiting the identified NA nodes, it is possible to collect data from all nodes in the network using routes with 2 or less hops.

Step 2: A set of *Intermediate Navigator (IN)* nodes are selected for guiding the MDH between the Navigation Agents. This is needed since the NA nodes may not always be radio neighbors to each other. The IN nodes act as the intermediate MDH routing entities between two NA nodes. Nodes that are on the shortest path between two NA nodes are identified as the IN nodes. In the network shown in Figure 7-1, nodes B, C, D, F, G, H, J, K, and M are identified as the IN nodes.

Phase 2: A distributed Traveling Salesman Problem (TSP) mechanism is used to compute a path that traverses through all the NA nodes. The TSP solution is computed on a graph consisting of NAs as the vertices, and link costs are defined as the hop counts of the shortest paths between the NA node pairs. Two NA nodes are considered connected only if they are at most $2k+1$ hops apart. Figure 7-2 shows a TSP path that is computed for the example network.

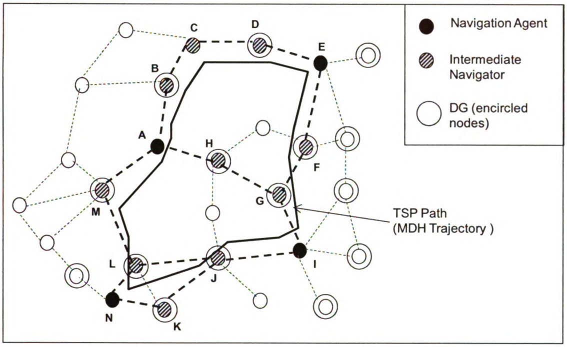


Figure 7-2: Network computed TSP path covering NA nodes

Phase 3: A runtime navigation system is designed for physically routing the MDH along an appropriate sequence of the NA and IN nodes along the computed TSP path so that the network wide collection-time for the MDH is minimized. This component of the MDH navigation process is similar to network layer routing except that in this case the MDH is physically routed through the NAs and the INs as opposed to the packet routing through multiple routers. The NAs and INs on the network-computed path guide the mobile data harvester on a trajectory for data collection through the network as indicated in Figure 7-2.

Phase 4: The first-hop neighbors of the chosen NAs declare themselves as *Designated Gateways*. When $k=1$, all data is uploaded to the MDH using single hop transmissions. But, when $k > 1$, nodes that do not come in direct contact with a Navigation Agent need to route their data to an appropriately chosen Designated Gateway using multi-hop paths so that the data may be buffered and uploaded to the MDH. Therefore, routing paths need to be chosen from each sensor node to an appropriately selected Designated Gateway.

7.2 NADC: Algorithmic Details

7.2.1 Construction of Overlay Navigation Topology

The choice of Navigation Agents and the Intermediate Navigators on the overlay logical topology will have significant impact on the MDH's round trip tour time t_{coll}, and the subsequent *delivery delay*. The problem of finding the smallest set of NAs by visiting which the MDH would be guaranteed to have passed through *k*-hop topological distance of all network nodes can be formulated as the ***minimum k-hop dominating set finding problem*** [145] in a connected graph. Since finding the minimum dominating set is known to be NP-Hard, we have adopted a heuristics based approach based on [146] as follows.

Tree Creation: A network-wide tree is created using an arbitrarily chosen root node. At the end of the tree creation process, every node knows its distance to the root and also those of its neighbors.

Navigation Agent Identification: This algorithm identifies the navigation agents based on the logical overlay topology construction as outlined in Section 7.2.1. After a node determines its distances from the root, it declares itself as a 'Navigation Agent' by sending a '*Declare-NA*' message with TTL (Time to Live) value set to $k-1$. All nodes that receive this declaration message accept the Navigation Agent as their parent NA, decrement the value of the TTL and re-broadcast the '*Declare-NA*' message. If a node finds the received TTL value to be zero, the receiving node sends out an '*Accept-NA*' message, otherwise, they decrement TTL and re-broadcast the message. A node that receives an '*Accept-NA*' message with TTL=0 from all of its neighbors that are closer to the root (have lower hop-count to the root), declares itself as a 'Navigation Agent'. This process continues until all nodes are marked either as a Navigation Agent (NA) or as Covered. The actions taken by nodes when they receive the '*Declare-NA*' and '*Accept-NA*' messages are summarized in the pseudo code in Figure 7-3.

```
Algorithm for Navigation Agent Identification:

Initializations for NA identification process:
Set number of Accept-NA messages received to zero

Process Declare-NA message from a neighbor:
Mark neighbor as a Navigation Agent in neighbor table
if ( ttl value in Declare-NA message == 0)
  Broadcast an Accept-NA message to all neighbors with TTL = $k$-1
else {
   Decrement TTL value in Declare-NA message
   Re-broadcast Declare-NA message
}

Process Accept-NA message from a neighbor:
lower_neighbor_count = number of neighbors with lower hop count to root node
if ( TTL value in Accept-NA message == 0) {

   if ( number of Accept-NA messages received == lower_neighbor_count ) {
     //none of my neighbors is an NA, so declare myself as one
     Mark myself as a Navigation Agent
     Broadcast a Declare-NA message to all neighbors with TTL = $k$-1
   } else {
     // keep track of how many Accept-NA messages were received
     Increment Accept-NA messages received
   }
   // ttl value is greater than 0
} else {
   Decrement TTL value in Accept-NA message
   Re-broadcast Declare-NA message
}
```

Figure 7-3: Pseudo code for Navigation Agent identification

These self-elected NA nodes represent a minimal dominating set so that the *k* hop neighbors of all the NA nodes include the entire node population in the network. This implies that if the MDH trajectory visits all the NA nodes, then it is guaranteed to be passing through the network such that it is within *k*-hop distance of all nodes at least once during its trajectory. Therefore, in order to collect all network data using up to *k*-hop routing, it is sufficient for the MDH to traverse through the virtual topology formed by the NA nodes that are self elected using the distributed process as outlined above.

By definition, the Navigator Agent nodes are at least $k+1$ and up to $2k+1$ physical hops away from each other. As a result, while performing the MDH navigation, the data harvester needs to be physically routed through $k+1$ to $2k+1$ Intermediate Navigator (IN) nodes. For example nodes *B, C* and *D* in Figure 7-1 act as the IN nodes while physically routing the MDH between the NA nodes *A* and *E*. Similarly, nodes *F* and *G* act as the IN nodes while routing the MDH between the NA nodes *E* and *I*. To enable the MDH navigation, a set of such IN nodes need to be selected for completing a round trip trajectory across the network.

Intermediate Navigator Identification: The IN nodes ($k+1$ to $2k+1$) between two NAs can be identified by finding the nodes on the shortest hop physical routes between the NA nodes. To identify such INs, each NA constructs a minimum hop tree rooted at itself and spanning up to a depth of $2k+1$ physical hops. A standard beacon based tree creation process is used for this step. Once created, such trees represent the minimum hop routes between every NA neighbor pairs on the logical topology. Nodes on the route between such NA neighbor pairs are then identified as the Intermediate Navigators. The NA and IN node identification completes the distributed construction of the overlay navigation topology, as demonstrated in Figure 7-1.

7.2.2 Distributed Trajectory Computation

The goal of this algorithmic module is to compute an MDH trajectory that minimizes the traveled distance while passing through all the NA nodes. The trajectory computation can be formulated as a Traveling Salesman Problem (TSP) on the overlay topology consisting of the NA nodes. The objective is to obtain a minimum cost tour of the NA nodes such that each NA is visited exactly once before returning to the starting point. Cost of the links in the overlay topology represents the shortest distance between the NAs. Distance is measured in terms of hop count as an estimate of the physical distance.

An *Ant Colony Optimization-TSP* solution as proposed in [147] has been adapted for this step in the following manner. The underlying idea is to use exploratory ant packets to search for different round trip solutions to the TSP and then to use a positive feedback mechanism for reinforcing good quality solutions represented by the trajectories of exploratory ant packets that successfully complete the full round-trip with desirable trip costs. Ant packets are originated at each NA and they build solutions to the tour by moving from an NA to another until all NAs are visited. Note that ants are routed between NAs using multi-hop paths consisting of the Intermediate Navigators. Ants that prematurely return at its starting NA without visiting all NA nodes are discarded. After the completion of a complete tour during the *search* round, an ant makes another *reinforcement* round along its already traversed path and lays a certain quantity of pheromone on each edge of the path depending on the total cost of the tour. This process serves as the positive reinforcement of a path found during the *search* round based on its quality. During the search phase, when an ant arrives at an NA node, it selects the next NA node to visit based on the pheromone that has already been deposited on the outgoing edges from the current node. To summarize, at any point in time the pheromone trail on an edge represents the knowledge that has been accumulated about the desirability of the edge in the TSP solution.

7.2.3 Run-time Navigation (RN)

For navigation purposes, the framework relies on an accurate Direction of Arrival (DoA) [148] detection capability using phased array antenna systems at the MDH.

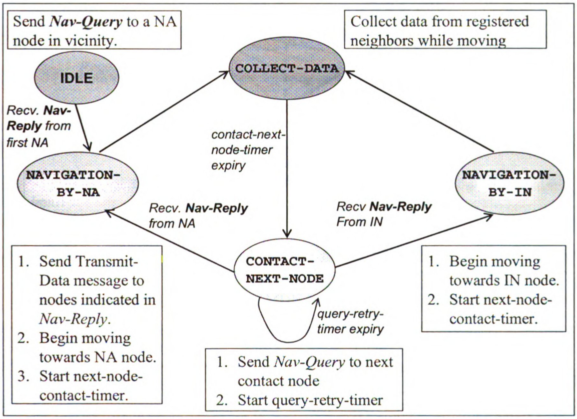

Send *Nav-Query* to a NA node in vicinity.
Collect data from registered neighbors while moving
IDLE
COLLECT-DATA
Recv. **Nav-Reply** from first NA
contact-next-node-timer expiry
NAVIGATION-BY-NA
NAVIGATION-BY-IN
Recv. **Nav-Reply** from NA
Recv **Nav-Reply** From IN
CONTACT-NEXT-NODE
query-retry-timer expiry
1. Send Transmit-Data message to nodes indicated in *Nav-Reply*. 2. Begin moving towards NA node. 3. Start next-node-contact-timer.
1. Begin moving towards IN node. 2. Start next-node-contact-timer.
1. Send *Nav-Query* to next contact node 2. Start query-retry-timer

Figure 7-4: State machine for navigation and data collection

The MDH starts its navigation from an arbitrary location in the network. Initially, it collects neighborhood information from periodic hello messages broadcast by the sensor nodes. Once it receives a hello from an NA in the vicinity, the MDH sends a '*Nav-Query*' message, as a response to which the NA sends a '*Nav-Reply*' back to the MDH. Upon receiving the *Nav-Reply* message, the MDH enters the NAVIGATION-BY-NA state as shown in Figure 7-4.

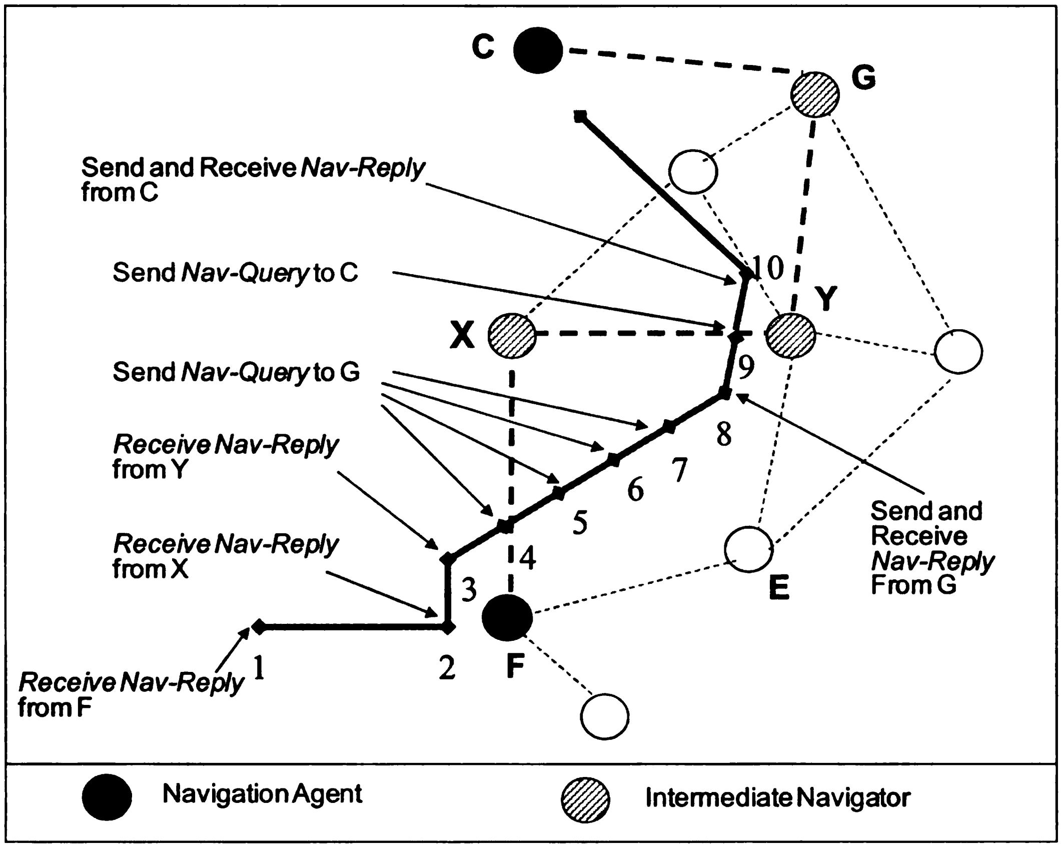

C
G
Send and Receive *Nav-Reply* from C
Send *Nav-Query* to C
X
Y
Send *Nav-Query* to G
Receive Nav-Reply from Y
Receive Nav-Reply from X
Receive Nav-Reply from F
Send and Receive *Nav-Reply* From G
E
F
1
2
3
4
5
6
7
8
9
10
Navigation Agent
Intermediate Navigator

Figure 7-5: Navigation of MDH from NA node F to C (*k*=2)

The *Nav-Reply* message contains the following information:

1. Trajectory related: The next NA on the trajectory that the MDH should visit, and the next IN node that will be able to route the MDH to that next NA node.

2. Data collection related: A list of all the NA's 1-hop neighbors from which the MDH needs to collect data when it is present in the vicinity of this particular NA.

3. Direction related: While receiving the *Nav-Reply* message, the antenna system of the mobile data harvester evaluates the signal's Direction of Arrival (DOA) [148] so that the MDH knows which direction to move to get closer to the NA. In the absence of any location information, this DOA based mechanism provides an approximate yet practical way for navigation in the present framework.

The MDH determines the DOA of the *Nav-Reply* message, and based on the direction, starts moving towards the NA. It starts a contact-next-node-timer which expires after the MDH has traveled for a certain distance. The MDH then starts data collection from the nodes registered with the NA by sending a *Transmit-Data* message. The *Transmit-Data* message is broadcast periodically and enables data transmission from individual nodes in the neighbor-list in the *Nav-Reply* message. This mechanism ensures that only the nodes in the list transmit data to the MDH until the MDH makes contact with the next NA. After completing these actions, the MDH transitions to the COLLECT-DATA state. Figure 7-5 illustrates the sequence of events during the navigation of the MDH between two NA nodes *F* and *C* using three IN nodes *X*, *Y* and *G*. The sequence starts with the reception of the *Nav-Reply* from node *F*. The MDH enters NAVIGATION-BY-NA state and performs related actions as described above.

In the COLLECT-DATA state, the MDH collects sensed data from the nodes. Once the contact-next-node-timer expires (point 2 in Figure 7-5), the MDH moves to CONTACT-NEXT_NODE state and sends a *Nav-Query* to establish contact with the IN node indicated in the Nav-Reply message from the NA. It repeats the *Nav-Query* message until it receives a reply from the next contact node. In the example, the MDH sends the

Nav-Query to IN node *X*. Once it receives a reply message from the IN node, the MDH moves into the NAVIGATION-BY-IN state.

```
Network-assisted Runtime Navigation and Data Collection:

Process Nav-reply from current_host_node:
//initializations based on Nav-reply
next_host_node = Nav-reply_msg(next_node_to_visit)

If (current_host_node is Navigation-Agent) {
    // Nav-reply contains list of neighbors
    Neighbor_List = Nav-reply_msg (neighbors (current_host_node))
    //Collect data from all of its neighbors
    For (each neighbor in Neighbor_List) {
        Set Tx-Enabled bit in Transmit-Data message
    }
    Set Tx-Enabled bit for current_host_node
    Send Transmit-Data message
    Move towards new_host_node(Nav-reply_msg)
    Send direction_query to new_host_node
    } else {  // current_host_node is Intermediate Navigator
S1:   Move towards current_host_node(Nav-reply_msg)
      Send Nav-Query to new_host_node
      If (Nav-reply_msg not received)
          go to S1
      }

 Move towards a host_node (Nav-reply_msg):
  Find angle of arrival (DOA) of Nav-reply_msg
  Set destination to x,y in the direction of DOA at distance = delta
  Physically move to x,y

```

Figure 7-6: Collect-on-Move algorithm description

Actions in the NAVIGATION-BY-IN state are similar to NAVIGATION-BY-NA. Again, the MDH determines the DOA and starts moving towards the IN node. It starts a contact-next-node-timer and enters the COLLECT-DATA state and continues to collect data. Upon timer expiry, it contacts the next-contact-node indicated in the *Nav-Reply* from the IN node. The reply message from the IN node could have contained another IN node or a NA node as the next contact node on the TSP path. This is because any two NA nodes can be either k+1 or 2k+1 hops away from each other. In the example in Figure 7-5, the Nav-Reply from X contains IN node Y as the next contact node. So, in this case, the MDH enters the NAVIGATION-BY-IN once more before going to NAVIGATION-BY-NA. If the next contact node is an NA node, then it enters the NAVIGATION-BY-NA and repeats the actions described earlier. Thus, data collection from the nodes registered with NA takes place from the time the MDH receives a *Nav-Reply* from the NA to the time it receives a *Nav-Reply* from the next NA. In Figure 7-5, the registered neighbors of F are allowed to transmit data to the MDH between points 1 and 10.

7.2.4 Designated Gateway (DG) Identification and Routing

One hop neighbors of Navigation Agents act as the DGs. After an NA is identified according to the algorithm described in Section 7.2.1, all its one-hop neighbors declare themselves as DGs. When the MDH visits the Navigation Agents, it is able to directly collect data from the Designated Gateways using a single hop transmission. The NA identification process also ensures that every sensor node has at least one DG within $k-1$ hops and at least one NA within k hops.

Sensor nodes that do not directly come in contact with the MDH route their data to the closest DG. The route from any node to the NA is found using the NA-rooted trees during the Intermediate Navigation identification process mentioned in Section 7.2.1. Along the route to an NA, a packet from a sensor node gets to a DG along the path where it is buffered. The idea behind buffering data at the Designated Gateways rather than at the Navigation Agents is two-fold. First, it saves energy by eliminating one extra transmission for every data packet. Second, it helps to prevent buffer overflow that can occur if all DGs route packets to their NA for buffering until the MDH comes into range.

The choice of the Designated Gateway to which a node routes its data can be based on various criteria. For instance, the hop-count to DG, the residual energy of the DGs, the quality of the link between the DG and the MDH, etc. In this work, we have chosen to have nodes send their data to the closest Designated Gateway node.

7.3 Summary

In this chapter, we developed the distributed protocol and algorithms required to perform network assisted data collection called Network-Assisted Data Collection (NADC) based on the generalized framework proposed in Chapter 7. NADC does not rely on the availability of localization services for the sensors and the MDH. The various components of NADC and the algorithmic details were presented.

Chapter 8. Network Assisted Data Collection: Performance Evaluation

In Chapter 7, a distributed network assisted data collection framework for mobile data harvesters was developed and the required algorithms were described in detail. The performance of the proposed NADC framework has been evaluated through an NS2 simulation implementation. The details of the experiments and the results obtained are presented in this chapter.

8.1 Simulation Setup

The simulation model represents sensor networks deployed for monitoring applications that generate data at constant rate. The sink represents mobile devices such as Unmanned Ground Vehicle (UGV) or mobile robot. Both sensors and the MDH in our experiments are assigned a communication range of 30m. Each node generates data at a constant rate of one 128 byte packet per second, and makes transmissions using an IEEE 802.11[149] MAC protocol. The speed of the mobile data harvester is set to 3m/s which is the typical speed of a mobile sensor system [83]. Each experiment was run for duration of 1200s to compute the collection statistics.

8.1.1 Implemented Protocols

The following mechanisms have been implemented.

Network Assisted Data Collection (NADC): The following flavors of the proposed NADC framework based on the mechanisms presented in Chapter 7 have been implemented.

1. Centralized (CENT): Centralized versions of the navigation overlay construction and the TSP-based MDH trajectory computation as presented in Sections 7.2.1 and 7.2.2. This scheme is expected to provide a performance upper bound for the fully distributed

NADC.

2. Distributed with Minimum Dominating Set Overlay (*D-MDO*): Corresponds to the distributed solution for minimum dominating set based navigation overlay computation, ANT TSP based MDH trajectory determination, and runtime navigation as presented in Sections 7.2.1, 7.2.2 and 7.2.3. This is the completely distributed version of NADC.

3. Distributed on Physical Network Topology (D-PNT): In this naïve approach, no navigation overlay is computed. As a result, all network nodes act as navigation agents. Therefore the MDH has to pass through each individual node as opposed to going only through the overlay nodes used in the other strategies. The performance of this mechanism is expected to provide the lower bound for NADC.

Random Walk (RW): A Random-Walk-based mobility strategy from [106] has been implemented. The key idea is that after every pause, the MDH randomly picks the direction and travels for a pre-set distance equal to the communication range. While traveling along the chosen direction, the MDH collects data from the nodes that come in contact using *k*-hop paths.

Area-Scan: Given the dimensions of the sensor field, a centralized entity computes a trajectory to scan the entire area. The trajectory depends on the dimensions of the sensor field, the sensor communication range and the value of *k*. This scheme does not require the sensors to have localization information.

In the case of *Area-Scan* and *RW*, as the MDH moves, it sends out *k*-hop beacons in response to which all nodes within *k*-hop distance forward the data stored in their buffers to the sink. So, unlike NADC, data is not buffered at the DGs prior to the arrival of the MDH.

8.2 Single Hop Data Collection (*k*=1)

In a 150m x 150m sensor field, 60 nodes were placed such that they form a connected network in various configurations namely three weak clusters forming a triangle (*3-Tri*), three clusters placed in a line along the diagonal (*3-Line*), nodes distributed around voids (*Void*), uniformly distributed (*Uni*) and placed in a lattice (*Latt*) as shown in Figure 8-1. Each node has been configured with a data buffer of 200 packets.

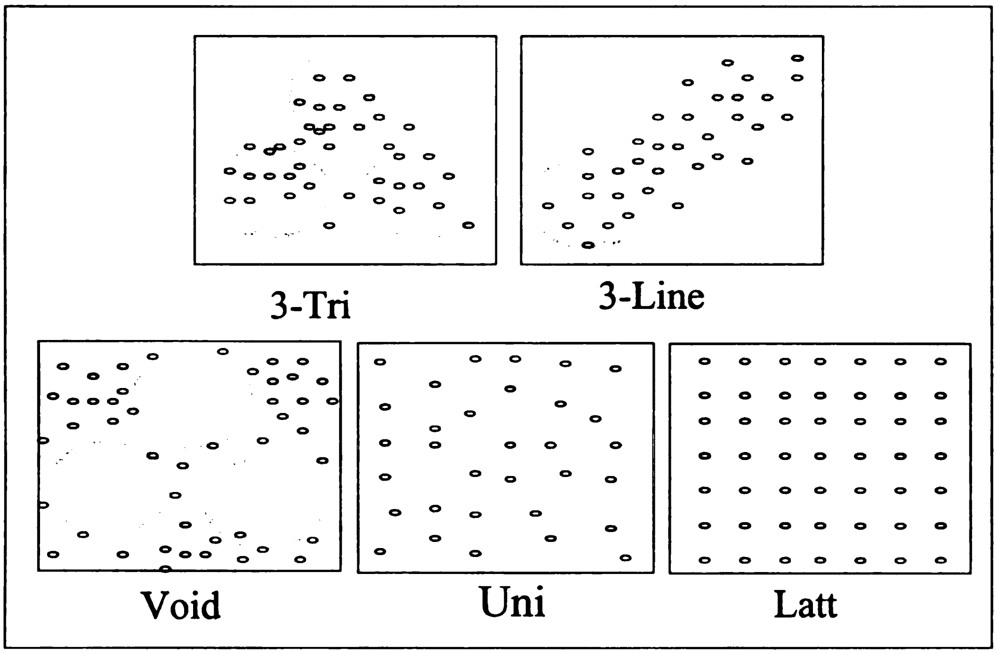


Figure 8-1: 60 node topologies used for experiments

The average *delivery delay* for the evaluated protocols is shown in Figure 8-2. We use *D-PNT* as the baseline mechanism to compare with the *delivery delay* of the other protocols. The results indicate that choosing a subset of nodes to visit for the purpose of data collection brings down the *delivery delay* significantly for all of the different topologies considered. For instance, for *3-Tri* topology, the *CENT* solution reduces the delay by 59% compared to the *D-PNT* topology tour. In *CENT*, the identification of the Navigation Agents and TSP solution on the corresponding overlay are computed centrally with global topology information. As a result, it performs the best thus providing an upper bound. For the same *3-Tri* topology, the *D-MDO* solution, which is a fully distributed counterpart of *CENT*, achieves 54% delay improvement. Observe that the delay performance achieved by *D-MDO* is better than *D-PNT* and very comparable to *CENT* for all the clustered topologies. It was found that the *delivery delay* is directly related to the trip collection-time and hence the round trip distance of the MDH trajectory.

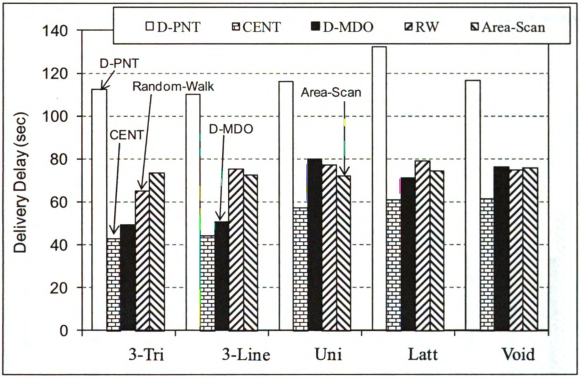


Figure 8-2: Delay performance for various topologies

Figure 8-3 shows the *trip collection-time* and round trip distance for all protocols except *RW*, since these metrics do not apply to random-walk-based mechanisms which are not guaranteed to complete round trips. *Area-Scan* scheme shows a steady *delivery delay* because the trajectory followed by the sink is the same for all the topologies since the dimension of the sensor fields is the same in all cases. It can be observed that the trends for the *delivery delay* (except *Area-Scan*) follow the collection-time and distance very closely. This is because data is collected from a node periodically when it visits the node's parent NA approximately every *trip collection-time* or t_{coll}. Upon further investigation, we found that the collection-time is actually related to the number of Navigation Agents identified during the overlay construction phase. For example, the number of NAs identified for *3-Tri* topology by *CENT* and *D-MDO* were 7 and 10 respectively. In the case of 60 node *Uni* topology, *CENT* and *D-MDO* identified 9 and 16 NAs respectively. Higher number of NAs gives rise to longer trips and subsequently higher packet *delivery delay*.

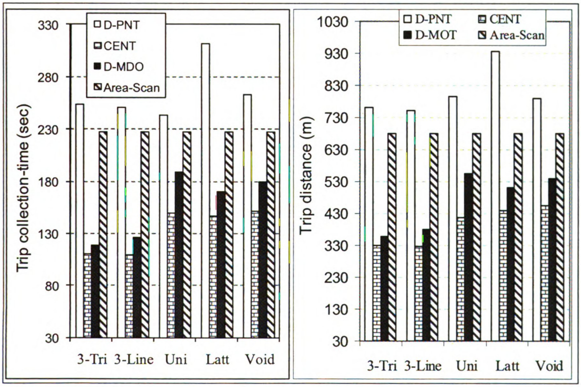
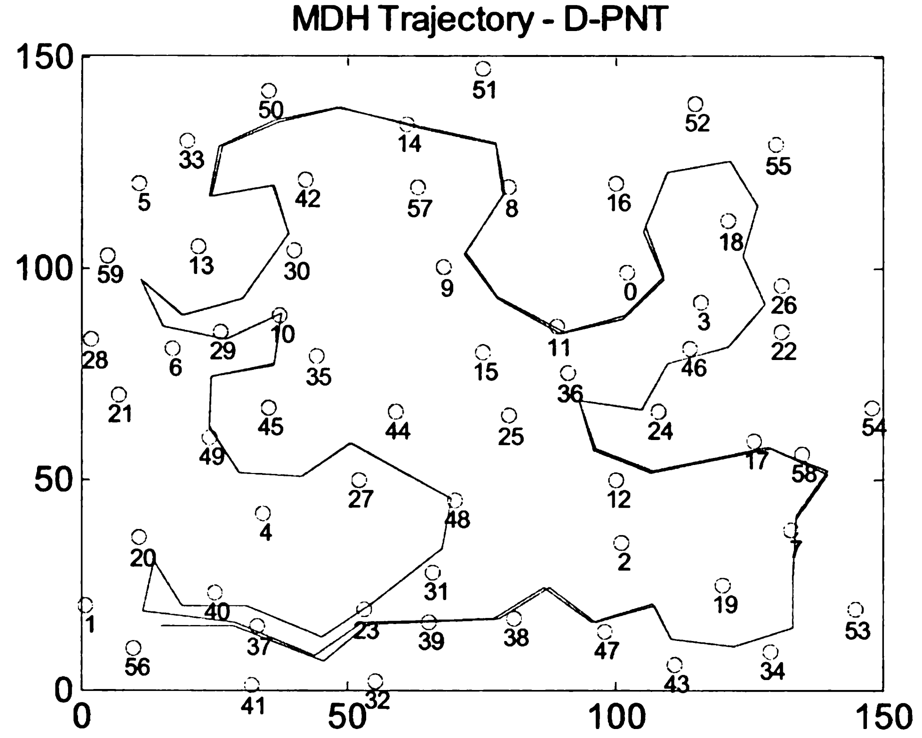


Figure 8-3: Trajectory indices – collection-time and distance



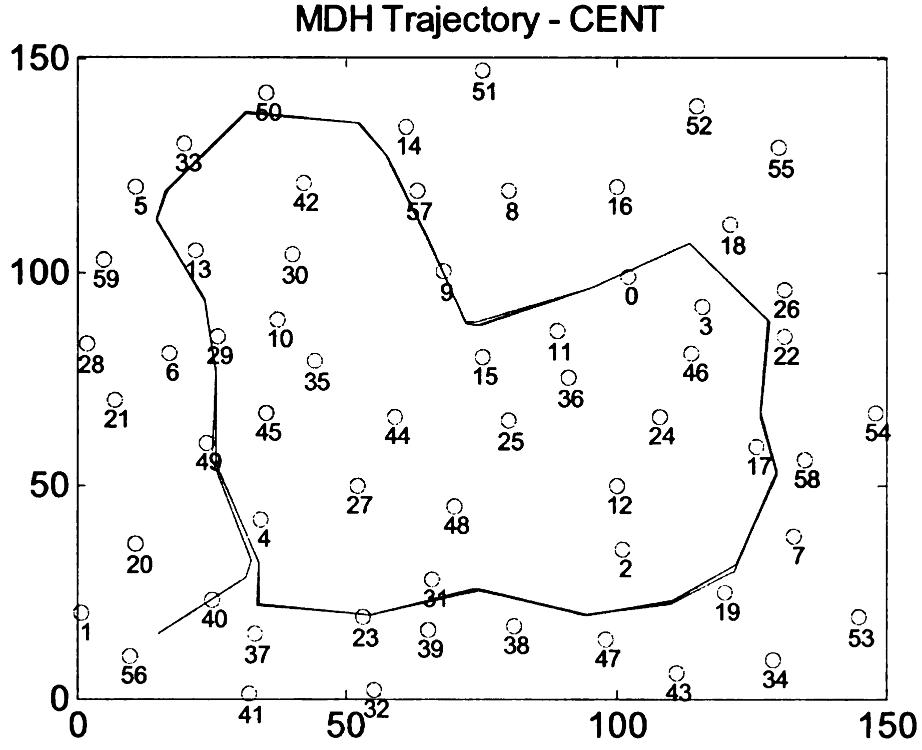
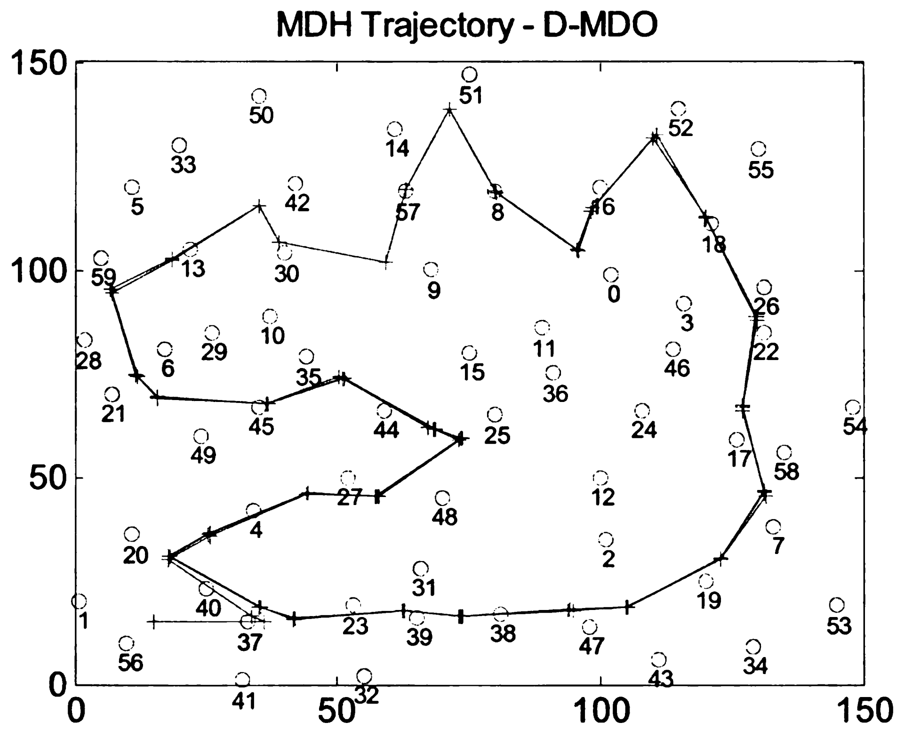
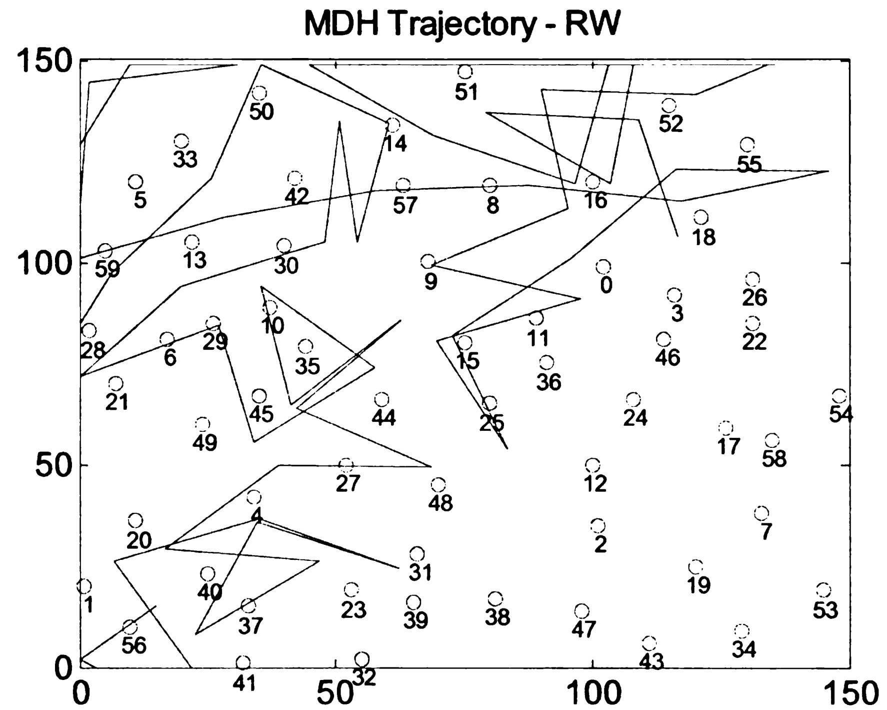


Figure 8-4: MDH trajectories for 60 node uniform topology

Figure 8-4: MDH trajectories for 60 node uniform topology (contd.)





The actual trajectories along which the Run-time Navigation System guides the MDH for collecting data from 60 node *Uni* network are shown in Figure 8-4. This figure bears out the *collection-time* and round trip distance information shown in Figure 8-3 in a visual form. It can be clearly seen that for *D-MDO* and *CENT* schemes, the trajectory is shorter than *D-PNT*. For *RW*, the trajectory is completely independent of the node positions and parts of the network are not visited at all.

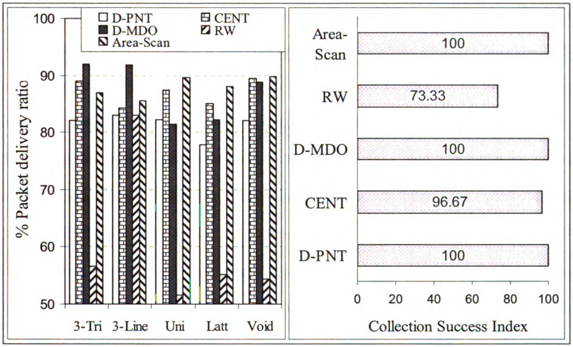


Figure 8-5: Packet delivery and Collection success index

In Figure 8-2, it can be noticed that random walk (RW) provides average delivery delay comparable to D-MDO for all topologies, however at the expense of very low packet delivery rates (Figure 8-5) and high delay variations. In Figure 8-5, a lower than 60% packet delivery ratio in most cases for RW is because while "random walking" the MDH may collect data from the same nodes multiple times before visiting other nodes. In fact, as seen in Figure 8-4, it may never visit few nodes, which explains the low collection success index as depicted in Figure 8-5. In addition, we found that the delay variation experienced by data packets for RW is large because of the large variation in the successive visits to the sensor nodes. Due to the low coverage, packet delivery ratio and high delay variation for data, RW is not suited for practical use in realistic sensor networks.

The collection success index represents the percentage of nodes from which at least one packet has been collected. Figure 8-5 shows the collection success index values for 3-Tri topology and it indicates that RW has collected data (at least one packet) from about 73% of the nodes while all the other strategies achieve close to 100%. The low collection index also helps the RW mechanism to reduce the average delivery delay by shortening the average duration between the MDH's visits to a node, and also by reducing the queuing delay at the visited nodes. Similar collection success index values were observed for other topologies. Also, as shown in Figure 8-5, the delivery ratio achieved by CENT, D-MDO and Area-Scan are significantly higher compared to RW. However, they are not 100% because few data packets remain uncollected in the node buffers when the simulation is terminated.

8.3 Multi-hop Data Collection ($k > 1$)

This subsection reports the impacts of varying k on various network performance metrics in a 60 node lattice sensor network (*Latt*) as shown in Figure 8-1. The value of k was varied from 1 to 3. For the chosen topology, k=4 represents the static sink case. Meaning, for this particular topology, there exits a topological location to place a static sink so that each node can reach the sink with up to 4 hop routing. Note that *D-PNT* is not included in this section because it does not apply to multi-hop data collection scenario.

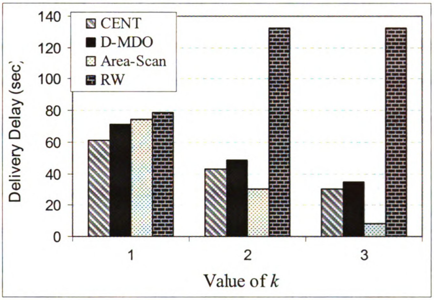


Figure 8-6: Impact of k on average *delivery delay*

Results in Figure 8-6 indicates that the packet *delivery delay* progressively goes down for protocols *CENT*, *D-MDO* and *Area-Scan* as the value of k increases. The scenario with k=1 has the highest *delivery delay* as it represents the single-hop collection. For all three mechanisms, as the value of k increases, the overlay construction process as outlined in Phase 1 of Section 4 identifies smaller number of Navigation Agents (and hence Designated Gateways), and therefore the sink has to visit fewer nodes during each *collection cycle* within the sensor field. This shrinkage of the sink trajectory length reduces the *trip collection-time* t_{coll} as reported in Figure 8-7.

Since the packet *delivery delay* is a function of the *trip collection-time*, the *delivery delay* in Figure 8-6 closely follows the *trip collection-time* trends in Figure 8-7. Note that the *trip collection-time* is not meaningful for the Random Walk (*RW*) solution [106] since the sink is almost guaranteed to not complete the round trip on a given *collection cycle*. That is why it is not included in the trip collection-time chart in Figure 8-7. As expected, for all *k*, the centralized protocol *CENT* chooses a smaller set of Navigation Agents than the distributed *D-MDO* mechanism as outlined in Section 7.2. This explains *CENT*'s shorter trajectory, *trip collection-time* and the packet *delivery delay*.

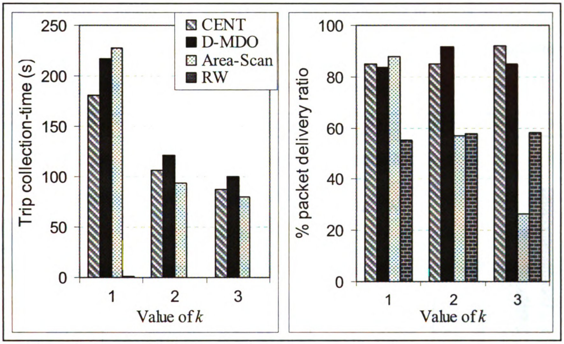


Figure 8-7: Impact of *k* on collection time and delivery

In the case of *Area-Scan*, as the MDH moves, it sends out a *k*-hop beacon in response to which all nodes within *k*-hop distance forward the data stored in their buffers to the sink. Unlike for the *CENT* and the *D-MDO* mechanisms, no Designated Gateways (DG) are used for data aggregation and upload to the sink. The increase in *k* leads to an increase in the amount of data that needs to be simultaneously collected by the sink. This increase in data causes MAC layer (802.11 has been used for these experiments) contentions and subsequent packet drops, which, in turn, causes the delivery ratio for the *Area-Scan* to degrade. This is evident in Figure 8-7. MAC contention does not pose such a problem for *D-MDO* and *CENT* because for these mechanisms the sensor nodes route their data to Designated Gateways (DG) where it is buffered prior to the arrival of the MDH. When the sink arrives, the data is uploaded to the sink using single-hop transmissions only from the DGs, as opposed to from all up to *k*-hop nodes as in the case of *Area-Scan*. This helps the MAC layer drops to remain insensitive to *k* for *CENT* and *D-MDO* protocols. Unlike other protocols, Random-Walk scheme experiences an increase in packet *delivery delay* with increasing *k*. This is caused primarily due to its unpredictable sink trajectory, although the packet delivery remains fairly insensitive to *k*.

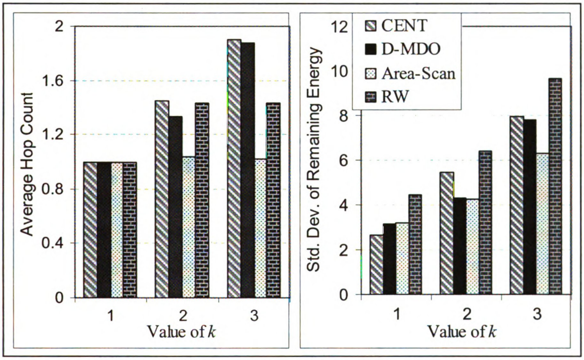
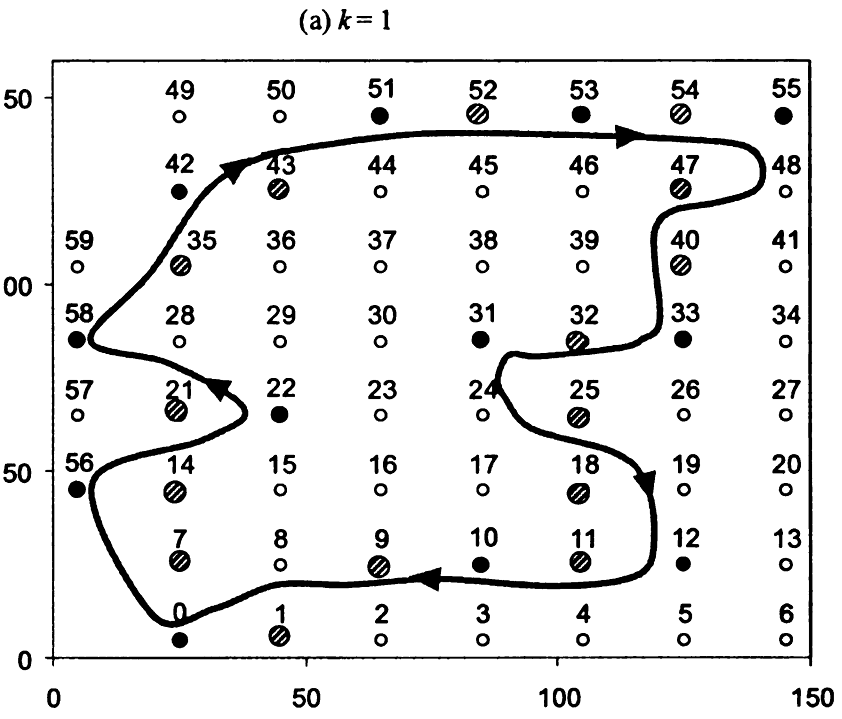


Figure 8-8: Impact of *k* on hop count and remaining energy

As shown in Figure 8-8, with increasing *k*, the average hop count for a data packet increases and hence the energy required to route it increases. The increase in multi-hop routing leads to uneven energy drainage, which, in turn, increases the *Standard Deviation of Remaining Energy* or *SDRE* metric as evidenced in Figure 8-8. The combined effect of the increase in energy/packet and *SDRE* leads to a decrease in network lifetime. Results in Figure 8-6 thru Figure 8-8 confirm the delay-lifetime tradeoff and its sensitivity to the hop bound factor *k*, as postulated in Chapter 6. Parameter *k* can be used as a 'knob' to achieve a balance between delay and lifetime.



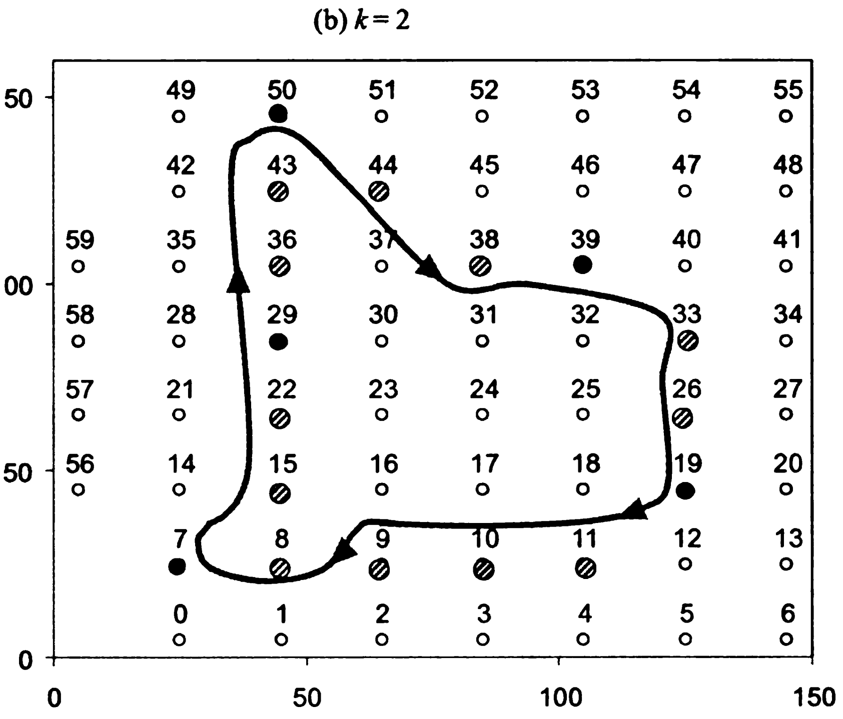
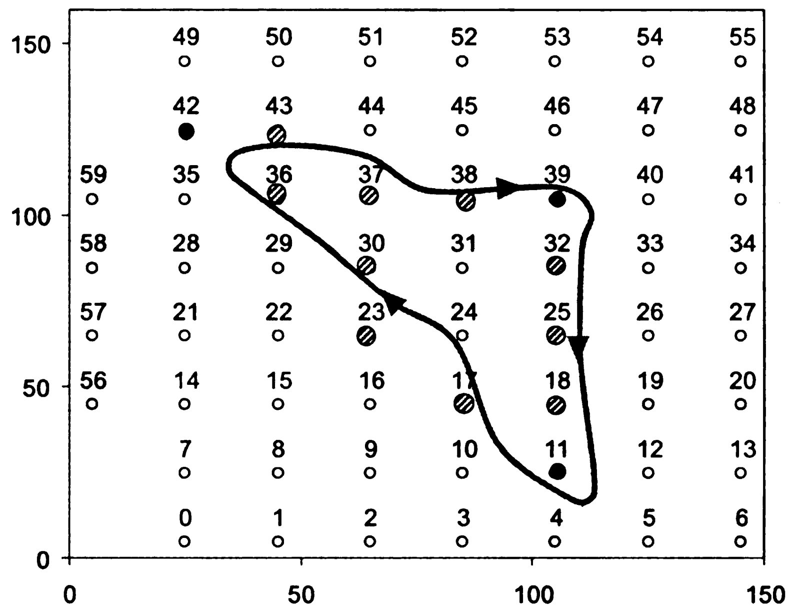


Figure 8-9: MDH trajectories for varying *k*

Figure 8-9: MDH trajectories for varying *k* (contd.)

(c) $k=3$



● Navigation Agent	▨ Intermediate Navigator

Figure 8-9 shows the D-MDO trajectory followed by MDH in the network for $k = 1, 2$ and 3. These experimentally observed trajectories visually confirm that as k increases, the number of NAs identified goes down which results in shorter trajectories and subsequently lower trip collection-time. Multi-hop data collection experiments were carried out on a number of additional topologies with different node counts, node distributions, and densities. Results from those topologies showed similar patterns, confirming the energy-delay tradeoff using hop-bound k as shown above.

8.3.1 Impact of Network Density

Experiments were also conducted with varying node density or node degree. With a constant sensor field area of 150m X 150m, the number of nodes was varied to achieve an average nodal degree of 6.5, 9.7 and 15. The results obtained are shown in Figure 8-10.

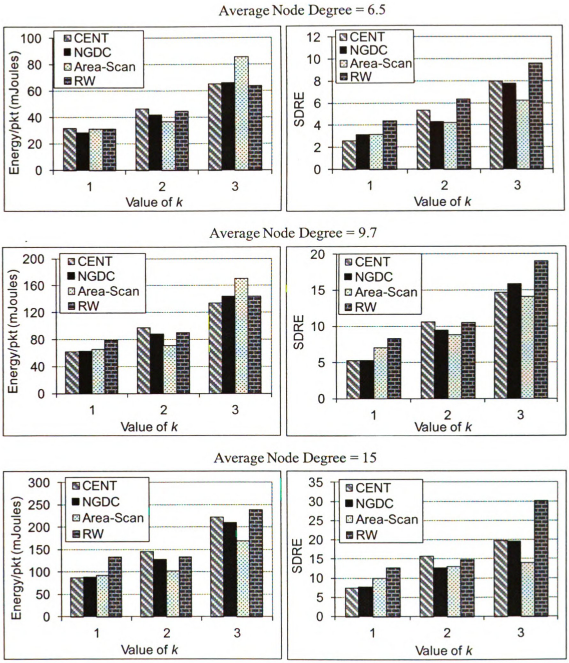


Figure 8-10: Impact of node density on energy metrics

The results for D-MDO indicate that with increasing density for a given value of *k*, the absolute values for the energy metrics AEPP and SDRE goes up. This indicates that the on the basis of the model analysis. Thus, for regions of different densities, it is clear that different values of k must be chosen to ensure that overall lifetime objective is optimized.

8.4 Performance Summary

The performance results lead us to conclude the following about the proposed NADC framework: 1) it reduces collection-time and hence *delivery delay* compared to other strategies while maintaining complete coverage and high delivery ratio 2) Direction information can be used for MDH navigation in the absence of location information 3) The tunable parameter k can be used as a knob to achieve a desired balance between delay and network life.

8.5 Summary and Conclusions

In Chapters 6 thru 8, a generalized non-real-time data collection framework for jointly determining the MDH trajectory and routing was developed. Analysis of the model for the framework established some bounds for the hop bound factor k. We explored the idea of using a network assisted sink navigation and data collection framework to perform k-hop sensor data collection without node localization services. We proposed a generalized solution framework and developed distributed algorithms which allow the sensor network to 1) identify a subset of nodes as sink navigation agent and data collection points, 2) compute a sink navigation path using ANT based distributed TSP solution, 3) identify Designated Gateways (DGs) and routing paths from sensors to DGs, and 4) dynamically navigate a sink by the navigation agents along the computed path. An NS2 based simulation model was developed to evaluate the proposed protocol and to compare it with a number of other data collection mechanisms including one with centrally computed navigation paths. Experimental results indicate that the proposed distributed mechanism achieves better performance than previously used distributed strategies and comparable to centralized solutions.

Chapter 9. Stability Analysis of Multi-hop Data Collection

9.1 Motivation

In Chapter 6, a model was developed based on the hop-bound factor k to represent different data collection scenarios, namely, single-hop with mobile sink, multi-hop with static sink, and different levels of mobile-sink-based multi-hop routing in between. In this chapter, this model is used as a means to study the broader impacts of multi-hop routing on data collection delay performance. The basic goal is to investigate if any arbitrary amount of multi-hop routing can be used in the context of mobile sink deployments. In the process, a number of thresholds of k that impact stability and acceptable performance will be derived from the model. Finally, the system performance trends obtained from the model are validated using a packet level simulation in ns2.

9.2 Routing Thresholds

9.2.1 Stop threshold

The MDH collects data from all *Designated Gateways* once every collection cycle (τ). Hence, during each cycle, all generated data during the most recent collection cycle needs to be uploaded to the MDH. Thus, T_u, the time required to upload the data that has been generated since the MDH's previous visit, can be expressed as $(\tau\lambda n)/C$. If $T_u \le T_p$, the MDH can collect all generated data without stopping at any DG. Therefore, the lowest value of k for which stops are required is referred to as the stop threshold k_{stop}. The condition for obtaining the stop threshold is:

$$\frac{\tau\lambda\pi R^2 d}{C}(\tau)/C > \frac{3L}{(2k_{stop}+1)v} .$$

Now substituting the value of L from Eq. 6-1, we get:

$$\frac{\left(\tau\lambda\pi R^2 d\right)}{C} > \frac{3\left(\pi R^2 / 2rk_{stop} + 2R_m\right)}{\left(2k_{stop}+1\right)v} \quad \text{...... (Eq. 9-1)}$$

Simplifying the equation, we obtain:

$c_1 k_{stop}^2 + c_2 k_{stop} + c_3 > 0$, where $c_1 = \dfrac{2\tau\lambda\pi R^2 dv}{3C}$, $c_2 = \dfrac{\tau\lambda\pi R^2 dv}{3C} - 2R_m$ and

$$c_3 = -\frac{\pi R^2}{2r}$$

After solving the above equation for k_{stop}, the positive root is considered as the stop threshold k_{stop}. If k is larger than this <u>*stop threshold*</u>, the mobile sink is required to stop around one or multiple DGs in order to collect all accumulated data since its last visit. Because of these mandatory stops, *Collect-on-Move* operation is not feasible for values of k beyond k_{stop}. Note that there is no distinction between the stop thresholds for the *Stop-on-Demand* and *Collect-on-Move* modes. This implies that if the value of k is above this <u>*stop threshold*</u>, the MDH has to be capable of reducing its speed, stopping or executing a holding pattern.

The stop threshold k_{stop} can be particularly significant in choosing the appropriate MDH technology for a given application. For example, since the fixed-winged UAVs are not able to stop or hover, the deployed amount of multi-hop routing has to be below the identified threshold k_{stop}. Alternatively, if a higher extent of multi-hop routing is needed from other application and system constraints such as delay and sensor network life, then a rotary-winged UAV technology will have to be used for satisfying the hover/stop requirements.

9.2.2 Stability threshold

As mentioned in Section 6.2, the system is stable only when the trip collection-time is smaller than the collection cycle, that is $t_{coll} \le \tau$. From Eq. 6-4, the condition can be written as $T_m \le \tau$ or $T_m + (T_u - T_p) \le \tau$ depending on the value of the stop component. In either case, as k increases, the quantities T_m and T_p decrease as seen in Eq. 6-2 and 6-4. The lowest value of k that allows the trip collection-time to be within the collection cycle τ is referred to as the stability threshold $k_{stability}$.

<u>*Stop-on-Demand*</u>: Since in this case the stop component of the collection time is non-zero, the condition for stability is $T_m + (T_u - T_p) \le \tau$. Substituting the values of T_m, T_p and T_u, we can write:

$$\frac{\left(\pi R^2 / 2k_{stability} r\right) + 2R_m}{v} + \frac{\lambda \tau \pi R^2 d}{C} - \frac{3\left(\pi R^2 / 2k_{stability} r + 2R_m\right)}{v\left(2k_{stability} + 1\right)} \le \tau .$$

Upon simplification, we get: $c_1 k_{stability}^2 + c_2 k_{stability} + c_3 \le 0$, where

$$c_1 = 2\left(\frac{2R_m}{v} + \frac{\lambda \tau \pi R^2 d}{C} - \tau\right), \quad c_2 = \frac{\pi R^2}{rv} - \frac{6R_m}{v} + \frac{\lambda \tau \pi R^2 d}{C} - \tau, \quad \text{and} \quad c_3 = -\frac{\pi R^2}{rv} .$$

Similar to the stop threshold, solving this equation and choosing the positive root provides the k-threshold for stable operation.

<u>*Collect-on-Move:*</u> In this case, since the stop component of the collection time is zero, the stability condition can be written as $T_m \le \tau$, or

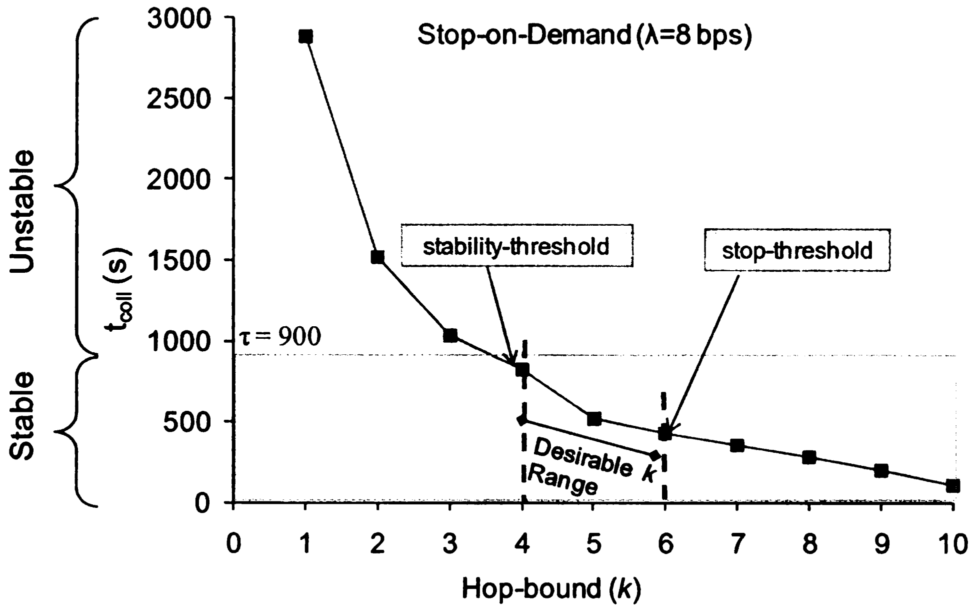
$$\frac{\left(\pi R^2 / 2k_{stability}\, r\right) + 2R_m}{v} \leq \tau .$$

Solving for $k_{stability}$, we get $k_{stability} \geq \left(\dfrac{\pi R^2}{2r(v\tau - R_m)} \right)$.

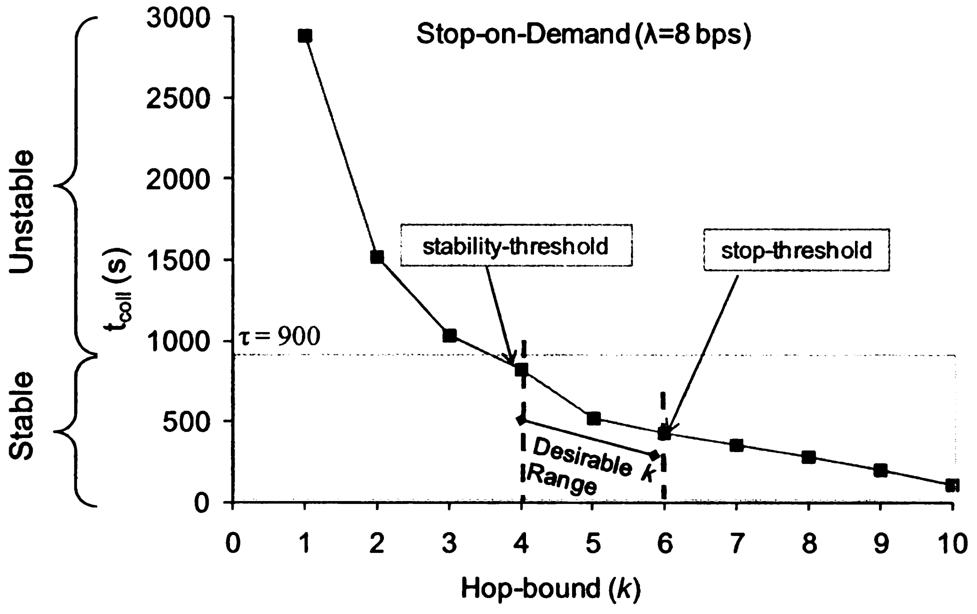
The results above indicate that certain degree of multi-hop routing can be essential in certain applications to ensure operational stability of a mobile sink in both Stop-on-Demand and Collect-on-Move modes. Note that this analysis assumes that the nodes have infinite buffers and therefore no packets are lost due to buffer overflow. The instability is defined only in terms of when the trip collection-time exceeds the collection cycle τ. However, when nodes have limited buffers, instability would also manifest in the form of unacceptable packet loss rates.

9.2.2.1 Bound for Data Generation Rate

Consider the stability condition: $T_m + (T_u - T_p) \leq \tau$. When the upload time T_u dominates both T_u and T_p , then $T_u \leq \tau$ or $(\tau \times \lambda \times n)/C \leq \tau$. This gives a bound $\lambda \leq C/n$ for the data generation rate from each node. Meaning, if λ becomes greater than C/n, the data rate is not sustainable and the system will not be stable. This is intuitive because the aggregated data generation rate of the entire network has to be less than the rate C at which the MDH can collect data.



(a)



(b)

Figure 9-1: Impact of hop-bound k on trip collection-time, t_{coll} in case of *Stop-on-Demand* collection

9.3 Numerical Evaluation of the Collection Time

<u>*Stop-on-Demand*:</u> Figure 9-1 shows the trip collection-time t_{coll} for the

Stop-on-Demand mode with varying *k*. Figure 9-1:a and Figure 9-1:b report the results for two different values of λ. For all λ, with increasing *k*, t_{coll} decreases monotonically, initially at a fast rate. With higher λ, it decreases at a lower rate. To understand this behavior, in Figure 9-2 we plot the quantities T_m, T_p and T_u. The move time and the productive move time T_m and T_p are dependent on *k* but not on λ. The upload time T_u, on the other hand, depends on λ but does not vary with *k*. It is evident that with increasing *k*, initially the quantity T_m decreases quite sharply, and then the rate of decrease reduces until *k* reaches a $k_{critical}$, which represents the static scenario as shown in Figure 6-1:a. This is because with increasing *k*, the sink trajectory includes fewer and fewer *Navigation Agents* and hence it becomes shorter.

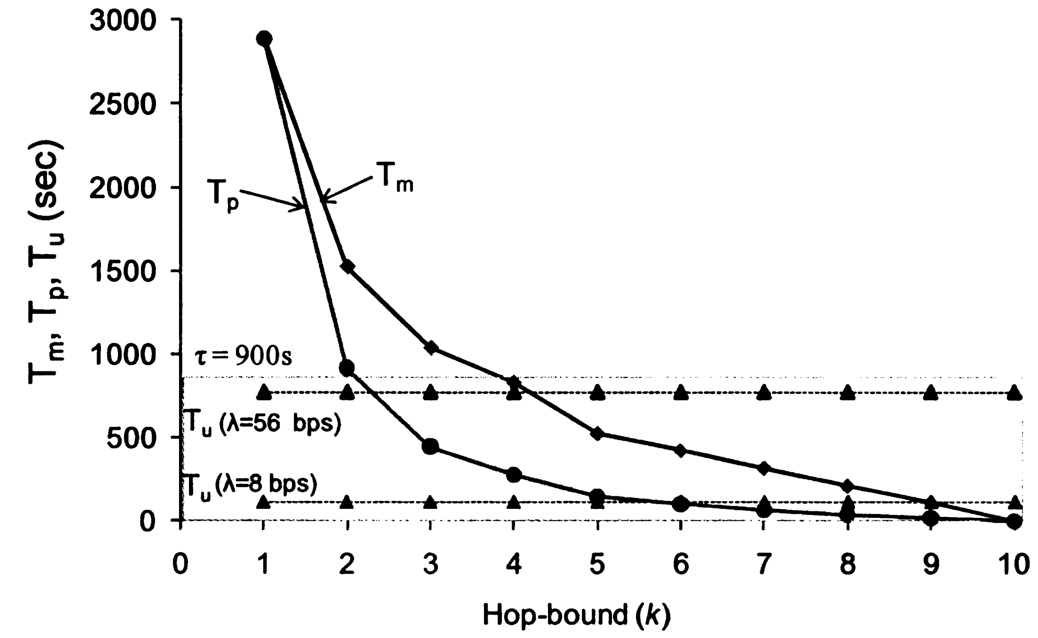
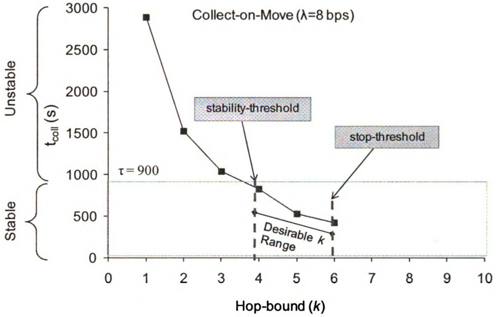
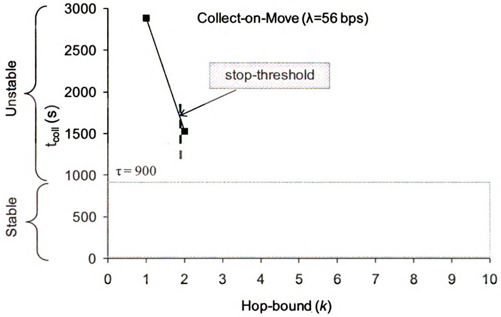


Figure 9-2: Impact of hop-bound *k* on T_m, T_p and T_u

The quantity T_p goes down at a much faster rate because as the *Navigation Agents* are spread farther apart, the MDH is not in contact with any DG during an increasing fraction in its trajectory. For all λ, as k increases, the collection-time goes down fast until the *stop threshold* k_{stop} is reached, and then with a slower rate until $k_{critical}$ is reached.



(a)



(b)

Figure 9-3: Impact of hop-bound k on t_{coll} in case of *Collect-on-Move* collection

As seen in Figure 9-1, t_{coll} is initially dominated by T_m, but as T_p drops below T_u (see Figure 9-2), the condition $t_{coll} > T_m$ (see Eq. 7-4) holds. At $k = k_{critical}$, t_{coll} is the same as T_u. For the $\lambda = 8$ bps case in Figure 9-1:a, the values of $k < 4$ define the region of instability as the trip collection-time becomes greater than τ. For $k < 6$, the MDH can collect data without stopping because $T_p > T_u$, and therefore $k_{stop} = 6$. For $k > 6$, the MDH has to stop at certain DGs for stable data collection. In contrast, for $\lambda = 56$ *bps* in Figure 9-1:b, the *stop threshold* $k_{stop} = 2$, and the stability threshold $k_{stability} = 9$. Thus, in this case the stop threshold occurs before the stability threshold.

Collect-on-Move: Figure 9-3 shows the impacts of varying *k* on trip collection-time for different data generation rates. In this case, the trip collection-time t_{coll} is equal to the move time, T_m as shown in Figure 9-2. In Figure 9-3, the stop threshold is identified for both scenarios $\lambda = 8$ bps and 56 bps. In this mode, the stop threshold as discussed in Section 9.2 defines the upper bound on the extent of multi-hop routing for stable operation. Hence, the values of *k* beyond k_{stop} ($k > 6$ for $\lambda = 8$ bps and $k > 2$ for λ=56 bps) are not shown in Figure 9-3. For $\lambda = 8$, the desirable range of k is $k \geq k_{stability}$ and $k < k_{stop}$. As shown in Figure 9-3:b, in the case of λ = 56 bps, no value of *k* supports stable *Collect-on-Move* operation. This is because values of $k < 4$ lead to unacceptably long trajectories while values $k > 2$ require stops.

Based on the analysis thus far, the following inferences can be drawn.

a) For given network parameters, there is a lower-bound of *k* ($k_{stability}$) for stable data collection which implies that single hop collection may not always be feasible.

b) There exists a threshold k_{stop} which impacts the choice of the MDH technology. For example, since the fixed-winged UAVs are not able to stop or hover, the deployed amount of multi-hop routing has to be below the identified threshold k_{stop}. Alternatively, if a higher extent of multi-hop routing is needed based on other application and system requirements such as delay and network life, then a rotary-winged UAV technology will have to be used for satisfying the hover/stop requirements.

c) In *Collect-on-Move*, the range of k (i.e. the extent of multi-hop routing) that allows stable operation is limited compared to *Stop-on-Demand.*

d) In applications where it is required to use value of $k < k_{stability}$ or $k > k_{stop}$, multiple sinks need to be deployed for stable collections.

e) There is no specific relationship between the thresholds $k_{stability}$ and k_{stop}.

f) If λ is low enough to allow $k_{stability} < k_{stop}$, then it is desirable to pick a k in between them to provide stable operation without the MDH having to stop. The reduction in t_{coll} beyond k_{stop} is small since stops have to be incorporated, but higher energy cost is incurred due to increased multi-hop routing.

g) MDH trajectory should be computed such that $T_p \approx T_m$ to push k_{stop} as high as possible, and thereby increase the desirable operating range for *Collect-on-Move*.

9.4 Properties of the Stop and Stability Thresholds

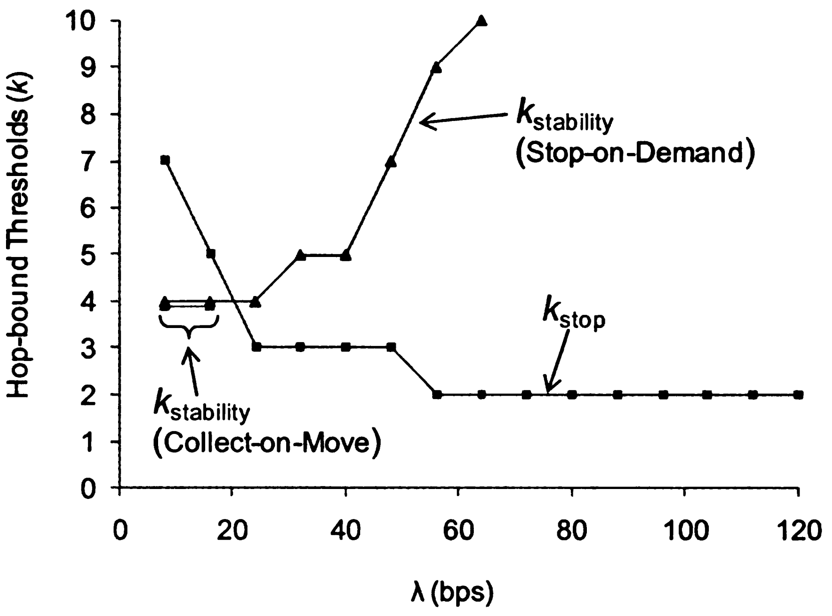


Figure 9-4: Impacts of data rate on stop and stability thresholds

<u>*Stop-on-Demand*</u>: As seen in Figure 9-1, the rate at which nodes generate data relative to the MDH-DG contact period can impact the stop and stability thresholds. Figure 9-4 shows the trends of $k_{stability}$ and k_{stop} with increasing data rate λ. As λ increases, $k_{stability}$ increases due to the higher data upload times (as seen in Figure 9-1:b) which result in higher t_{coll}. Hence, in order to maintain stability, it takes a higher value of k for the trip collection-time to be lower than the collection cycle τ. This implies that as the data rate goes up, the amount of multi-hop routing that is required to allow stable data collection also goes up.

The stop threshold, on the other hand, experiences a downward trend with increasing λ. This is because, the value of k for which the mobile sink has to stop and collect data decreases due to the increasing upload time. Thus, for sufficiently high data rates, even single hop data collection ($k = 1$) can require the sink to stop.

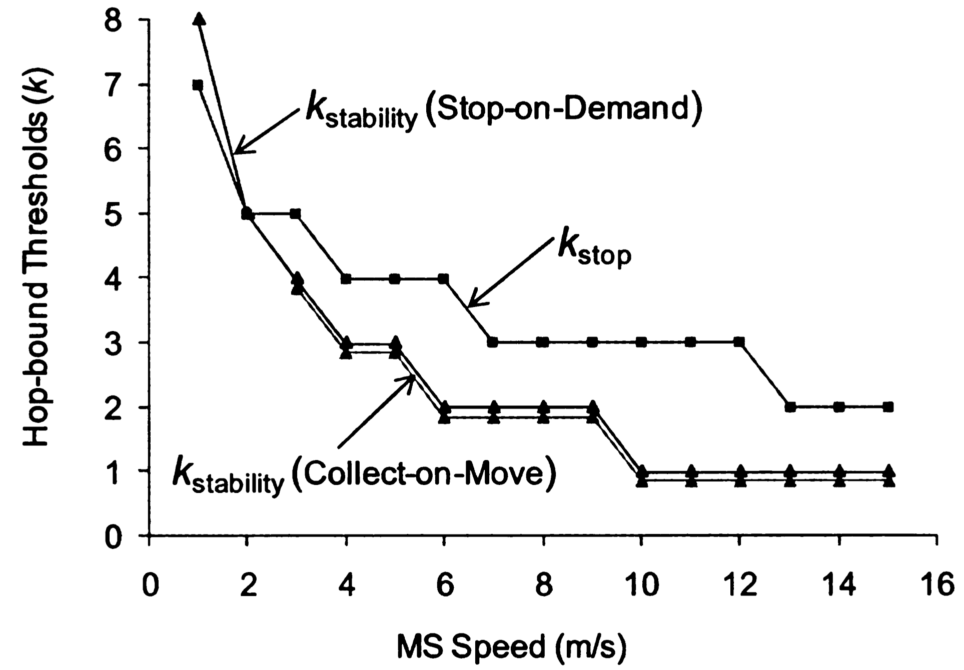


Figure 9-5: Impacts of sink speed on routing thresholds

Figure 9-5 reports the impacts of sink speed on the thresholds. As the sink speed increases, the DG-MDH contact times decrease, necessitating sink stops. This is why the stop threshold decreases. At the same time, since the mobile sink moves faster, the move-time T_m decreases. This effect reduces t_{coll} and enables lower and lower values of k to achieve trip collection-times below the collection cycle τ, pushing the stability threshold downwards. Thus, a faster moving sink will be able to support stable data collection with lesser degree of multi-hop routing. However, it will more likely to need the stopping/hovering capability at a relatively lower data rate.

The impact of node density is shown in Figure 9-6, in which the trend resembles the data rate scenario (in Figure 9-4) to some extent. The stop threshold goes down as the network density increases because more nodes and hence more packets are buffered at the DGs. The stability threshold goes up as longer stops become essential to collect the buffered data.

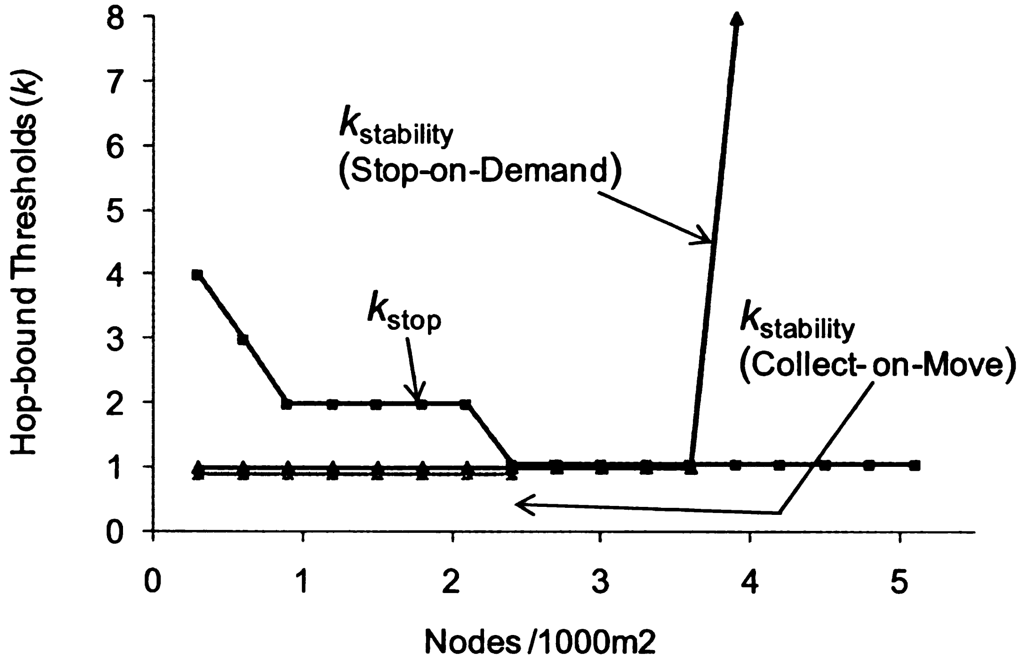


Figure 9-6: Impacts of node density on routing thresholds

Collect-on-Move: As mentioned in Section 6.3, for a *Collect-on-Move* sink, the trip-collection time is equal to its travel time. As seen in Figure 9-3, with increasing data rate, the stop threshold moves down. Since the values of *k* above the stop threshold are not stable, the range of *k* usable in this mode is restricted to $k < k_{stop}$ and $k \geq k_{stability}$. It can be observed that the range of *k* for stable operation continually shrinks with increasing data rate until it reaches zero. As shown in Figure 9-6 similar logic holds when the node density is increased.

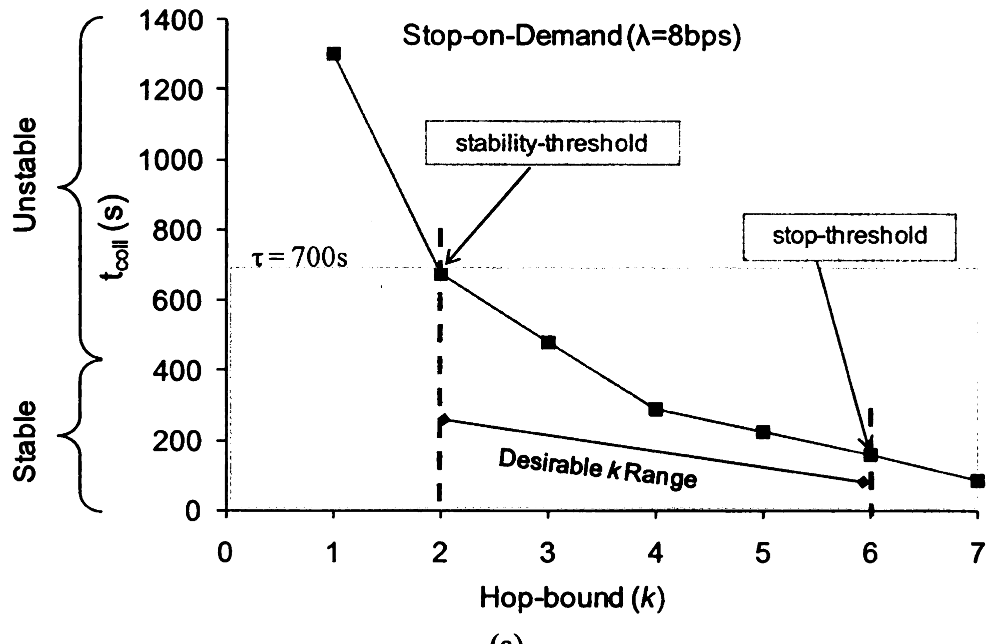
Similar to the *Stop-on-Demand* case, with increasing speed, the sink move time reduces and hence the $k_{stability}$ progressively goes down. The threshold k_{stop} goes down as well due to decreasing contact time with the DGs as mentioned earlier.

9.4.1 Applicability to Non-circular Terrains

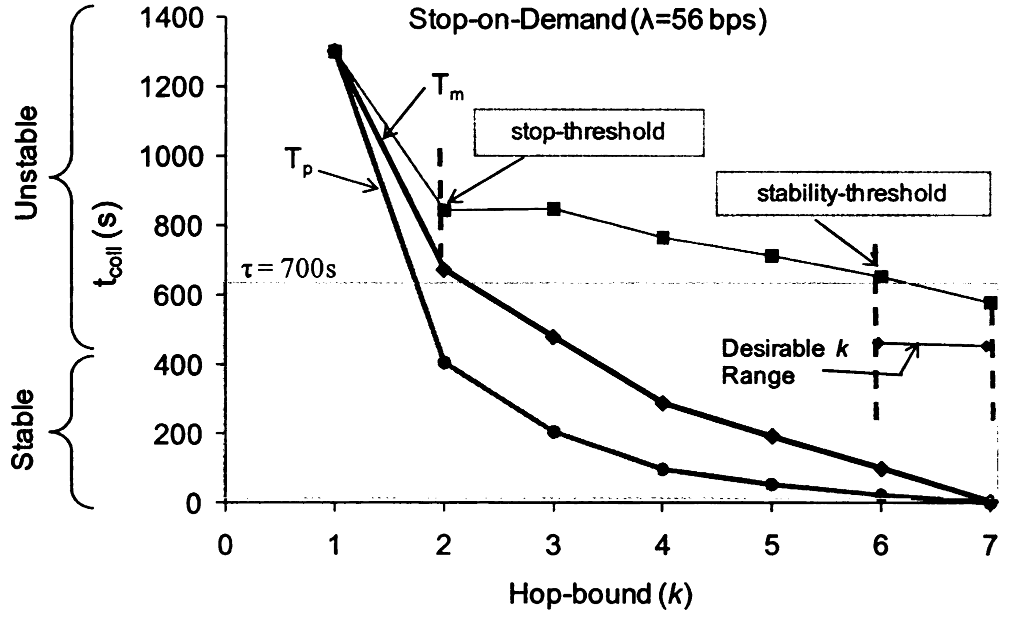
Although the stop and stability thresholds (i.e. k_{stop} and $k_{stability}$) have so far been determined and analyzed for a circular network terrain, the analysis in Sections 9.2.1 and 9.2.2 can be also applied to other non-circular terrains as long as the trajectory length, L and its resulting move time T_m are monotonically decreasing functions of k (see Figure 9-1 and Figure 9-2). Regardless of the terrain shape, by definition, as the hop-bound factor k increases, the amount of data that needs to be uploaded via each Designated Gateway (DG) increases. That means, for a given per-sensor data generation rate, the number of needed DGs decreases. Since DGs are chosen to be the 1-hop neighbors of the Navigation Agents, reducing the number of DGs gives rise to fewer needed Navigation Agents. The end result is a shorter trajectory L, and the resulting move time T_m. Therefore, the condition is true that both L and T_m reduce monotonically with increasing hop-bound factor k, confirming that the analysis in Sections 9.2.1 and 9.2.2 will hold for general terrain shapes in the presence of hop-bounded routing as defined in this thesis.

To validate the existence of the stop and stability thresholds for other terrain shapes, we repeated the related analysis with a square shaped terrain. We consider a square region whose side is A meters long. A trajectory consisting of concentric squares can be constructed similar to the one for circular terrain as shown in Figure 6-5. The trajectory length L is computed as:

$$L = 4 \sum_{i=1}^{A/2\sqrt{2}kr} \left[A - \left(\sqrt{2}kr(2i-1) \right) \right] + 2R_m \qquad \text{..........} (6)$$



(a)



(b)

Figure 9-7: Impacts of k on t_{coll} for square network region

We evaluated the trip collection-time for square network terrain with side =500m, transmission range =50m and τ =700s. For this network $k_{critical} = 7$. The trip collection-time results obtained for different data generation rates for *Stop-on-Demand* data collection are shown in Figure 9-7. It can be observed that the results and their patterns are very similar to those shown in Figure 9-1 for the circular terrain.

9.5 SIMULATION EXPERIMENTAL RESULTS

9.5.1 Impacts of Data Rate

In Section 9.4, it was observed that the data generation rate of the nodes has impacts on the stop and stability thresholds. In addition, increasing data rate can affect the data collection performance due to higher MAC layer contention and the resulting collisions. We conducted data collection experiments on a 144-node network with nodes arranged in the form of a regular square lattice. With this network size and the chosen wireless transmission range, the hop-bound factor k can be varied from 1 through 5 for the mobile sink scenario. The case for $k = 6$ represents the static sink scenario indicating that $k_{critical} = 6$.

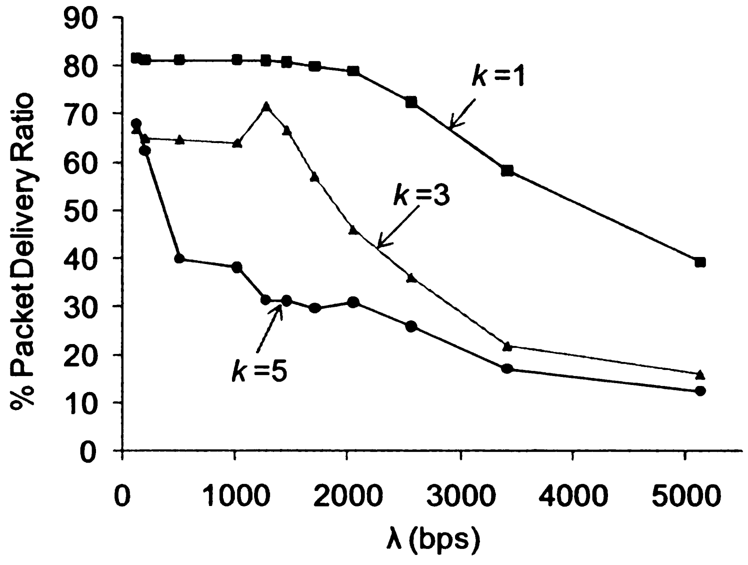
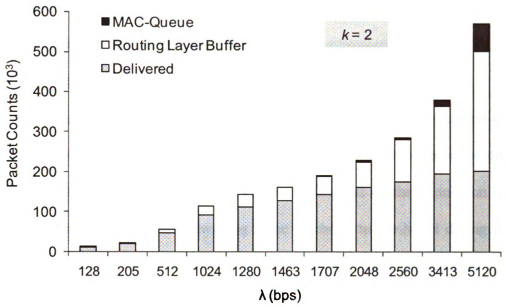
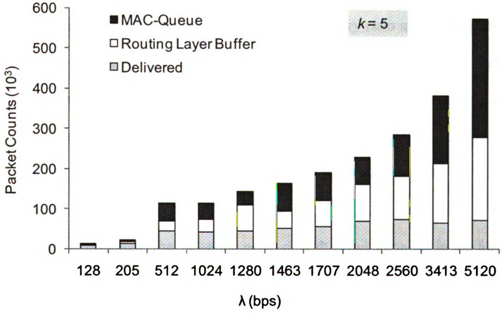


Figure 9-8: Packet delivery performance for varying hop-bound factors (k)



(a)



(b)

Figure 9-9: Delivery and drop performance for varying data rate when (a) k=1, and (b) k=5

Figure 9-8 shows the packet delivery ratio performance with varying data rate (λ) for different k values. In order to analyze the delivery results, we collected packet level statistics which include the number of packets:

a) delivered to the mobile sink

b) buffered at the routing layer in the DGs

c) dropped by the Data Link Control (DLC) layer interface queue

The following observations can be made from the results. First, the packet delivery performance is lower in general for higher k for all λ values. The number of delivered packets in Figure 9-9 corroborates this effect. At the end of the simulation, packets remain in the routing layer buffer waiting to be uploaded to the MDH during the next collection cycle. At lower data rates, higher the value of k, larger the amount of data accumulated at the buffers at the end as seen in Figure 9-9. At higher data rates, higher k leads to higher drops from the DLC interface queue because of MAC collisions and also insufficient contact duration with the MDH. Second, delivery ratio in general goes down significantly after a certain data rate. Higher the value of k, the precipitous decline in delivery ratio occurs sooner. For instance, the decline occurs at λ=2048, 1280 and 128 bps for k=1, 3 and 5 respectively. This can also be explained by higher collisions and drops due to limited contact duration with the MDH, as seen in Figure 9-9.

9.5.2 Thresholds for Hop-bound (*k*)

We conducted experiments the 144-node network with nodes arranged in the form of a regular square lattice mentioned earlier. The trip collection-time obtained for varying k is shown in Figure 9-10. As predicted by the model in Figure 6-7 and seen in Figure 8-3 for a 60-node network, we can observe that as the hop-bound increases, t_{coll} goes down due to the reduction in trajectory length. Figure 9-11 depicts the variation of the stability threshold $k_{stability}$ with increasing collection cycle τ. Since the stability threshold $k_{stability}$ indicates the minimum amount of multi-hop routing that allows system stability

($t_{coll} < \tau$), as τ decreases, the amount of multi-hop routing that is required for stability goes up. In other words, higher the degree of multi-hop routing (i.e. higher k), lower the collection cycle duration that can be supported by the system.

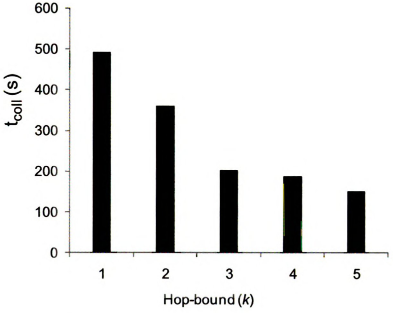


Figure 9-10: Variation of Trip collection-time

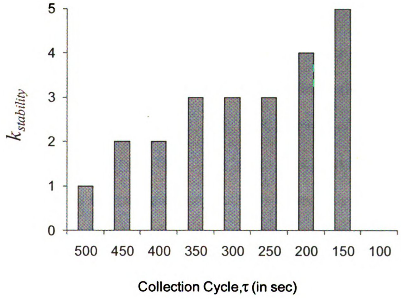


Figure 9-11: Variation of stability threshold

Now, we explore the impact of data generation rate on the thresholds. As observed in Section 9.4, while the stability threshold $k_{stability}$ is expected to be insensitive to the data generation rate λ for *Collect-on-Move* mode, the stop threshold k_{stop} is not. Guided by this observation, we experimentally investigate the impacts of λ on k_{stop}. In order to determine the stop threshold k_{stop} for different data rates, it is necessary to determine the *k* at which the MDH-DG contact duration becomes insufficient to collect data buffered at the DGs.

To find that threshold value of *k* experimentally, it is important to identify scenarios in which the sink moves out of the range of a node when it still has packets in its wireless network interface queue. Such events happen when the DG-MDH contact duration is insufficient (i.e. the value of *k* is already larger than k_{stop} for the given network parameters), as a result of which the interface queue starts dropping packets after the MDH leaves prematurely. Stop thresholds identified using this experimental mechanism are reported in Figure 9-12.

Figure 9-12 and Figure 9-13 show the impacts of data generation rate and sink speed on the experimentally found stop threshold. The MDH speed was varied from 3 m/s (typical speed of mobile sensor robots [83]) to 25 m/s (typical speed of mini-UAVs [150]).In Figure 9-12, it can be observed that for a given sink speed, the stop threshold progressively go down as the data rate increases. This behavior is in agreement with the trend of k_{stop} as found via the analytical model in Section 9.5.1 and Figure 9-4. As the data rate increases, the amount of data buffered at the DGs increases implying that the sink would need to stop at lower values of *k*. For a given λ, we can also observe that the stop thresholds are lower for higher speeds. This can be attributed to the fact that increasing speed results in shorter contact durations between the MDH and the DGs.

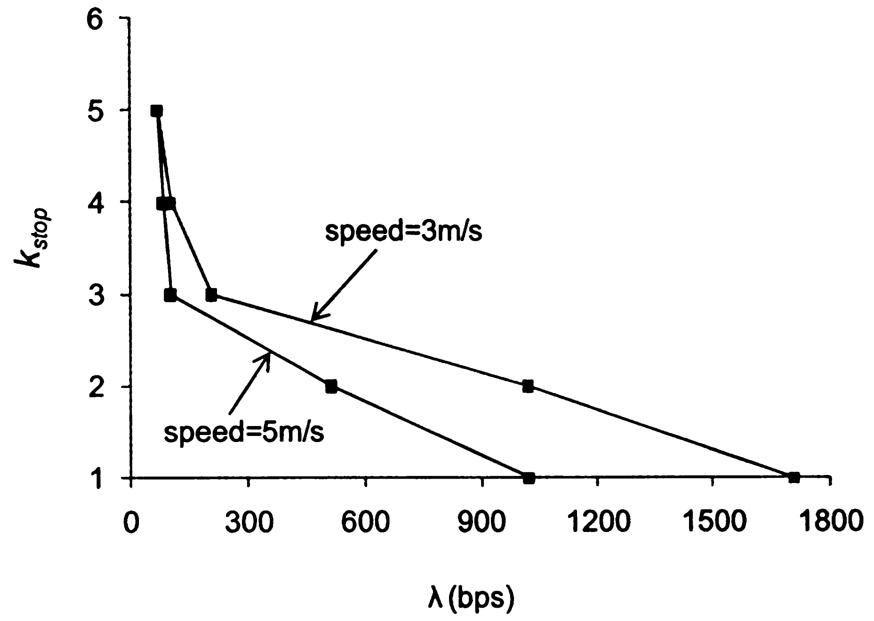


Figure 9-12: Impact of data rate on the stop threshold

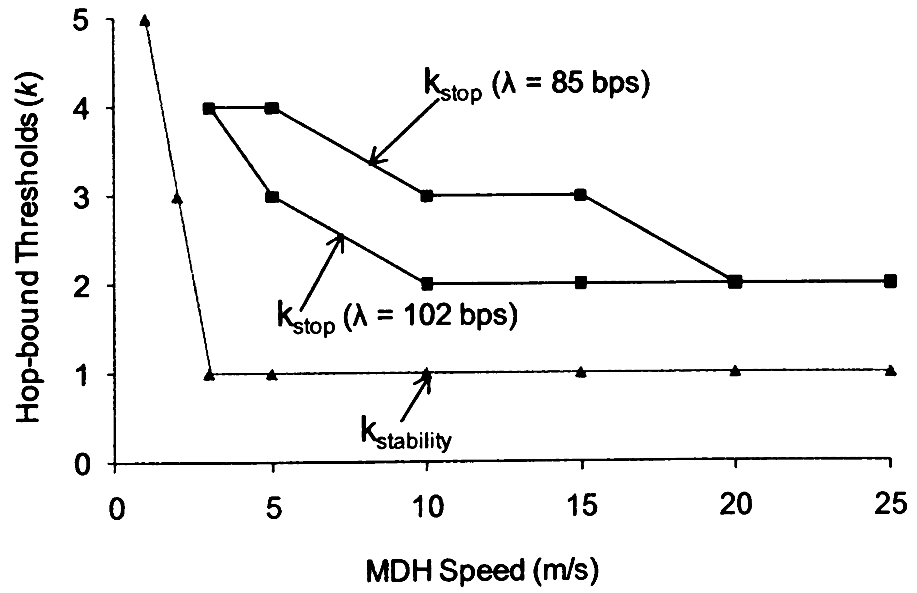


Figure 9-13: Impact of speed on the stop threshold

The stability threshold is not affected by the changing data rate, but it is sensitive to speed as shown in Figure 9-13. As the speed increases, the stability threshold moves down because the sink move time T_m progressively reduces. For example, with a sink speed of 1m/s, while all $k \geq 5$ represent stable operation, for 2m/s, all $k \geq 3$ are stable. For speeds 3m/s and higher, all values of k are stable. Figure 9-13 also shows the impacts of sink speed for different values of λ on k_{stop}. As the sink speed increases, the stop threshold moves down for reasons mentioned above. The higher the data generation rate, the lower the stop threshold. This behavior also matches the trends found through the analytical model in Section 9.4 and Figure 9-5.

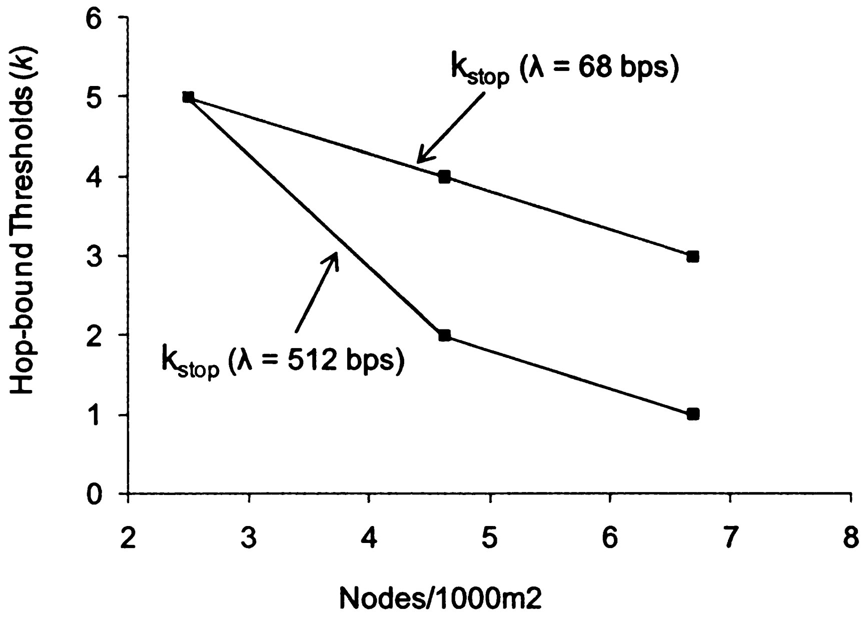


Figure 9-14: Impact of node density on stop threshold

Figure 9-14 shows the impacts of node density on the stop threshold at different data generation rates with a sink speed of 3m/s. At this speed, the value of $k_{stability}$ turns out to be 1, which corresponds to single hop data collection. This also means that all possible values of k can offer stable operation. As predicted from the model (see Figure 9-6) it was experimentally found that the node density does not affect the stability threshold for the *Collect-on-Move* operating mode. The stop threshold, on the other hand, reduces (see Figure 9-14) with increasing density. This trend matches with what was observed from the model in Figure 9-6. For a given data rate, with increased node density, a sink has to collect data from an increased number of DGs around each *Navigation Agent*. Hence the amount of data to be collected within the DG-MDH contact duration increases, requiring stops at lower levels of multi-hop routing. This brings the stop threshold down. For a given density, higher the data rate, the lower the stop threshold as shown in Figure 9-14.

9.5.3 Guidelines for Choosing Hop-bound Factor *k*

The analysis presented above shows a tradeoff between energy consumption and data collection-time for the variable amount of multi-hop routing represented by the parameter k . Based on this observation, the following recommendations for choosing k can be made.

1) The sink technology dictates the choice of k. For sink devices that are incapable of stopping, k has to be smaller than the stop threshold k_{stop}. For sinks that can stop, $k > k_{stop}$ can be chosen.
2) For a given collection cycle τ, k should be chosen such that $k \geq k_{stability}$.
3) If the sensor application has an upper bound on tolerable delay, then the minimum value of k (subjected to the conditions $k \geq k_{stability}$ and $k \leq k_{stop}$ irrespective of the relative values of $k_{stability}$ and k_{stop}), that meets the delay requirement should be chosen. This ensures that while meeting the delay requirement, the network life is elongated as much as possible.

4) An energy-constrained MDH such as UAV can typically move for a limited duration (say α) before it needs to be recharged or refueled. For a given τ, the minimum value of k that results in $t_{coll} < \alpha$ should be chosen in such situations. In a hostile terrain, a UAV style mobile sink may also have a required upper bound of the exposure time during data collection. The same k selection strategy could be used with α defined here as the maximum allowable exposure time.
5) MDH trajectory should be computed such that $T_p \approx T_m$ to push k_{stop} as high as possible, and thereby to increase the desirable operating range.

9.6 Summary and Conclusions

In this chapter, the limits on the extent of multi-hop routing that can employed in the context of mobile sink data collection were investigated. To this end, hop-bound k thresholds referred to as the stop threshold and stability threshold were derived. The stop threshold represents the k beyond which the MDH has to switch to *Stop-on-Demand* mode. The stability threshold, on the other hand represents the minimum value of k that results in a trip collection-time that satisfies the collection cycle (τ) constraint. These thresholds are used to analyze the working of *Collect-on-Move* and *Stop-on-Demand* collection modes from a stability standpoint. As a result, the feasible region of operation for the hop-bound factor between 1 and $k_{critical}$ are identified. The impact of key network and MDH parameters such as data generation rate, density and MDH speed are investigated. The result trends that were obtained from the model are validated with simulation experiments. Based on the insight gained from the analysis, guidelines for choosing the value of k were identified.

Analytical and experimental results presented in this chapter prove that there are clear limits to the extent of multi-hop routing (k) that can be used in a network. In other words, single hop data collection using a mobile sink ($k = 1$) or static sink data collection may not always be feasible. The feasible range of k is affected by a number of network and MDH parameters such as the data generation rate, node density, collection cycle duration, MDH speed and the data collection mode. For *Collect-on-Move* mode, the possible feasible range scenarios are shown in Figure 9-15. The scenarios are obtained by varying the data rate, but the same holds good when other parameters are varied. For low data rate, when $k_{stop} > k_{stability}$, the range is $k_{stability}$ to k_{stop}. On the other hand, when $k_{stop} < k_{stability}$, there is no value of k that is conducive to stable operation.

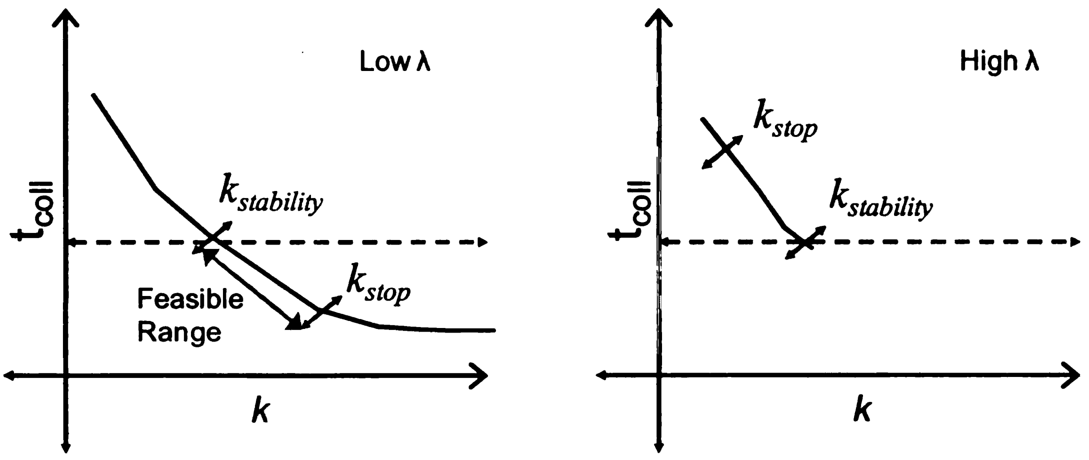


Figure 9-15: Feasible range for *Collect-on-Move* mode

Similarly, Figure 9-16 shows the feasible range possibilities for *Stop-on-Demand*. Regardless of the data rate and the relative positions of $k_{stability}$ and k_{stop}, the feasible range is $k_{stability}$ to $k_{critical}$.

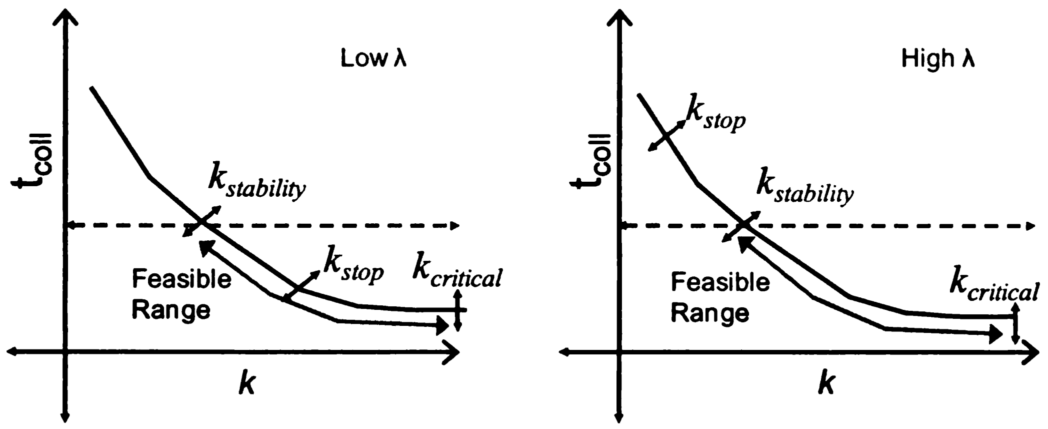


Figure 9-16: Feasible range for *Stop-on-Demand* mode

Thus, for a given set of application parameters, the data collection mode dictates the feasible amount of multi-hop routing. The mode, in turn directly depends on the choice of mobile sink technology (ability of sink to stop or hover) and vice versa. Experimental results show that the choice of *k* outside the feasible region can lead to poor data collection performance. Therefore, appropriate choice of *k* is very important to ensure stable and high delivery data collection.

Chapter 10. Capacity-Aware Stability Analysis

10.1 Motivation

The generalized formulation of the data collection problem presented in Chapter 7 and the stability analysis performed in Chapter 9 apply to networks where the data rate is low compared to the capacity of the MDH-DG links. Results presented in Figure 9-8 and Figure 9-9 show that as the data rate increases, the delivery and drop performance deteriorates. Also, as the data rate increases, lower values of *k* become more appropriate to maintain reasonable delivery performance. Also, the trajectory planning mechanisms proposed in the existing literature consider the sink-sensor link capacity constraint which plays a significant role in determining the data collection performance in data intensive applications. In the light of insights gained thus far, the goal of this chapter is to extend the model to accommodate the capacity constraint into account and study the impact on the collection performance and the thresholds identified in Chapter 9.

10.1.1 Feasibility Range for Stable Operation

As explained in Section 6.2, in order for the system to be stable, the trip collection-time t_{coll} should be less than or equal to the collection cycle τ. Since t_{coll} is the cumulative time required for the MDH to physically move along the trajectory and duration of all the necessary stops, the system could be unstable in any of the following situations. First, when the length of the trajectory is so high that the time required to complete it is higher than τ. Now, as explained in Section 6.6.2.1, the trajectory length is inversely proportional to *k*. Therefore, the value of *k* below which the MDH traversal time over the trajectory is greater than τ, marks the lower bound of the feasible range of *k*, which is denoted as k_{st_low}. At low data rates, this corresponds to $k_{stability}$ discussed in Chapter 6.

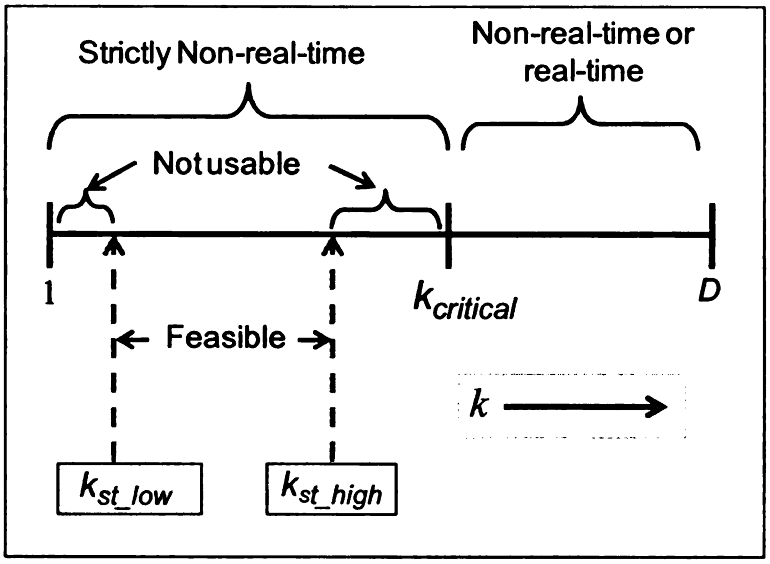


Figure 10-1: Feasible range within non-real-time collection

Second, when the MDH needs to stop at the DGs for so long that the overall trip collection-time is larger than τ. As *k* increases, for a given data rate, the MDH needs to collect more data from each DG. Therefore, for a given wireless link capacity, the MDH has to stop at DGs, leading to higher t_{coll} values. The value of *k* at which t_{coll} becomes higher than τ due to those high stop durations (caused by high data upload times) marks the upper bound for the feasible range of *k*, which is denoted by k_{st_high}. For given network density, data origination rate, MDH-DG link capacity and the MDH speed, the problem of determining the feasible range of *k* within the non-real-time operating region is equivalent to identifying k_{st_low} and k_{st_high} as shown in Figure 10-1.

10.2 Theoretical Analysis of Feasibility Range

10.2.1 Modifications to the Model

Data Upload Time:

As seen in Section 9.2.1, the upload time T_u depends on the capacity of the link (C) between the MDH and the DGs. However, the total link capacity, *C* is not available for the DGs to upload data to the mobile sink due to the interference caused by transmissions in the neighborhood of the DGs. Thus, the upload time is dependent on the effective capacity of the link between the DGs and the MDH referred to as C_{eff} and $C_{eff} < C$. The upload time is equal to the amount of data generated in the network since the previous visit by the MDH divided by the effective capacity of the MDH-DG links. Thus, $T_u = {(\tau \times \lambda \times n)}/{C_{eff}} = {(\tau\lambda\pi R^2 d)}/{C_{eff}}$.

The effective capacity of the MDH-DG link can be expressed as the total link capacity minus the link capacity lost due to interference namely C_{in}. Thus $C_{eff} = C - C_{in}$. If a separate channel is used for MDH-DG communication, then the transmissions around the DGs do not interfere with the MDH data collection and hence the entire capacity of the channel is available for uploading data resulting in $C_{eff} = C$. This provides an upper bound for the data collection performance that can be achieved by the mobile sink. On the other hand, if the channel used by the mobile sink for data collection is same as the channel used for communication between the sensors, then the extent of interference is dependent on the data rate of the nodes, the node density and the hop-bound factor *k*. When the MDH is visiting a Navigation Agent and collecting data from the DGs, interference is caused by the nodes up to two hops away from the MDH sending data to the DGs. The worst case for interference or C_{in} occurs when all the data generated in the network gets forwarded to the DGs. C_{in} can be computed as the contact time of the MDH with the DGs around an *Navigation Agent* multiplied by the amount of data (bps) forwarded by interfering nodes. The contact distance with at least one DG in the vicinity of a *Navigation Agent* can range from *2r* (when a *Navigation Agent* and all its DGs are collocated) to *4r* (when at least two DGs of a *Navigation Agent* are located on two diametrically opposite points on the circle of radius *r* around the *Navigation Agent*) leading to an average of *3r*. The data forwarded by the interfering nodes is essentially all the data generated in the network except the data generated by the DGs themselves. Thus, we have:

$$C_{in} = {}^{3r}\!/_{v}\left[\pi(kr)^2\lambda d - \pi r^2 \lambda d\right] = {}^{3\pi r^3 \lambda d\left(k^2-1\right)}\!/_{v} \quad \text{..............}(Eq.\ 10\text{-}1).$$

From Eq. 10-1, it is clear that the amount of interference increases as a function of increasing data rate, node density or *k* for a given transmission range and MDH speed. Substituting the value of C_{in}, the upload time is given by:

$$T_u = \frac{\left(\tau\lambda\pi R^2 dv\right)}{\left(Cv - 3\pi r^3 d\lambda(k^2-1)\right)} \quad \text{.......}(Eq.\ 10\text{-}2)$$

10.2.2 Stable Operating Range

As mentioned in Section 6.2, the system is stable only when the trip collection-time is smaller than the collection cycle or $t_{coll} \le \tau$. Thus, in the case of *Stop-on-Demand* collection, Eq. 7-5 implies that $T_m + \left(T_u - T_p\right) \le \tau$. Therefore,

$$\left[\frac{L}{v} + \frac{\tau\lambda\pi R^2 vd}{\left(Cv - 3\pi r^3 \lambda d(k^2-1)\right)} - \frac{3L}{\left(2k+1\right)v}\right] \le \tau \quad \text{.....}\ (Eq.\ 10\text{-}3)$$

Eq. 10-3 and Eq. 6-1 indicate that as *k* goes up, the trajectory length *L* goes down and therefore, T_m and T_p (i.e. first and the third terms in Eq. 10-3) go down. At the same time, with increasing *k*, T_u the second term goes up. Therefore, it can be expected that as *k* increases, t_{coll} (the left hand side of Eq. 10-3) goes down up to a certain point and then as the second term becomes dominant, it goes up again. Thus, we can expect to have a range of k for which $t_{coll} \le \tau$ that defines the stability region. The lower threshold referred to as k_{st_low} below which the system is unstable and higher threshold is k_{st_high}. Note that this analysis assumes that the nodes have infinite buffers and therefore no packet losses are considered. The instability is defined only in terms of trip collection-time when it exceeds τ. However, when nodes have limited buffers, instability would also manifest in the form of unacceptable packet loss rates.

10.3 Impact on Stop Threshold

The MDH collects data from all Designated Gateways (DGs) once every collection cycle (τ). Hence, during each cycle, the data that has been generated during the most recent τ units of time has to be uploaded. As mentioned earlier, T_u is the time required to upload the data. If $T_u \le T_p$, the MDH can collect all the generated data without stopping at any DG. In this case:

$$(\tau \times \lambda \times n)/C_{eff} \le 3L/(2k+1)v \text{ or } k \ge (3LC_{eff}/2\tau\lambda nv) - 1/2$$

In other words, $k_{stop} = \left(\dfrac{3LC_{eff}}{2\tau\lambda nv}\right) - \dfrac{1}{2}$ defines the threshold beyond which the MDH has to incorporate stops. As pointed out in Chapter 9, this implies that if the value of k is above this *stop threshold*, the mobile sink has to be capable of reducing its speed, stopping or executing a holding pattern.

10.4 Impact of Capacity-awareness: Numerical Evaluation

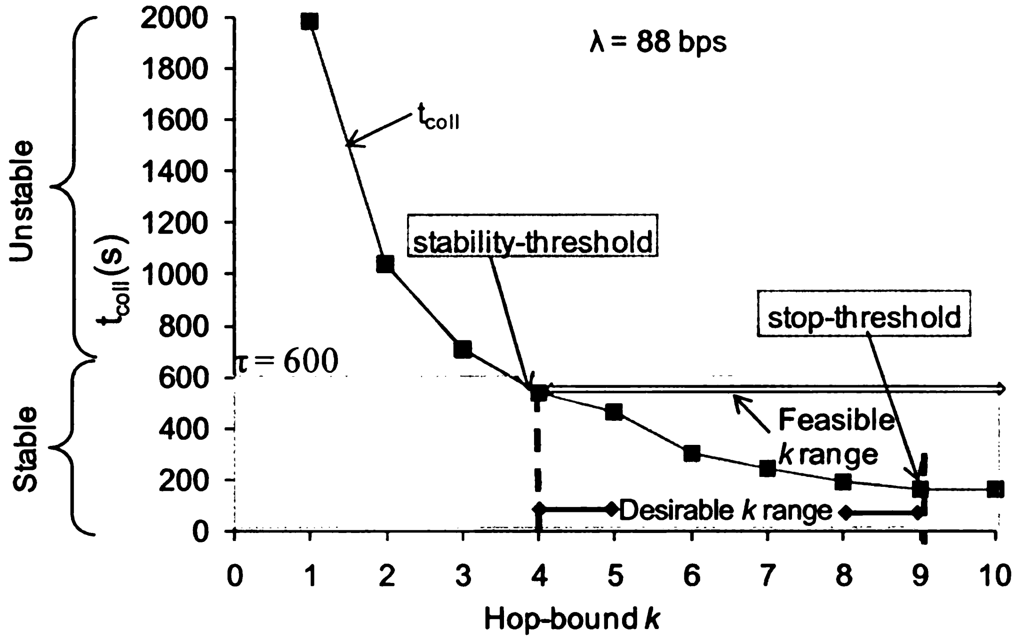
The numerical evaluation of trip collection-time that was performed in Section 9.3 was repeated using the expression for upload time obtained in Eq. 10-2. Trip collection-time for a circular network terrain of radius 300m and sensor transmission range of 30m was obtained by using Eq. 6-4. Thus, the minimum number of hops at which static sink operation is possible is 300/30 = 10 hops (i.e. $k_{critical}$ =10). The value of the collection cycle, τ was set to 600s and the MDH speed v was set to 3m/s.

10.4.1 Trip Collection-time

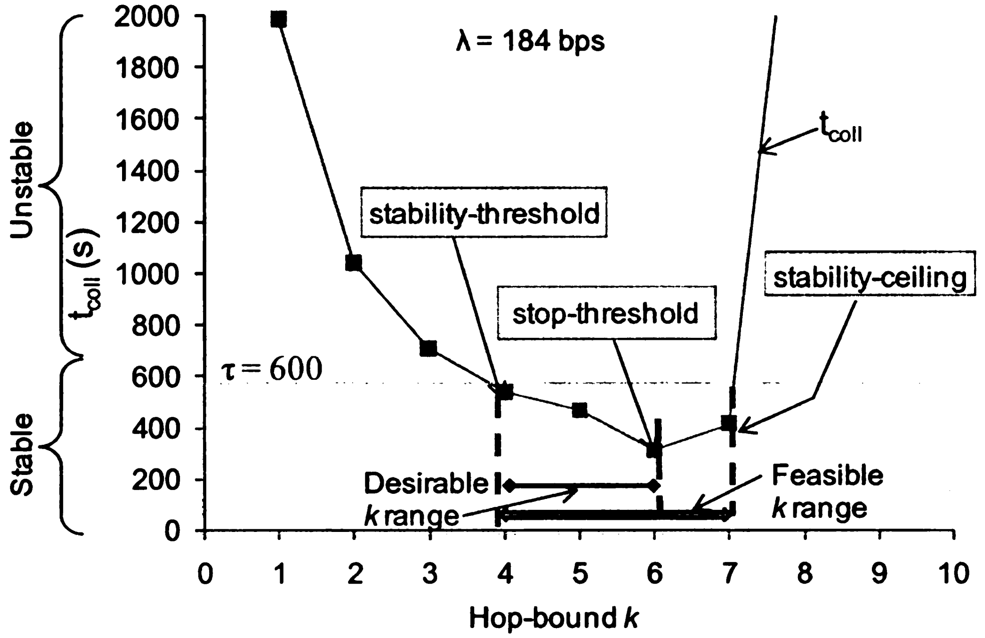
Figure 10-2 shows the trip collection-time t_{coll} for different values of λ and k ranging from 1 to $k_{critical}$. For all λ, as observed in Section 6.6.2.1, with increasing k, the collection-time, t_{coll} goes down fast until the *stop threshold* k_{stop} is reached. In case of low λ, t_{coll} goes down at a slower rate or stabilizes as seen in Figure 10-2:a. On the other hand, when λ is large, the trip-collection time goes up as k increases beyond the stop threshold. To understand the behavior of trip collection-time, we plot T_m , T_p and T_u in Figure 10-3. The behavior of move time, T_m and productive move time, T_p is same as observed in Figure 9-2. These quantities are not affected as they are not sensitive to link capacity. On the other hand, data upload time, T_u depends on the effective MDH-DG link capacity as mentioned in Section 10.2.1. In Figure 10-3, it can be observed that T_u is relatively constant at low λ. As the data rate increases, the upload time increases due to reduced effective capacity of the MDH-DG link. In Figure 10-3, the upload time, T_u increases with k due to the increasing interference experienced by the MDH-DG links resulting in lower effective capacity, C_{eff} . Higher the data rate, higher the rate at which T_u increases as observed in Figure 10-2 for $\lambda = 88$ and 184 bps.

Note that values of $k \leq 4$ are unstable irrespective of the data rate as the move time, T_m and therefore the resulting t_{coll} is higher than τ. For sufficiently high values of *k* and λ, the trip collection-time goes down and then begins to increase as the upload time becomes dominant. In Figure 10-2:b, this phenomenon can be observed for λ = 184 bps where values of $k \geq 7$ results in $t_{coll} > \tau$ and therefore are unstable. Thus, there exists a stability ceiling, k_{st_high} beyond which the system is not stable. Thus values between k_{st_low} and k_{st_high} constitute the *stable* or *feasible* range for the hop-bound factor *k*. When λ=88 bps, the feasible range is given by $k_{st\_low} \leq k \leq k_{critical}$ or $4 \leq k \leq 10$. On the other hand, for λ=184 bps, the stable range is $k_{st\_low} \leq k \leq k_{st\_high}$ or $4 \leq k \leq 7$. This implies that for sufficiently large value of λ, static sink scenario ($k \geq k_{critical}$) data collection may not be stable. In such situations, using mobile sink for data collection and choosing an appropriate *k* can make data collection possible.

As pointed out in Section 9.2.1, the stop threshold k_{stop} indicates the point beyond which MDH has to stop during its trajectory to complete data collection. Therefore, the rate at which the trip collection-time decreases with *k* is lower after the stop threshold. This implies that the energy cost associated with higher amount of multi-hop routing may not be worthwhile. Hence, the desirable range of *k* is given by $k_{st\_low} \leq k \leq k_{stop}$. Note that it is possible for the stop threshold, k_{stop} to be lower than k_{st_low} implying that all values of *k* that result in stable operation require stops. In such cases, *k* has to be chosen from the feasible region.



(a)



(b)

Figure 10-2: Impact of *k* on trip collection-time ($C_{eff} < C$) for different data rates

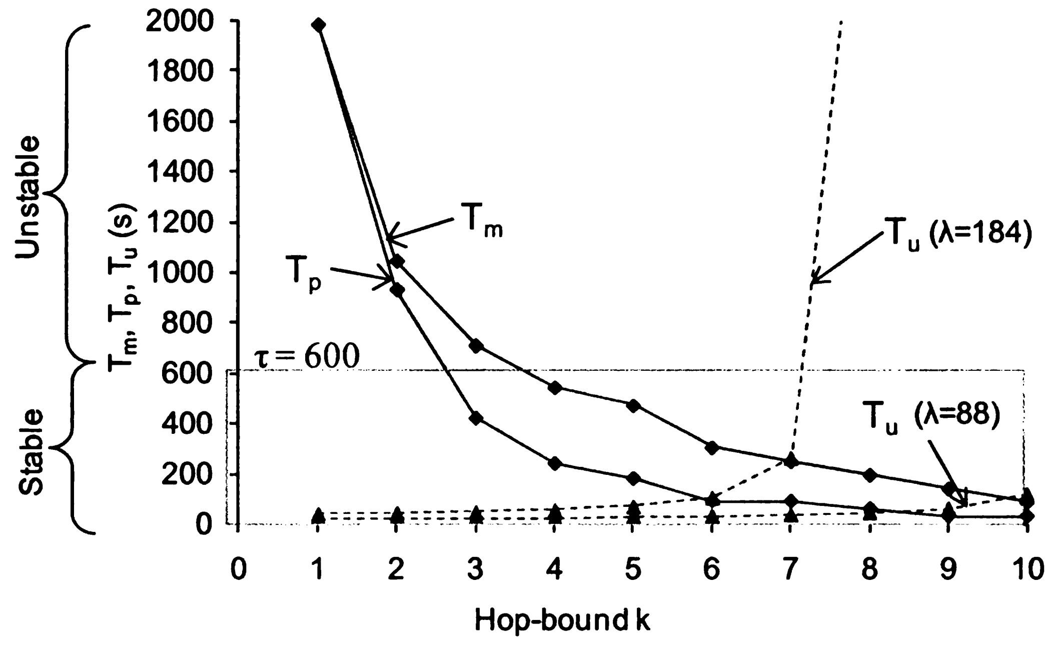
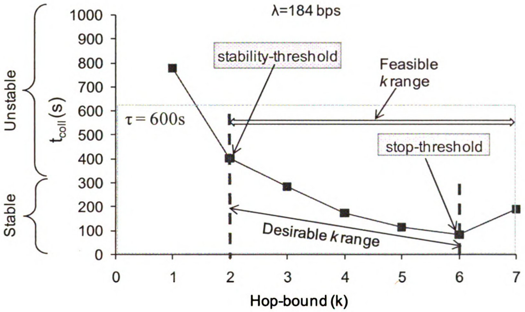


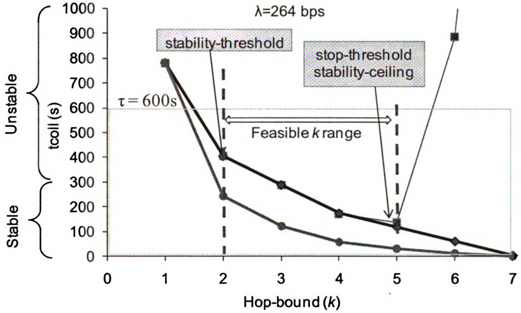
Figure 10-3: Impact of k on T_m, T_p and T_u

Figure 10-2:a depicts the scenario when the MDH-DG communication channel has the same capacity as that of the DG-sensor channels. In this case, the effective capacity C_{eff} is lower than the total link capacity C. When a separate channel is used for MDH-DG communication, then the transmissions in the vicinity of the MDH do not cause any interference to the DG-MDH data upload rendering $C_{eff} = C$. In this case, the upload time, T_u is not sensitive to k and only depends on λ . Therefore, t_{coll} goes down monotonically with increasing k. In other words, the behavior of t_{coll} is very similar to what is observed for λ = 88 bps in Figure 10-2:a. This separate channel scenario provides the upper bound for data collection performance. Figure 10-2:b on the other hand, assumes that the MDH-DG and the sensor-sensor communication occurs on a single channel causing maximum interference for the upload activity. This case provides the worst case T_u and trip collection-time performance and hence represents the lower

bound for k_{st_high}.



(a)



(b)

Figure 10-4: Impacts of *k* on t_{coll} for square network region

Non-circular Terrain: The numerical evaluation performed was repeated with square terrain for the reasons outlined in Section 9.4.1. We evaluated the trip collection-time for square network terrain with side = 500m, transmission range =50m and τ = 700s. For this network $k_{critical} = 7$. The trip collection-time results obtained for different data generation rates for *Stop-on-Demand* data collection are shown in Figure 10-4. It can be observed that the results are very similar to those shown in Figure 10-2 for a circular region. Thus, the trip collection-time trends and the observed thresholds hold in general.

10.4.2 Stop and Stability Thresholds

In this section, we focus on the impact of various network parameters such as data rate, node density, link capacity and MDH speed on the thresholds associated with *k* and therefore the feasible range of *k*.

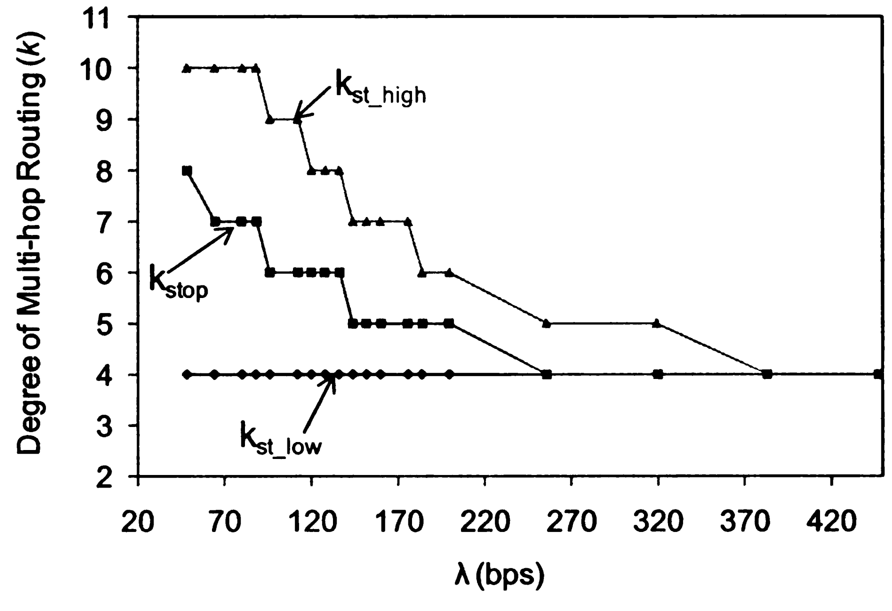


Figure 10-5 : Impact of data rate on the stability range and stop threshold

Figure 10-5:a shows the impact of data rate on k_{st_low}, k_{st_high} and k_{stop}. As the data rate increases, the value of k_{st_low} stays constant because it depends only the move

time for the trajectory which is only dependent on k. However, k_{st_high} goes down as the amount of data that needs to be collected goes up increasing the stop durations ultimately causing trip collection-time to exceed τ. Consequently, the range of k that is stable goes down until it shrinks to zero implying that the system is not stable for any amount of multi-hop routing. For instance, in Figure 10-5, for λ=384 bps, the stability range is zero. On the other hand, k_{stop} goes down with increasing data rate as the upload time resulting in the need for stops at lower and lower values of k. Note that k_{stop} is always lower than the stability ceiling i.e. $k_{stop} \leq k_{st\_high}$. Since inclusion of the stops due to reducing capacity is what drives the t_{coll} up, stop threshold has to lie before k_{st_high}.

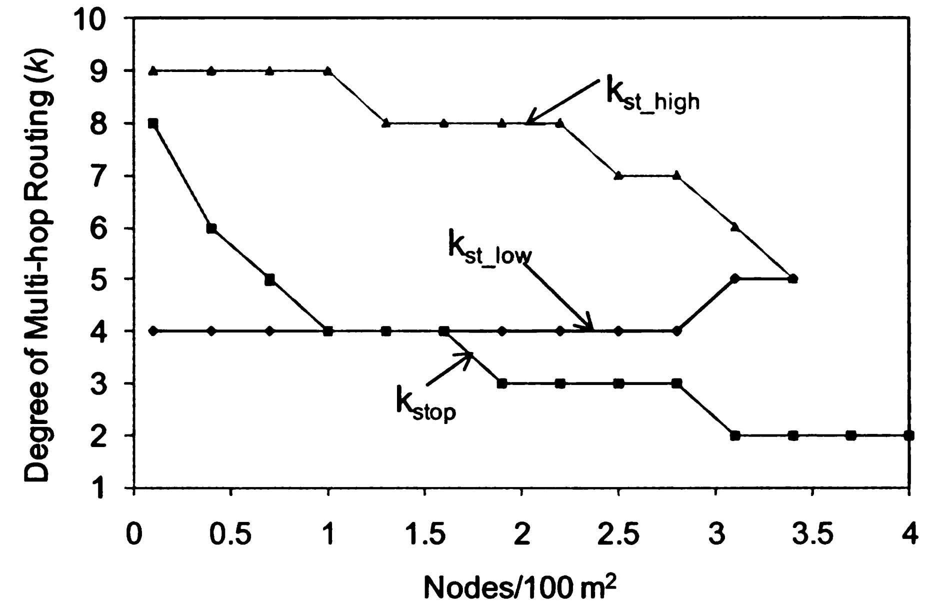


Figure 10-6: Impact of node density on the stability range and stop threshold

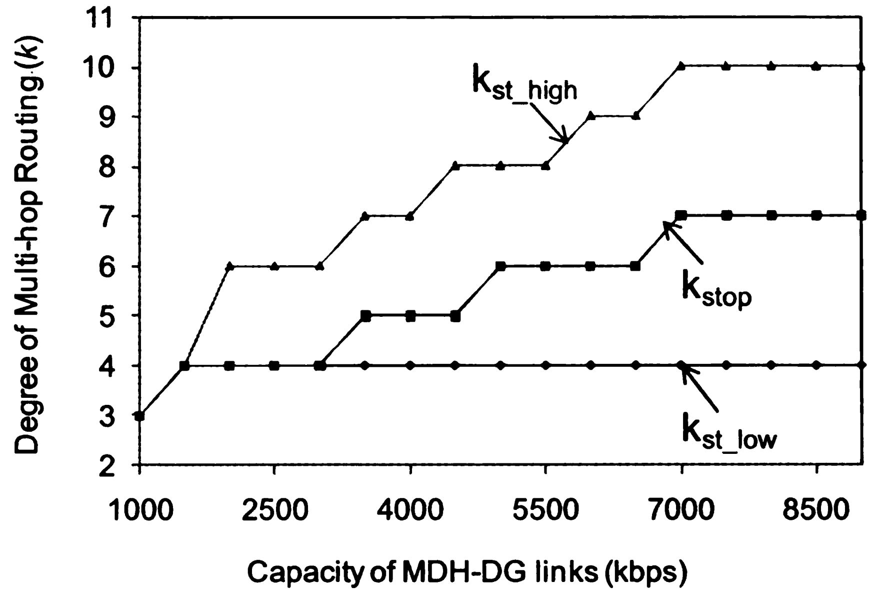


Figure 10-7: Impact of link capacity on the stability range and stop threshold

Figure 10-6 shows the impact of increasing node density on the thresholds. Node density affects the thresholds in the same manner as λ because in both cases the amount of data that has to be collected from each DG goes up. In other words, the stability range shrinks with increasing density and the stop threshold goes down. The stability threshold, k_{st_low} increases due to the increasing trip collection-time resulting from inclusion of stop durations. At the same time, the stability ceiling k_{st_high} comes down due to the increasing trip collection-time resulting from higher upload times due to reduced effective link capacity.

For a given node density and λ, Figure 10-7 shows the impact of the capacity of the links on the stability range. The impact is exactly the opposite of increasing the data rate or density. This is because as the link capacity increases, the MDH stop durations shrink or are not required. This leads to reduced t_{coll} causing k_{stop} and k_{st_high} to move upwards enlarging the stability range. The impact of MDH speed is shown in Figure 10-8.

As the speed increases, the time required to complete the trajectory or the move time, T_m goes down. As a result, the time during which data can be collected, T_p also goes down resulting in stop durations being required at lower values of k. Therefore, k_{stop} goes down. The decreasing move time also leads to significantly lower trip collection-time at lower values of k causing the stability threshold, k_{st_low} to go down.

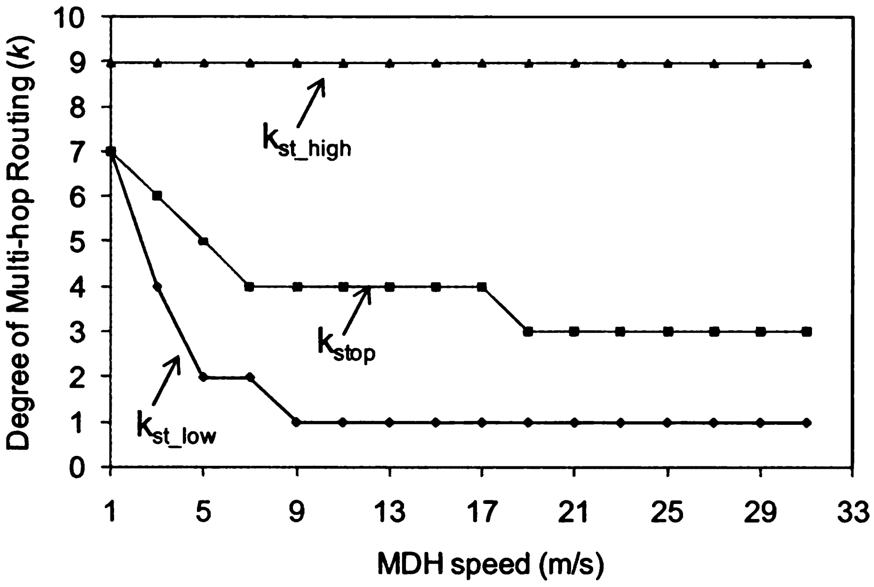


Figure 10-8: Impact of MDH speed on the stability range and stop threshold

10.4.3 Discussion

Based on the analysis thus far, the following inferences can be drawn.

1. For given network parameters, there is a lower-bound of k (k_{st_low}) for stable data collection which implies that single hop collection may not always be feasible. If MDH-DG link and sensor-sensor links use the same channel for communication, then for sufficiently high values of λ and k, there exists a stability ceiling, k_{st_high} beyond which the system becomes unstable.

2. For sensor network scenarios with sufficiently high data generation rate static sink data collection can be unstable. In such cases, employing a MDH for data collection offers a viable solution.
3. In applications where it is required to use value of $k < k_{st\_low}$ or $k > k_{st\_high}$ or $\lambda > C/n$, multiple sinks should be deployed for data collection.
4. If λ is low enough to allow $k_{st\_low} < k_{stop}$, then it is desirable to pick a k in between them. The reduction in t_{coll} beyond k_{stop} is small since stops have to be incorporated, but higher routing congestion is incurred due to increased multi-hop routing.

10.5 Performance Evaluation via Simulation Experiments

The validity of the performance trends predicted by the model was verified using an ns2 simulation model with IEEE 802.11 as the MAC layer protocol. If data collection is incomplete within the contact duration, the MDH stops at the *Navigation Agent* to upload all the buffered data from the DGs before moving to the next DG. Note that the simulation results are not expected to be numerically comparable with the results in Section 10.4 for two main reasons. First, the model assumes a perfect collision-free MAC, whereas the simulation uses 802.11 with all its associated overheads, including collisions. Second, the model assumes infinite buffer availability whereas in simulation buffers are limited.

We conducted our baseline experiments on the 144-node network with nodes arranged in the form of a regular square lattice. As mentioned before, with this network size and the chosen wireless transmission range of 30m, the hop-bound factor k can be varied from 1 through 5 for the mobile sink scenario. The case for $k=6$ represents the static sink scenario indicating that $k_{critical} = 6$. Unless otherwise stated, the speed of the MDH was set to 3m/s. The duration of the collection cycle (τ) was set to 500s. For all experiments, we measured the trip collection-time.

10.5.1 Impact of Data Rate on Performance

The trip collection-time obtained by varying the data generation rate for different values of *k* is shown in Figure 10-9. The results indicate that for lower data rates (λ=128 bps) as *k* increases, the trip collection time, t_{coll} goes down steadily due to reduction in trajectory length as seen in Figure 10-10. However, for higher values of data rate ($\lambda \geq 512$ bps) although t_{coll} tends to decrease as *k* increases initially, the trip collection-time tends to go back up at higher values of *k*. The initial decrease in t_{coll} that is observed with increasing *k* is driven by shrinking trajectory length. However, at the same time, there is an opposing effect that tends to push the trip collection-time upwards. This effect is driven by two phenomenon that occur as *k* increases namely 1) the amount of data accumulated at each DG increases due to higher multi-hop routing 2) the amount of interference in the network also increases resulting in reduced effective capacity of the DG MDH link. These effects lead to higher upload times forcing the MDH to incorporate stops into the trajectory. Inclusion of stops into the trajectory dampens the reduction of t_{coll} as *k* increases and at some point overtakes the t_{coll} reduction due to shorter trajectory.

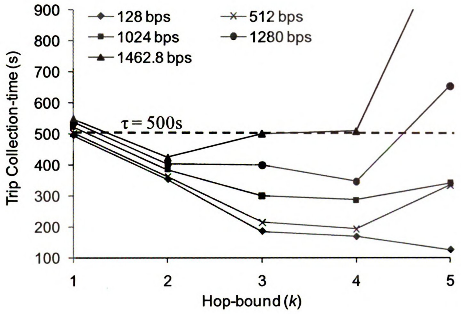


Figure 10-9: Impact of *k* on trip collection-time

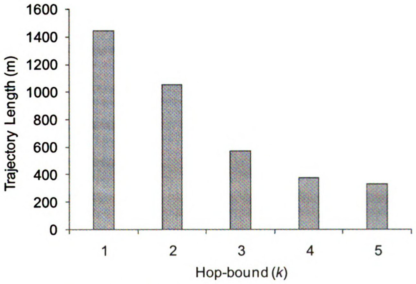


Figure 10-10: Impact of *k* on trajectory length

In Figure 10-9, it can also be observed that as the data rate increases the rate at which the trip collection-time reduces is lower. Also, the value of k at which the trip collection-time goes up is lower. For example, for λ=1024 bps, the trip collection-time goes up at k=5 while for λ=1462.8 bps, it happens at k=3. Thus, the feasible range shrinks as the data rate goes up until no *k* is feasible. These trends match the predictions of the analytical model as shown in Figure 10-2.

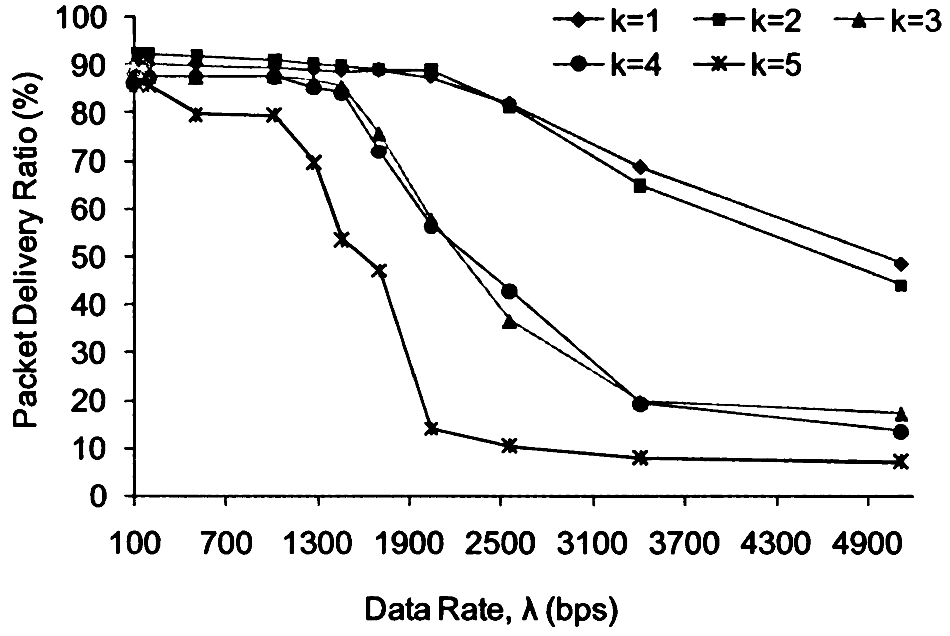


Figure 10-11: Packet delivery performance for varying data rates

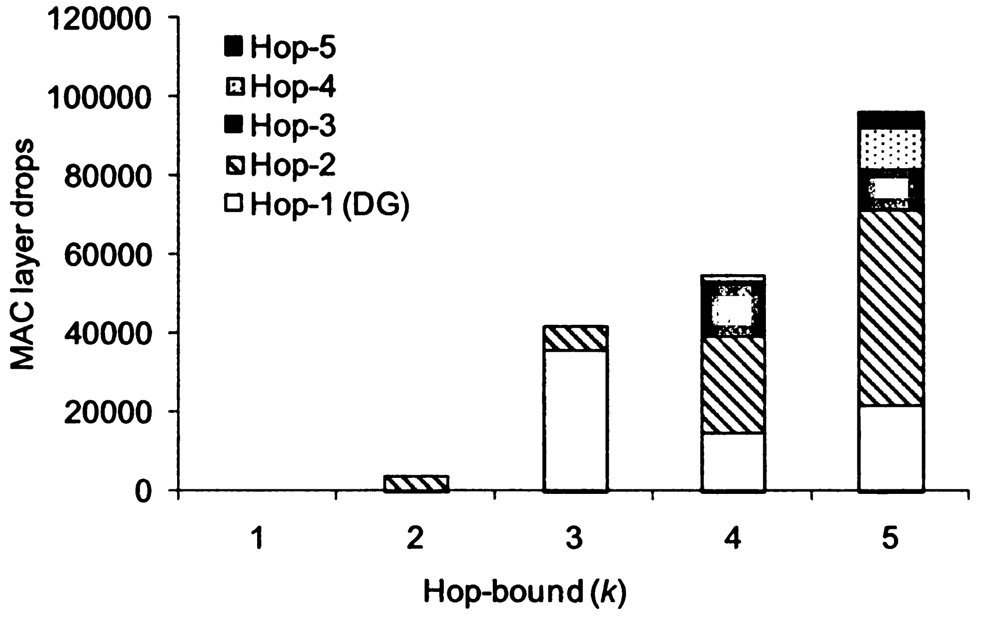


Figure 10-12: Drops experienced by nodes at different hop distance from the Navigation Agents

To understand the capacity issue further, we collected the packet delivery and drop statistics during the experiments. The packet delivery and MAC layer drop values obtained are shown in Figure 10-11 and Figure 10-12. As the data rate increases, the packet delivery is stable initially and then tends to drop. Higher the value of *k*, sooner the drops and higher the rate of packet drop. Since the MDH stops to collect data from the DGs, the low packet delivery ratio seen in Figure 10-11 at higher data rates is not intuitive. Figure 10-12 shows the drops seen at the 802.11 MAC layer at different distances from the *Navigation Agents* for λ=2048 bps. It can be observed that the number of drops is negligible for *k*=1. As *k* increases, nodes at higher and higher hop distances from the *Navigation Agents* experience packet loss. This is due to increased congestion at the MAC layer indicating that link capacity is insufficient to handle all the data generated in the network for the given values of λ and *k*.

10.5.2 Impact of Network Parameters on the Feasibility Range and Stop Threshold

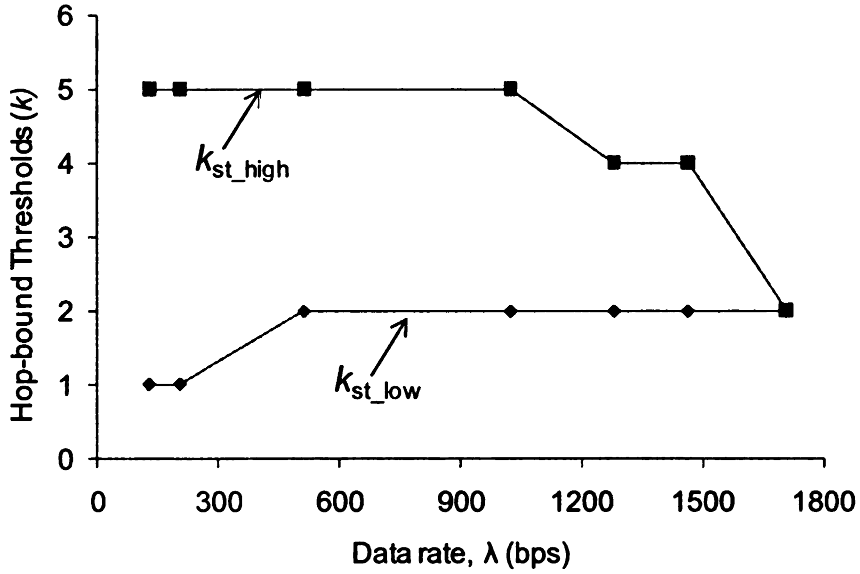
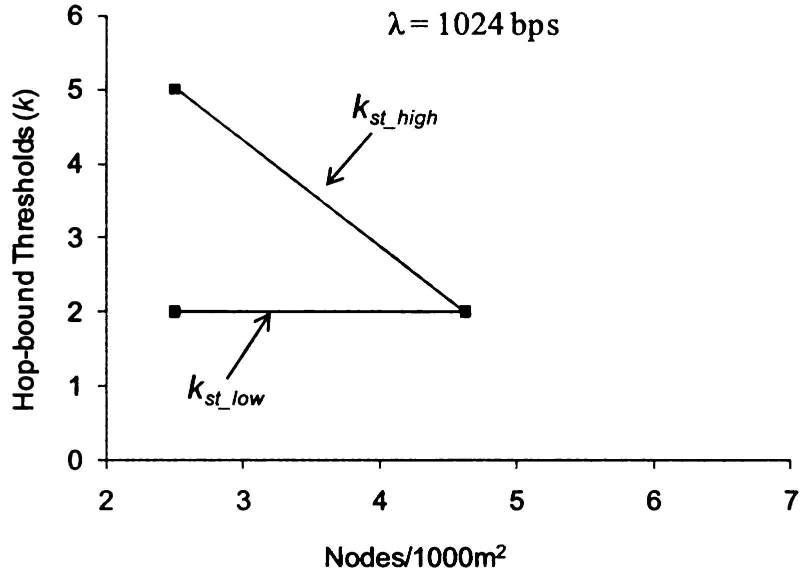


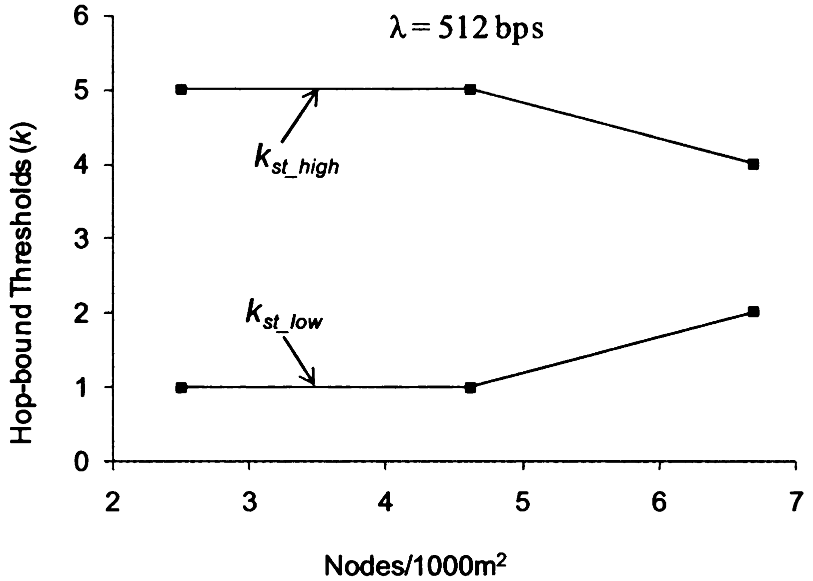
Figure 10-13: Impact of data rate on the feasibility range

Now, we explore the impact of data generation rate on the feasibility range for the hop-bound factor through simulation. As observed in Section 10.5.2, while the stability threshold k_{st_low} goes up, k_{st_high} is expected to go down resulting in progressively reducing the feasibility range. The results obtained from the simulation validate this trend as shown in Figure 10-13. The k_{st_high} drops with increasing data rate as the upload time increases causing t_{coll} to increase at lower values of k as seen in Figure 10-9. k_{st_low} goes up with increasing data rate also due to the same effect namely increase in upload time causing the lowest value that results in $t_{coll} < \lambda$ to increase. The feasible range shrinks as a result.

We conducted some experiments to study the impact of node density on the feasibility range of *k*. For the experiments we used the number of nodes in 240x240 meter area to be 144, 265 and 386. Figure 10-14 shows the feasibility range of *k* for the different densities (expressed in number of nodes per 1000m^2) for λ=1024 and 512 bps. In both cases, it is clear that the feasibility range shrinks with increasing density. With higher data rates, the rate at which the range shrinks is higher. For example, when λ=1024 bps, *k*=2 is the only feasible value for 265 node network. For λ=512 bps on the other hand, the feasible range is $1 \leq k \leq 5$. For 386 node network, when λ=1024 bps there is no value of *k* that is feasible. In other words, even single hop data collection ($k = 1$) is unable to meet the $t_{coll} \leq \tau$ constraint. On the other hand, for the same network when λ=512 bps, the feasibility range is $2 \leq k \leq 4$. These results from simulation validate the impact of density forecast by the model in Section 10.4.2.



(a)



(b)

Figure 10-14: Impact of density on the feasibility range for different data rates

Finally, we conducted experiments with varying MDH speeds from 1m/s to 20m/s. The data rate in this experiment was set to 1024 bps. The results obtained for the feasibility range shown in Figure 10-15. With increasing speed, the feasibility range expands as expected from the analytical model (see Figure 10-8). As the speed increases, the trip collection-time decreases allowing lower values of k to result in trip collection-time to be less than τ. This leads to the drop in k_{st_low}. In this experiment, the k_{st_high} stays at the highest possible value (k=5) for all values of the speed.

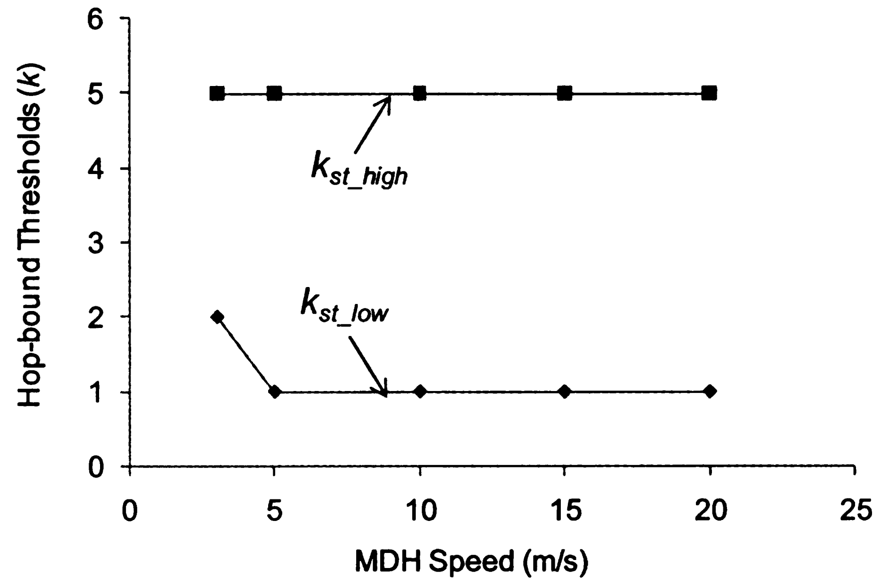


Figure 10-15: Impact of sink speed on the feasibility range

10.6 Summary and Conclusions

In this chapter, we have extended the generalized hop-bound model to accommodate the MDH-DG link capacity constraints. Upload time and trip collection-time equations were modified appropriately. A new stability threshold referred to as k_{st_high} was identified which provides the upper bound for the feasible range of k. The impact of capacity constraints on trip collection-time and the thresholds were numerically evaluated. The trends obtained from the analysis were validated using simulation experiments.

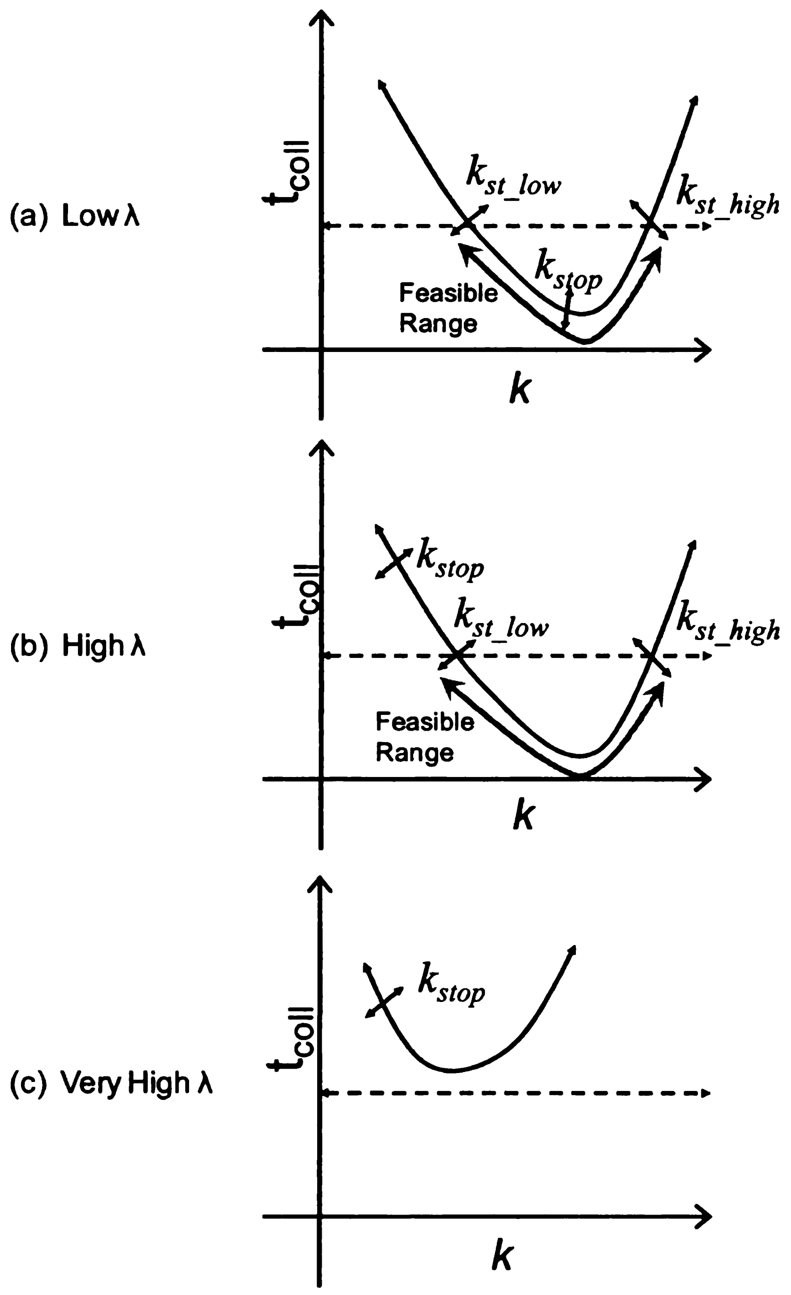


Figure 10-16: Possible feasible range scenarios

The analysis and experimental results presented in this chapter underscore the importance of taking the data generation rate (λ) into account for trajectory planning. When λ is small relative to the MDH-DG link capacity, sink trajectory can be based on hop-bound factor within the feasible range spanning $k_{stability}$ (or k_{st_low}) and $k_{critical}$. However, for larger λ, beyond the stop threshold, the trip collection-time (t_{coll}) is pushed by increasing upload time. If the data rate is large enough, t_{coll} can exceed the collection cycle duration for some $k < k_{critical}$ referred to as k_{st_high}. Thus, the feasible range is constrained within k_{st_high} implying that static sink data collection is not always feasible. Figure 10-16 shows the possibilities for the feasible range obtained for varying data rates. As long as k_{st_low} is non-zero (or t_{coll} crosses τ), irrespective of k_{stop}, the feasible range is k_{st_low} to k_{st_high}. However, in very high data rate case where the trip collection-time is driven up by increasing upload time before the trip collection-time reaches τ, there is no value of k for stable operation. Thus, the data generation rate of the nodes is a key parameter in determining the feasible region of the hop-bound factor that can be used for trajectory determination.

Chapter 11. Rate Aware Sink Trajectory Design for Heterogeneous Networks

11.1 Motivation

In Chapters 9 and 10, model and simulation results have established that the data generation rate of the nodes have a significant impact on the hop-bound thresholds and therefore, on the feasible range of *k* for stable operation. Thus far, we have assumed uniform data generation rate for all nodes. However, it is not difficult to foresee applications where the data generation rates are different in different regions of the network. For instance, in multi-mission sensor networks, data generation rates are dictated by the requirements of multiple applications or missions, and hence may not be uniform across different areas. In addition, concurrent applications [65, 66], differential aggregation in various parts of the network [151], and variations in sensing granularity depending on the quality of data required [152] could result in spatially varying data rates across the network. In such networks, a single value of *k* may not be able to balance the trip collection-time and energy consumption or lifetime, as different regions may have a different feasible ranges of *k*. Consequently, strategies for finding a sink trajectory with varying effective *k* based on heterogeneous λ need to be investigated.

Consider the example network shown in Figure 11-1. It has six regions, each with a different data rate. Let λ^{rx} denote the data rate of nodes in region-x. In the example network, assume $\lambda^{r1} > \lambda^{r3}$. If a single value of *k* is chosen for the entire network, then the MDH trajectory is evenly distributed among the regions. However, it seems intuitive that in areas where there is more data due to higher data rate such as λ^{r1}, the sink should visit more DGs to collect data to distribute the energy load. This implies that a lower amount of multi-hop routing (k) should be used for regions with higher data rates and vice versa.

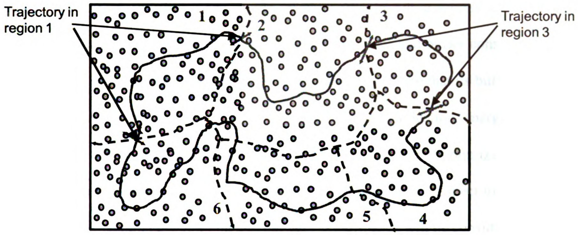


Figure 11-1: Network with six regions

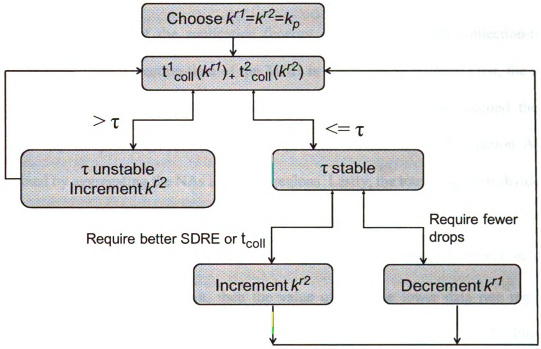
11.2 Related Work

In the existing literature as discussed in Section 2.4.2, [7] considered data collection using a mobile element from a set of Designated Gateways. However, the trajectory of the MDH was fixed and thus the nodes that came in direct contact with the MDH performed the role of DGs. Based on a specified trip collection-time constraint, the authors proposed to adapt the speed of the MDH based on the amount of data currently buffered at the DGs. The basic idea is to allow a subset of nodes with lot of buffered data to have longer contact time with the MDH during each trip. The goal is to maximize the quantity of data collected from each node. In this case, since the extent of routing is always constant, the amount of buffer required was higher and inversely proportional to the speed of the mobile element. In [7], the only factor that affected the packet delivery performance was MDH speed since the MDH path was fixed. The goal of this chapter is to study the inter-play between the trajectory based on the hop bound factor k (i.e. the extent of multi-hop routing), the heterogeneous data rate and the packet delivery performance.

The authors of [97, 98] consider the problem of scheduling mobile elements for single hop data collection to meet the data delivery deadlines and buffer constraints. The MDH has to visit each node with the frequency required to ensure that its buffer does not overflow. Since the problem was limited to single hop data collection, energy spent per packet or trip collection-time was not considered for the purpose of optimization. In this thesis, the hop bound factor k optimized for different regions with respect to overall trip collection-time, energy consumption and delivery performance will be considered. To our knowledge, optimization of the trajectory for heterogeneous network and traffic model in the context of multi-hop routing as proposed in this thesis has not been considered before.

11.3 Analysis of Region-specific Multi-hop Routing

In Section 9.2, the stop and the stability thresholds for k were discussed. These thresholds determine the feasible operating region for k and they depend on the data generation rate. Note that the trajectory length and AEPP are independent of λ, as derived in Sections 6.6.1 and 6.6.3 respectively. The upload time T_u and hence the trip collection-time t_{coll} depend on λ. For a given set of network parameters, as the data generation rate increases, the load on the DG-rooted routing trees increases. This also implies that the MDH would need to collect higher amount of data in the vicinity of an NA. As before, this results in the thresholds being moved to the left. In addition, the increased load on the different annular regions of the tree (see Figure 6-8) leads to higher variation in energy drainage rates around the DG resulting in increased SDRE. Increased SDRE for a given AEPP (only depends on k) would be expected to affect lifetime adversely. This implies that in heterogeneous networks, a single value of k for the entire network would not suffice. In order to achieve global objectives such as stable data collection, better trip collection-time and distributed energy drainage, different amount of multi-hop routing (k) would be required in different regions of the network.



Choose $k^{r1}=k^{r2}=k_p$
$t^1_{coll}(k^{r1}), t^2_{coll}(k^{r2})$
$> \tau$
$<= \tau$
τ unstable Increment k^{r2}
τ stable
Require better SDRE or t_{coll}
Require fewer drops
Increment k^{r2}
Decrement k^{r1}

Figure 11-2: Data-rate aware regional k selection procedure

In the following sub-sections, we present a specific example to outline the problem with using single value of k for the entire network regardless of data rate variations. The process of choosing different values of k for regions and the potential benefits are discussed. We consider an example where a network consists of two regions with data rates λ^{r1} and λ^{r2} respectively such that $\lambda^{r1} > \lambda^{r2}$. Figure 11-2 summarizes the process of choosing the extent of multi-hop routing for the regions using the example network. The k values chosen for region-1and region-2 are denoted by k^{r1} and k^{r2}. We conducted experiments with $\lambda^{r1} = 1024$ and $\lambda^{r2} = 205$ bps with the duration of the collection cycle τ set to 400s. Applying the insight that higher data generation rate requires a lower k, the k pairs were chosen such that $k^{r1} \le k^{r2}$. Figure 11-3 thru Figure 11-5 that shows the performance metrics for different regional k pairs.

To begin with a single value of k for the network, say k_p is chosen based on the primary requirement of the application (energy consumption or trip collection-time). The combined trip collection-time of the MDH is computed as follows. First, the set of Navigation Agents is identified in each region for its respective k value. Second, the tour distance is computed by solving the TSP problem on the set of Navigation Agents obtained by aggregating the NAs across all regions. Lastly, the tour distance is divided by the speed of the MDH to obtain the trip collection-time.

There are two possibilities based on whether t_{coll} satisfies the collection cycle constraint or not. If it does not, then the value of k in the lower data rate region is incremented and the process of computing the trip collection time and checking the stability condition is repeated (Section 11.3.1).

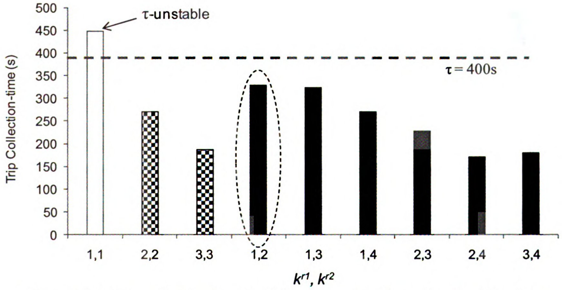


Figure 11-3: Trip collection-time for different combinations of regional k values

If the trip collection-time is within the specified collection cycle duration, then the application requirement determines the next action. If the application requires better energy load distribution or lower trip collection-time (Section 11.3.2), then the *k* value for the lower data rate region, namely region-2 is incremented and the stability check is repeated. On the other hand, if the application requires lower drops or data accumulation (Section 11.3.3), then the *k* value in the lower data rate region namely region-1 is decremented.

11.3.1 Unstable Operation ($t_{coll} > \tau$)

In heterogeneous networks, if a single value of *k* is chosen for the whole network, then it is required that the *k* be chosen to meet the primary needs of the application.

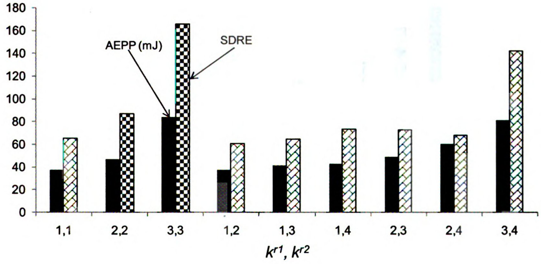


Figure 11-4: Energy metrics for different combinations of regional *k* values

If an application requires that the network be operational for a long time and desires to choose a *k* value for both regions, then $k^{r1} = k^{r2} = 1$ is the obvious choice. However, that results in a trajectory that is unstable as seen in Figure 11-3 because $t_{coll} > \tau$. Alternatively, choosing $k^{r1} = k^{r2} = 2$ results in a stable trajectory, but it is not desirable due to poor energy and drop performance as seen in Figure 11-4 and Figure 11-5. If k were to be chosen on a regional basis, then a higher value of k, say 2 can be assigned to region-2 with lower data. The resulting combination $k^{r1} = 1$ and $k^{r2} = 2$ leads to a trajectory that can satisfy the collection cycle constraint and yet achieve better energy and drop performance compared to $k^{r1} = k^{r2} = 2$.

11.3.2 Stable Operation ($t_{coll} > \tau$) with High Energy Expenditure

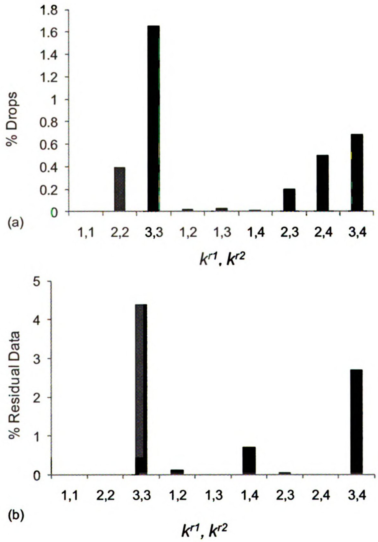


Figure 11-5: Drops and data accumulation for different combinations of regional k values

An application requiring to operate at the lowest k that results in a stable trajectory will need to choose $k^{r1} = k^{r2} = 2$. Although the resulting trajectory is stable, it suffers from high energy expenditure and poor drop performance as seen in Figure 11-4 and Figure 11-5. If k were to be chosen on a regional basis, then a lower value of k, say 1 can be assigned to region-1 with higher data rate to improve the energy and drop metrics. The resulting combination $k^{r1} = 1$ and $k^{r2} = 2$ leads to a trajectory that can satisfy the collection cycle constraint and yet achieve better energy and drop performance compared to $k^{r1} = k^{r2} = 2$.

11.3.3 Stable Operation ($t_{coll} > \tau$) with High Packet Accumulation and Drops

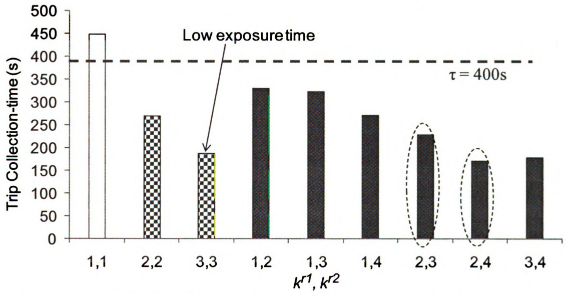


Figure 11-6: Trip collection-time for different combinations of regional k values

Now, consider an application whose prime requirement is low exposure time. In such a case, $k^{r1} = k^{r2} = 3$ which results in the lowest trip collection-time (see Figure 11-6) among the network-wide k assignments, is the logical choice. However, the resulting trajectory suffers from higher consumption energy consumption, increased drops and data accumulation at the DGs as seen in Figure 11-4 and Figure 11-5. In this context, choosing a region specific k obtained by reducing the value of k in region-1 which has higher data rate can address the energy and drop issues. For example, the combinations $k^{r1}=2$ and $k^{r2}=3$ or $k^{r1}=2$ and $k^{r2}=4$ offer exposure time comparable to $k^{r1}=k^{r2}=3$ (see Figure 11-6) and at the same time offer better performance in terms of energy per packet, variation in energy usage, packet drops and data accumulation (see Figure 11-4 and Figure 11-5).

11.4 Problem Formulation

Consider a network with m regions such that a region i has a data rate of λ^{ri} with an MDH that collects data in *Collect-on-Move* mode. The capacity of the MDH-DG link (C), the speed of the MDH (v) and the duration of the collection cycle (τ) are given. The problem is to find a set of Navigation Agents such that the resulting MDH trajectory satisfies the following:

1) The network operates within the stop-threshold (k_{stop})
2) The trip collection-time (t_{coll}) is minimized.

Note that constraint 1 inherently ensures that the extent of multi-hop routing in areas with higher data rates is lower and vice versa. Once the Navigation Agents are identified, data collection proceeds as described in Chapter 8. The trajectory resulting from the selected NAs has to be verified for compliance with the collection cycle (τ) constraint.

11.5 λ-Aware Trajectory Determination (LATD)

In this section, we develop a λ*-aware Trajectory Determination* or LATD algorithm based on the NADC framework presented in Section 7.2. The design goal of LATD is to select the Navigation Agents such that data collection can be completed without requiring the MDH to stop. This implies that the NA selection process has to ensure that the data buffered at the DGs can be collected within the contact duration between the MDH and the DGs around a Navigation Agent. The NA determination algorithm (phase-1 of NADC algorithm) described in Section 7.2.1 was data rate and link capacity independent in its operation. In this section, extensions to the NA determination algorithm required to achieve data rate awareness are presented. Rest of the NADC components can be used without any modifications.

11.5.1 Protocol Details

The Navigation Agent selection algorithm runs in multiple phases:

11.5.1.1 Data Forwarding Tree

In order to choose NAs in a data rate aware manner, LATD requires that every node keep track of the amount of data that is routed through it to the MDH. For this purpose, a tree is created rooted at a node referred to as the *Lambda-Root*. The tree construction is accomplished using the neighbor discovery *Hello* messages. In the Hello message, every node sends its data accumulation information per unit time, □ which is computed as the sum of:

1) Data that is forwarded by its children in the tree per unit time

2) Data generated by itself per unit time (λ)

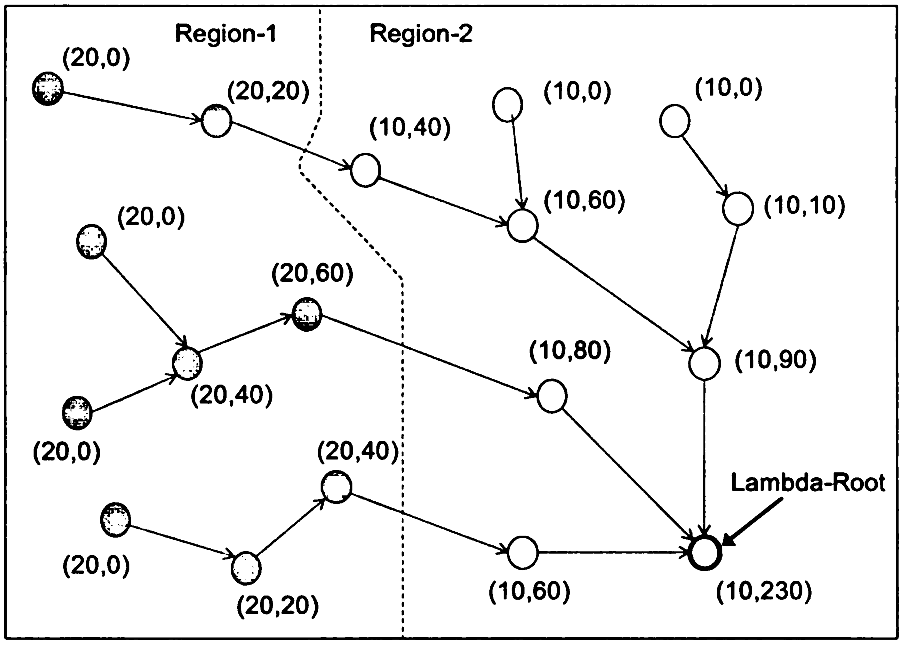


Figure 11-7: Example to illustrate the tree rooted at Lambda-Root

At the end of this phase, each node knows the amount of data that it has to upload to the MDH. This quantity is referred to as local data accumulation, □. Figure 11-7 shows an example network which consists of nodes that generate data at two different rates namely solid filled nodes generate data at 20 bps and clear filled nodes at 10 bps. The tuple next to each node is of the form (data generated per unit time by the node, data forwarded by the children per unit time) and the two values combined provide □.

11.5.1.2 Navigation Agent Selection

As mentioned in Section 7.2.1, the data rate unaware NA determination phase begins by establishing a tree at an arbitrarily chosen root node referred to as *NA-Root* in this section. In LATD, the *NA-Root* is chosen such that it is as far from the *Lambda-Root* as possible in terms of hop count as shown in Figure 11-9. The figure also shows the tree created for the example network. This ensures that the NA determination protocol messages flow towards the *Lambda-Root* and allows the protocol to estimate the local data accumulation at nodes to be as accurate as possible.

```
Algorithm for λ-Aware Navigation Agent Identification:

Process Declare-NA message from a neighbor:
Mark neighbor as a Navigation Agent in neighbor table
Set neighbor data-rate to zero
Broadcast Accept-NA message

Process Accept-NA message from a neighbor:
Call ComputeCumDataRate
// check if the amount of data accumulated is higher than upload capacity
if ( (cum-data-rate* $\tau$*$\gamma$) > (C * transmission-range / MDH-speed) ) {
  Mark myself as a Navigation Agent
  Set □to 0
  Broadcast Declare-NA message to all neighbors
} else {
   re-broadcast Accept-NA message
}

ComputeCumDataRate ():
 Set cum-data-rate to zero
// compute cum-data-rate as sum of data rate from all children
 For (each neighbor in Neighbor-List) {
    if (I am neighbor's parent ) {
          Add neighbor's □to cum-data-rate
    }
 }
 return (cum-data-rate)
```

Figure 11-8: Pseudo code for data rate aware NA selection

The root node initiates the NA discovery process by sending out an *Accept-NA* message. Every node in the network has an estimate of the amount of data that it can upload to the MDH when it comes in contact which is referred to $U_{\max}$. $U_{\max}$ is computed by multiplying the link capacity of the MDH-DG link with the amount of contact duration with the node. The contact distance can range from negligible to $2*r$ where r is the transmission range. Thus, on average a node can expect to have a contact time of $\frac{r}{v}$. So, $U_{\max} = \left(C * \frac{r}{v}\right)$. In the example network shown in Figure 11-9, the values of transmission range r, MDH-DG link capacity C and MDH speed v are assumed to be 1m, 40 kbps and 1m/s respectively resulting in:

$$U_{\max} = 40kbps * \frac{1m}{1m/s} = 40kbits.$$

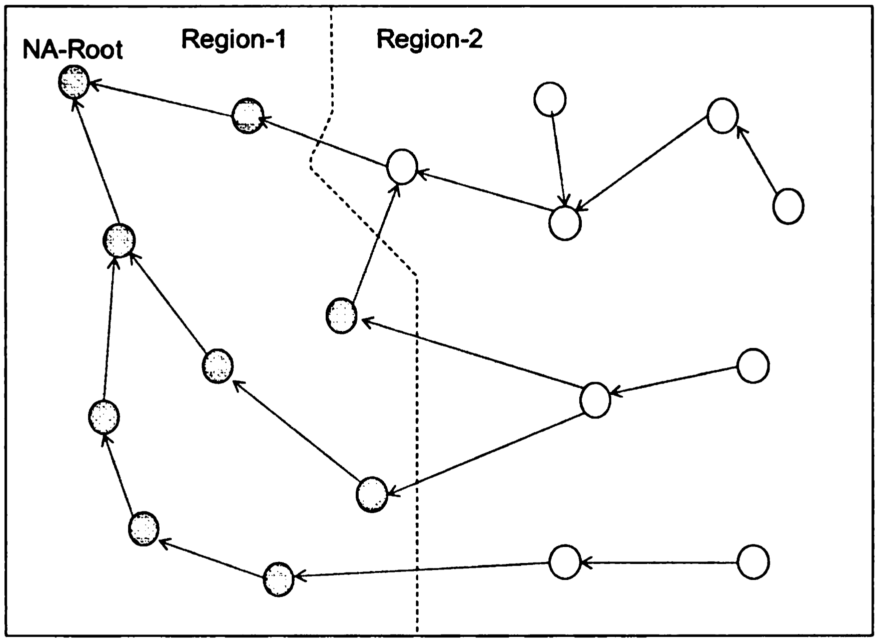


Figure 11-9: Tree rooted at NA-Root

Also, the local data accumulation per second, □ which was computed during the data forwarding tree creation process outlined in Section 11.5.1.1 is maintained at every node. The neighbors that receive the *Accept-NA* message compute the amount of data accumulated during a collection cycle as $\tau * \phi * \gamma$. γ is the scaling factor that is applied to the accumulated data to take into account the message overheads and the interference from other DGs in the neighborhood uploading their data at the same time to the MDH. γ is always greater than 1.

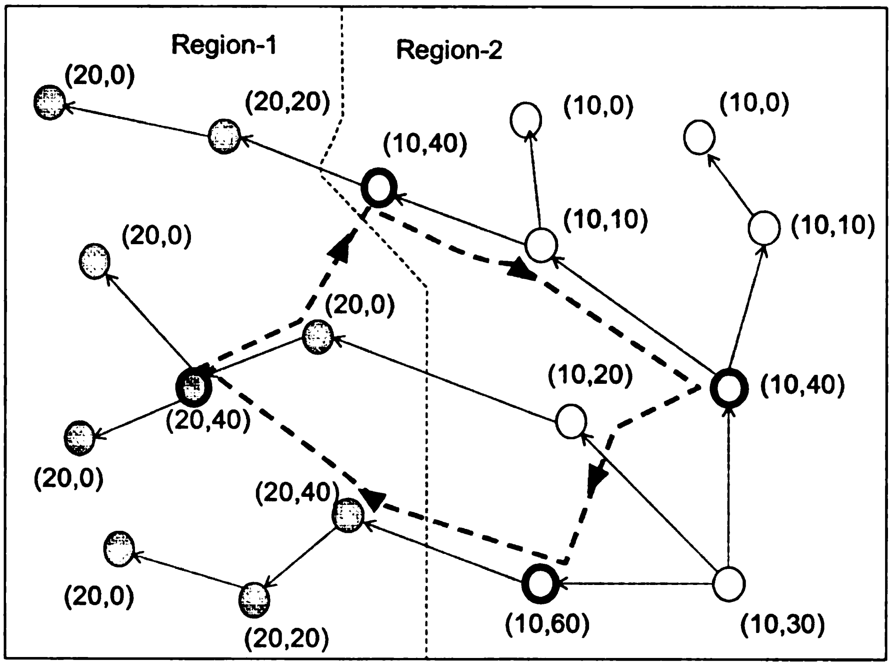


Figure 11-10: NA selection based on data accumulation and the resulting trajectory

If $\tau * \phi * \gamma \geq U_{\max}$, the node declares itself as a Navigation Agent by broadcasting a *Declare-NA* message. When nodes receive the Declare-NA message, the nodes that are have lower hop distance to the *NA-Root* ignore it. Nodes that have a higher distance to the *NA-Root* perform the following actions:

1) Mark the originator of the *Declare-NA* message as an NA node and set its □ value to zero.
2) It marks itself as a non-NA node broadcasts an *Accept-NA* message.

On the other hand, if $\tau * \phi * \gamma < U_{\max}$, the recipient node marks itself non-NA and broadcasts an *Accept-NA* message . This process continues until all nodes are marked as NA or non-NA nodes. Figure 11-8 presents the pseudo code for the LATD algorithm.

In the example shown in Figure 11-10, the total amount of data accumulated at the nodes is calculated as $\tau * \phi * \gamma$ as mentioned above. The value of γ was chosen to be 8. NA nodes are selected wherever the total amount of data accumulated exceeds $U_{\max}$ (calculated as 40kbits earlier). Based on that logic, nodes shown with dark circles in Figure 11-10 are chosen as the Navigation Agents and the resulting trajectory is indicated.

11.5.2 Protocol Behavior Analysis

The goal of LATD protocol is to identify NAs such that the resulting trajectory length is minimal and allow the MDH to collect all the data buffered at the DGs during a round trip. Therefore, the amount of residual data remaining at the DGs after contact with the MDH is negligible. LATD achieves this by identifying NAs along the tree rooted at *Lambda-Root* only where enough data has been accumulated to utilize the estimated contact duration efficiently. Therefore, the trees rooted at the DGs should tend to be longer in the lower data rate regions than in the higher data rate regions. Therefore, the hop distances of the nodes in heterogeneous regions would be expected to exhibit different profiles.

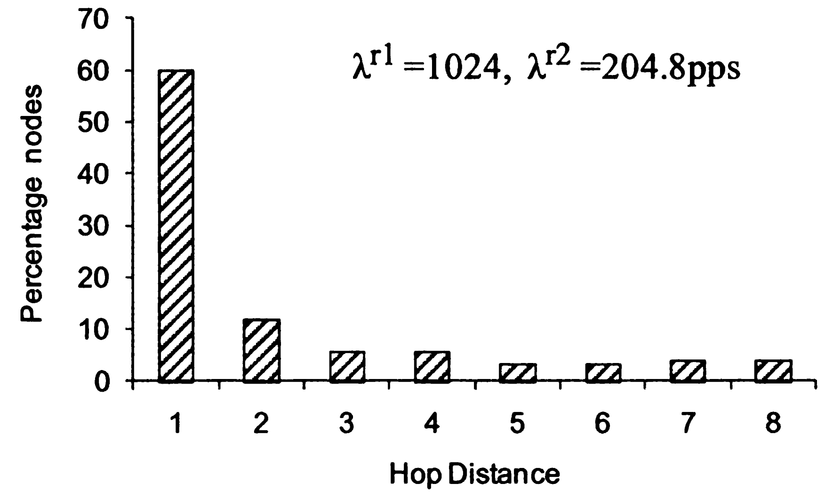


Figure 11-11: Distance of nodes from the MDH

We conducted data collection experiments with LATD as the trajectory determination mechanism on a two region network with nodal data rates of 1024 and 204.8 bps in the two regions respectively. The hop distance of every node to the MDH was collected. Figure 11-11 and Figure 11-12 show the hop distance information for the network as a whole and separately for the two regions. Results show that the region with the higher nodal data rate has lower hop counts. For example, in region-1 with 1024 bps, the hop distance of the nodes range from 1 to 4. In contrast, in region-2, the hop distance ranges from 1 to 8.

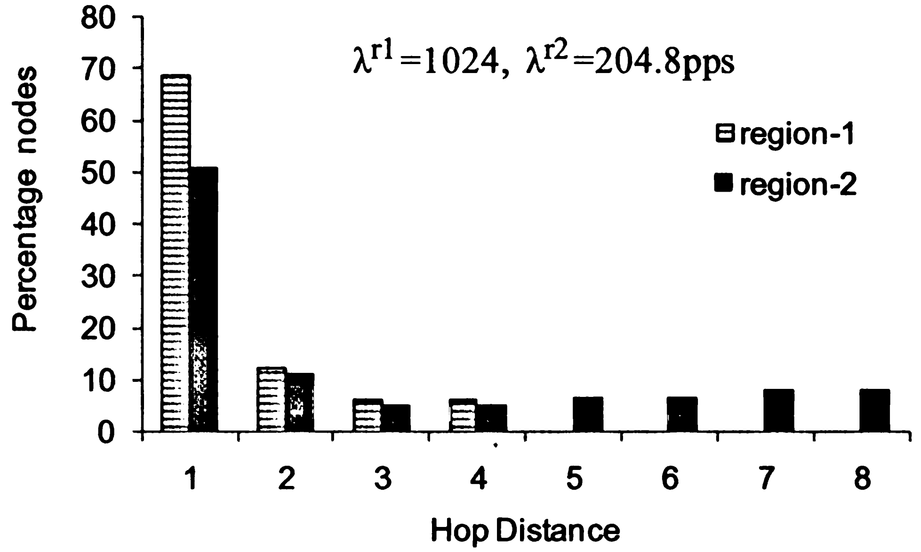


Figure 11-12: Regional breakdown of hop distances to the MDH

11.6 Simulation Performance Evaluation

LATD (λ-Aware Trajectory Determination) protocol as described in Section 11.5 was implemented in ns2 and experiments were conducted to evaluate its performance. The proposed LATD protocol was used in lieu of the Navigation Agent determination algorithm (described in Section 7.2) in NADC framework. The results obtained from the simulation experiments are presented in this section.

In Section 11.3, a detailed case was presented to choosing a k in different regions of the network based on data rates. The goal was to achieve data collection stability (from τ perspective) or better drop and residual data accumulation performance. In this sub-section, the trajectory obtained with the LATD algorithm is compared with the best trajectories identified using different regional k values namely $(k^{r1} = 1, k^{r2} = 2)$ for achieving stability, and $(k^{r1} = 2, k^{r2} = 3)$ or $(k^{r1} = 2,\ k^{r2} = 4)$ for drop and data accumulation performance.

11.6.1 Performance Comparison of LATD with Regional *k* Selection

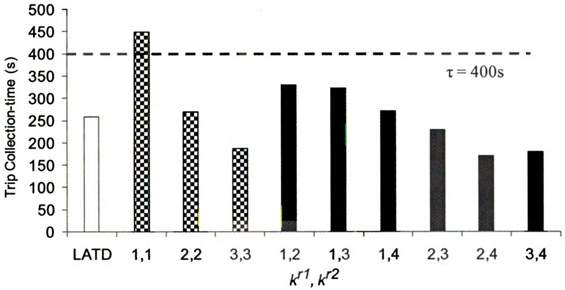
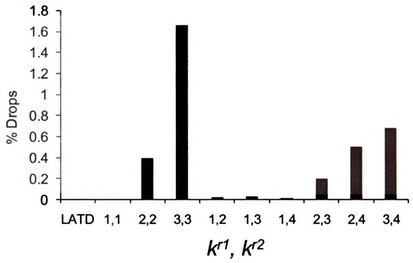


Figure 11-13: Comparison of LATD trip collection-time with regionally selected k trajectories



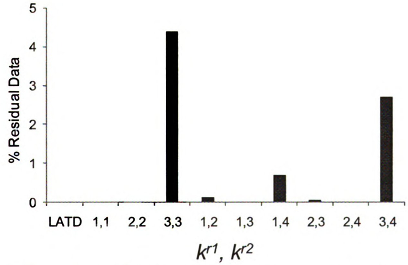


Figure 11-14: Comparison of LATD packet drop and accumulation characteristics with regionally selected *k* trajectories

The trajectory found by LATD results in trip collection-time that satisfies the collection cycle (τ) constraint similar to the regionally selected *k* values as seen in Figure 11-13. Also, Figure 11-14 shows that LATD trajectory has no residual data accumulation and drops indicating that it is operating within the *stop-threshold*. Close observation of results presented in Figure 11-13 thru Figure 11-15 show that the performance of LATD matches that of ($k^{r1} = 2$, $k^{r2} = 3$). In comparison to ($k^{r1} = 1$, $k^{r2} = 2$) and ($k^{r1} = 2$, $k^{r2} = 3$), the trajectory resulting from ($k^{r1} = 2$, $k^{r2} = 3$) represents the solution that minimizes the trip collection-time as much as possible while maintaining good drop and data accumulation performance (indicative of operation within the stop-threshold). The fact that the performance of LATD matches ($k^{r1} = 2, k^{r2} = 3$) proves that the design goals of LATD as outlined by the problem formulation in section 11.4 are being achieved by the distributed LATD algorithm.

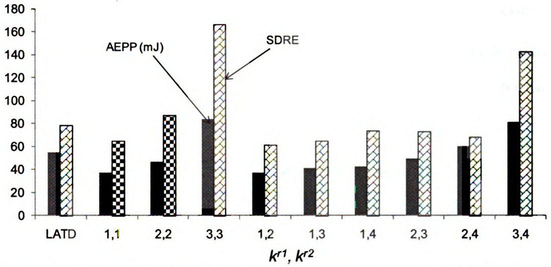


Figure 11-15: Comparison of LATD energy metrics with regionally selected *k* trajectories

11.6.2 Performance Comparison of LATD with Unconstrained NA Selection

In this sub-section, the following scenarios in networks with different data rates are compared:

a) Unconstrained NA selection scenario where a single value of *k* chosen for the whole network (Section 7.2).

b) λ-aware Trajectory Determination algorithm (Section 11.5)

11.6.2.1 Two-region Network Scenarios

Simulation experiments were conducted on a network with two regions each consisting of 64 nodes. An MDH moves continuously at a constant speed of 3m/s and collects data.

The collection cycle duration (τ) was set to 400s. Also, the scaling factor, γ discussed in Section 11.5.1.2 was set to 8. For the single *k* scenario, the value of *k* assigned for the two regions was chosen such that the following conditions are met:

1) The resulting trajectory is stable i.e. $t_{coll} \leq \tau$.
2) The drop performance for data collection along the resulting trajectory is acceptable. The threshold for acceptability was defined to be 2% in this case.

The nodal data rates for the two regions, λ^{r1} and λ^{r2} were set to three values namely (512,102), (1024, 205) and (1280, 512) bps respectively. These data rate combinations are presented as they essentially summarize the working of the LATD algorithm.

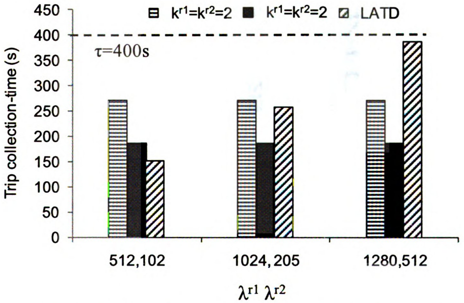
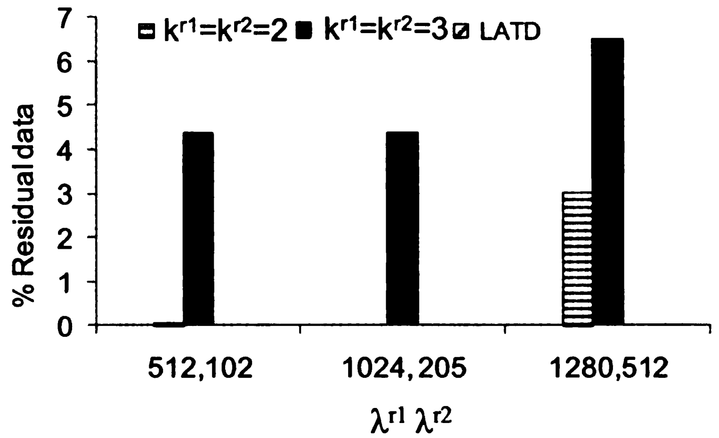


Figure 11-16: Trip collection-time for different regional data rates

Figure 11-16 shows the trip collection-time obtained for the experiments. In the (512,102) case, the $k^{r1} = k^{r2} = 2$ solution results in a trajectory that has a relatively high t_{coll} value, but no residual data accumulation and negligible drops (0.2%). On the other hand, $k^{r1} = k^{r2} = 3$ results in a shorter trajectory, but with 4% residual data accumulation at the DGs and relatively high drops (0.6%). In other words, $k^{r1} = k^{r2} = 3$ is operating beyond the *stop-threshold.*



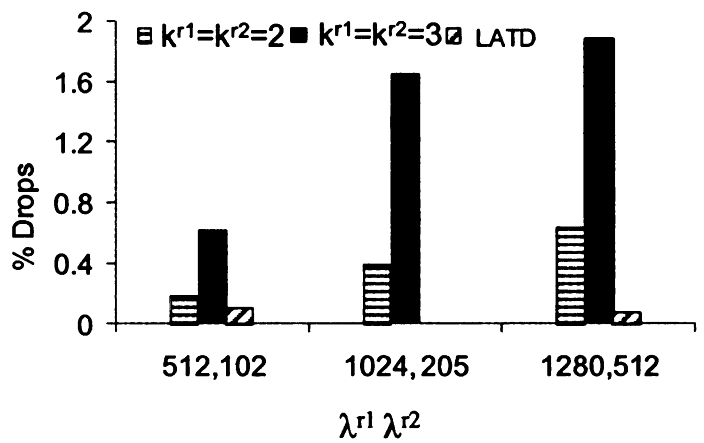
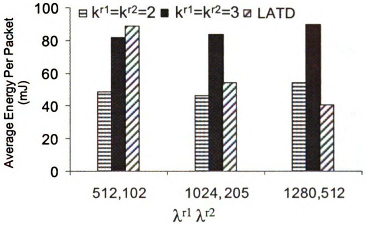


Figure 11-17: Residual data accumulation and drop performance for different regional data rates

In contrast, the trajectory determined by LATD has a lower trip collection-time while achieving no residual data accumulation and low drops (0.1%). This is a direct result of choosing the Navigation Agents only when there is enough data to utilize the contact time with the MDH. This is brought out by the fact that the average hop count for LATD is 2.55 and the maximum hop count is 7 in comparison to an average hop count of 1.41 for $k^{r1} = k^{r2} = 2$ and 1.66 for $k^{r1} = k^{r2} = 3$. A side-effect of increase in the average hop count is a corresponding increase in the energy per packet values as seen in Figure 11-18.

Thus, LATD effectively shrinks the trajectory in the low data rate region reducing the overall trip collection-time and at the same time ensuring that it operates within the stop-threshold.



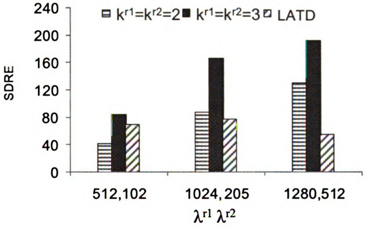


Figure 11-18: Energy metrics for different regional data rates

In the (1024, 205) bps case, the $k^{r1} = k^{r2} = 2$ trajectory is the same as the one for (512,102) bps case as it uses data rate unaware NA selection. It is able to avoid residual data accumulation and has slightly higher drops (0.4%). On the other hand, $k^{r1} = k^{r2} = 3$ which results in a shorter trajectory continues to have high residual data accumulation at the DGs and high drops (1.6%). Since this trajectory was beyond stop threshold for a lower data rate combination of (512,102) bps, for the higher data rate,

$k^{r1} = k^{r2} = 3$ is also operating beyond the *stop-threshold*. In this case, the trajectory determined by LATD has a trip collection-time comparable to $k^{r1} = k^{r2} = 2$ while achieving no residual data accumulation and no drops. So, LATD is unable to shrink the trajectory as much as in the lower data rate case of (512,102) bps. In effect, LATD shrinks the trajectory slightly but ensures operation within the stop-threshold.

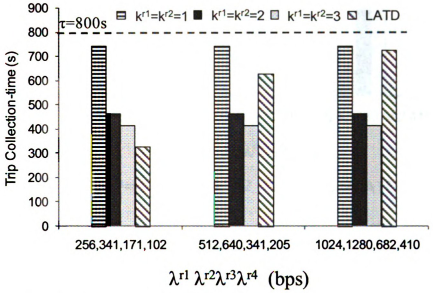
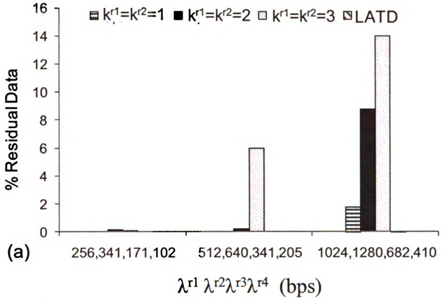


Figure 11-19: Impact of regional data rates on trip collection-time

In the (1280, 512) bps or high data rate case, the $k^{r1} = k^{r2} = 2$ and $k^{r1} = k^{r2} = 3$ trajectories result in high residual data accumulation and drops suggesting that they are beyond stop threshold. In this case, LATD trajectory is, in fact longer that both $k^{r1} = k^{r2} = 2$ and $k^{r1} = k^{r2} = 3$. However, there is no residual data accumulation along the trajectory and the drops are negligible. Thus, LATD, for high data rate scenario actually elongates the trajectory to ensure that it operates within the *stop-threshold*. This is borne out by the higher trip collection-time and the average hop count numbers. The average hop count for $k^{r1} = k^{r2} = 2$ and $k^{r1} = k^{r2} = 3$ is 1.39 and 1.6 respectively while LATD has 1.19. Note that in this case, the energy metrics AEPP and SDRE also benefit due to the longer trajectory and reduced hop counts as evidenced in Figure 11-18.



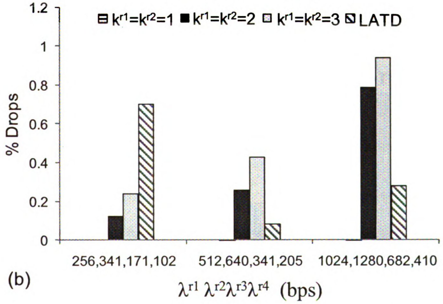
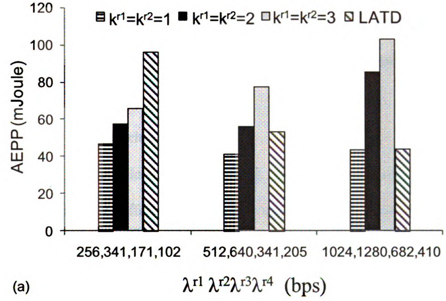


Figure 11-20: Data accumulation and drops for varying regional data rates

Thus, based on the above cases, it is possible to conclude that LATD adapts appropriately to the data rates of the regions and finds a trajectory that is able to operate within the stop-threshold. In other words, the trajectory allows the MDH to collect data without stopping.

11.6.2.2 Four-region Network Scenarios



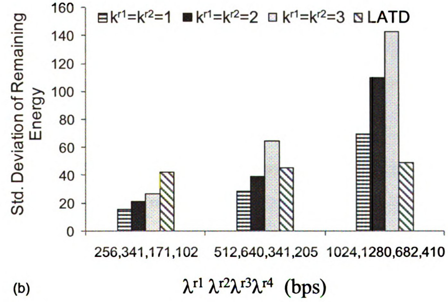


Figure 11-21: Energy metrics for varying regional data rates

Simulation experiments were conducted with a four region network with regional data rates of (256,341,171,102), (512,640,341,205) and (1024,1280,682,410) respectively.

The duration of the collection cycle, τ was set to 800s. For the single k scenario, the value of k assigned for the four regions was chosen based on acceptable drop rates as mentioned earlier for the two region cases. The single k values of 1, 2 and 3 were considered. The results obtained are presented in Figure 11-19 thru Figure 11-21. Results show that LATD works in the same manner as observed in Section 11.6.2.1 and adapts the trajectory to the regional data rates to ensure operation within the stop-threshold.

11.7 Summary and Conclusions

In this Chapter, the problem of determining the extent of multi-hop routing for networks with regionally differing data rates was considered. Potential problems of using a single k value or data rate unaware trajectory determination were investigated. Benefits of choosing the degree of multi-hop routing in a region specific manner were outlined. Applying the insights gained from the stability analysis in Chapters 10 and 11, LATD (λ-Aware Trajectory Determination) extensions were developed for the distributed data collection framework, NADC proposed in Chapter 7. Performance of LATD was evaluated using simulations and compared with the data rate unaware solution.

The analysis and the experimental results obtained in this chapter prove that there are numerous benefits to choosing region specific multi-hop routing in heterogeneous data rate networks. Performance benefits include stable operation, lower trip collection-time, better energy and drop performance depending on the regional data rates. Results show that the proposed distributed mechanism called LATD can effectively find a trajectory that ensures that the data collection occurs within the stop-threshold and at the same time minimize the trip collection-time. This is accomplished by identifying Navigation Agents such that enough data has been buffered to use the contact time with the MDH as efficiently as possible. In essence, LATD adapts the length of the trajectory based on the data rates in the network. In higher data rate situations, the trajectory is elongated to operate within stop-threshold and vice versa. In this chapter, the performance evaluation of LATD shows that a distributed trajectory determination can successfully adapt to the data rates and identify a trajectory that has good performance characteristics. Thus, the work thus far provides proof of concept for distributed data rate aware trajectory planning.

Ongoing work on the LATD algorithm has two main focal points. First, fine-tuning the parameter γ and investigating mechanisms to adapt it in real-time. Second, choosing the NAs such that MAC layer contention is minimized to further improve the data collection performance.

Chapter 12. Thesis Contributions and Future Work

12.1 Contributions

In this thesis, the role of controlled node mobility for enhancing data collection and energy performance in wireless sensor networks was investigated. Joint mobility and packet routing protocols were developed and the associated tradeoffs were analyzed from a network design standpoint. Mobility algorithms and protocols were developed for two types of deployments namely mobile sensor network with a static sink and static sensor network with a mobile sink.

In Chapter 4, this thesis explores a paradigm for extending network life by introducing energy-aware node mobility in wireless sensor networks. The concept of controlled mobility for energy saving has been motivated by a natural grouping behavior that is observed in Emperor penguins in the Antarctic region. In this thesis, a parallel was drawn between the heat loss of a penguin at the group periphery, and the routing energy burden of sensor nodes near the aggregating base stations in a sensor network. Then a fully distributed and localized mobility control algorithm, and packet routing protocols were developed for collective extension of the network life. Experimental results presented in Chapter 5 demonstrate that the proposed controlled mobility framework can significantly extend a sensor network's operating life even in situations where the energy cost of physical node movement is modeled as high as up to four orders of magnitude larger than the energy cost for packet transmission.

From Chapter 6 onwards this thesis focuses on deployments with static sensors and a mobile sink that collects data generated in the network. In Chapter 6, a generalized framework based on hop-bound factor *k* to achieve a desired balance between trip collection-time and energy consumption was proposed. This framework provides a joint optimization of MDH trajectory and routing in the network. The idea is to choose a set of DGs such that all nodes can route their data to one of the DGs within k hops. The value of k has to be chosen in accordance with the application trip collection-time and lifetime requirements. In addition, a model was developed for the generalized framework and analyzed. The analysis showed that hop-bound k can be used as a knob to balance the trip collection-time and energy consumption.

The stability analysis presented in Chapter 9 revealed that for given network and MDH parameters such as data generation rate λ, MDH speed v, collection cycle τ, link capacity C, there exist hop-bound thresholds that define the feasible range of k for data collection. Feasible ranges were identified for sinks that can stop (*Stop-on-Demand* mode) and cannot stop (*Collect-on-Move* mode). For both modes, the stability threshold, k_{st_low} dictates the lower bound of k below which the trip collection-time t_{coll} would be greater than τ making the system unstable. The stop threshold, k_{stop} on the other hand determines the point beyond which the MDH has to incorporate stops into its trajectory to complete data collection from the DGs. Therefore, k_{stop} indicates the upper-bound for *Collect-on-Move* data collection. This threshold impacts the choice of sink technology as pointed out in Section 6.6.2.

In Chapter 9, stability analysis was performed assuming that the nodal data rates were significantly lower than the MDH-DG links. In Chapter 10, the low data rate assumption was relaxed. The stability analysis was repeated with taking into account the MDH-DG link capacity and the impact of interference of the transmissions around the

MDH. The analysis showed that a stability-ceiling, k_{st_high} exists which upper-bounds the feasible region of *k*.

The energy-trip collection-time tradeoff and the thresholds predicted by the model were experimentally validated by developing a Network-Assisted Data Collection (NADC) as described in Chapters 7 and 8, which is a distributed and location-unaware sink navigation and data routing protocol. Finally, the NADC framework was extended for networks with spatially heterogeneous data generation rates in Chapter 11, and the routing feasibility results were validated for such scenarios.

12.2 Future Work

The work undertaken during this thesis has opened up a number of new avenues for future research. In the context of data gathering, the trajectory of the sink impacts the trip time and extent of routing required for packet delivery which in turn affect performance metrics such as packet delay, delivery rate, energy efficiency and network lifetime.

1) Multiple Mobile Sinks

One way to increase the feasibility range for multi-hop routing (*k*) is to employ multiple mobile sinks. This raises the problem of trajectory planning for multiple sinks such that they collect data from the network in a load-balanced way. Thus far, in the literature, sink trajectories [112, 153] are pre-programmed and thus are unable to adapt to dynamic changes in the network. Alternatively, the mobile sink randomly determines the next location to move to in real-time based on local information which has the disadvantage of sub-optimal decisions and unbounded delays. Distributed network assisted trajectory planning approach has the distinct advantage of being able to adapt to topological dynamics, changes in data generation profile or altered mission goals and offer a bounded delay. A number of design extensions would need to be developed for NADC (Network Assisted Data Collection) to handle multiple sinks. Trajectory planning solutions will be required for different scenarios emerging from the connectivity that is assumed among the sinks themselves.

2) Under Water Sensor Network Environments

Recently sensor networks deployed in underwater environments [154-156] have started to receive a lot of attention for oceanographic, assisted navigation, pollution monitoring and such other applications. Like the terrestrial networks, underwater sensors are also energy constrained. In addition, these networks face a number of challenges including low data rate, long propagation delays, unidirectional links, unpredictable link characteristics and constant mobility even of anchored nodes which float due to the currents. Trajectory planning for mobile sinks in such networks is a complex task. Centralized approaches to trajectory computation are not readily applicable here due to the dynamic environment. Distributed mechanisms have an inherent advantage since they can adapt to changing conditions. We plan to apply the insights gained from the thesis to sink path planning problem in underwater networks.

APPENDIX A: Numerical Analysis of Feasibility Range for Full Coverage Scenario

In Chapters 6 and 10, analysis was performed using $\delta_S = 2kr$ which is the separation between the circular sub-trajectories. This separation distance was chosen to ensure no overlaps in coverage for ensuring tractability of energy analysis. However, the analysis holds for any value of δ_S. In this section, we present the results obtained by repeating the analysis with $\delta_S = \sqrt{3}kr$ to ensure complete area coverage [77]. The equation modifications and the updated graphs are presented in this section.

A.1 Trajectory Length (L)

The radius of the outermost sub-trajectory is $R - \frac{\sqrt{3}kr}{2}$, and adjacent sub-trajectories are mutually separated by a distance of $\sqrt{3}kr$. Thus, the radius of the i^{th} sub-trajectory (starting from the periphery) = R-[$\frac{\sqrt{3}}{2}kr + (i-1)\sqrt{3}kr$] for $i = 1$ to $R/\sqrt{3}kr$, and the circumference is $2\pi[R - (kr(2i-1))]$. The total trajectory length L can be written as:

$$L = 2\pi \sum_{i=1}^{\lceil R/(kr\sqrt{3}) \rceil} \left[R - \left(kr\sqrt{3}(i - \tfrac{1}{2})\right)\right] + 2R_m$$

where R_m represents the transition distance (see Figure 6-5) from the outer-most to the inner-most sub-trajectory. The quantity $2R_m$ represents the distances traversed from one sub-trajectory to the next during a complete trip.

A.2 Update Time (T_u)

The contact distance with at least one DG in the vicinity of a *Navigation Agent* can range from *2r* (when a *Navigation Agent* and all its DGs are collocated) to $2\sqrt{3}r$ (when at least two DGs of a *Navigation Agent* are located on two diametrically opposite points on the circle of radius *r* around the *Navigation Agent*). With uniformly dense node distribution, the contact period can be assumed to be symmetric around the midpoint of the contact distance range (*2r* and $2\sqrt{3}r$) leading to an average of $(r+\sqrt{3}r)$. The data forwarded by the interfering nodes is essentially all the data generated in the network except the data generated by the DGs themselves. Thus, we have:

$$C_{in} = \left(r+\sqrt{3}r\right)\Big/_{v}\left[\pi(kr)^2\lambda d - \pi r^2\lambda d\right] = \left[2.73\times \pi r^3\lambda d\left(k^2-1\right)\right]\Big/_{v}$$

Substituting the value of C_{in}, the upload time is computed as:

$$T_u = \left(\tau\lambda\pi R^2 vd\right)\Big/\left(Cv - \left(2.73\times \pi r^3\lambda d(k^2-1)\right)\right)$$

A.3 Numerical Evaluation

The evaluation presented in Section 10.4 was repeated with the modified equations above. The results obtained are shown in the figures below.

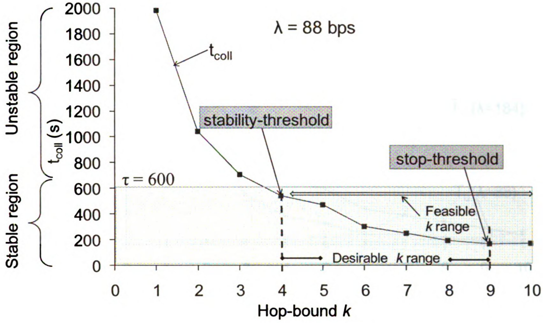


Figure A-1: Impact of *k* on trip collection-time for low data rates

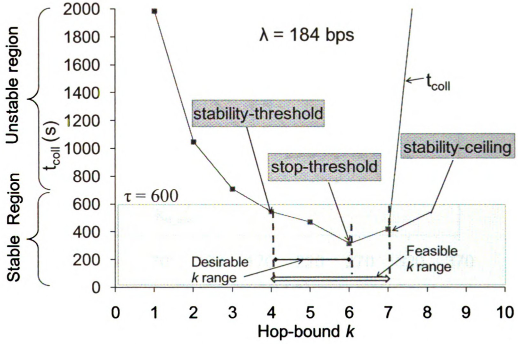


Figure A-2: of *k* on trip collection-time for higher data rates

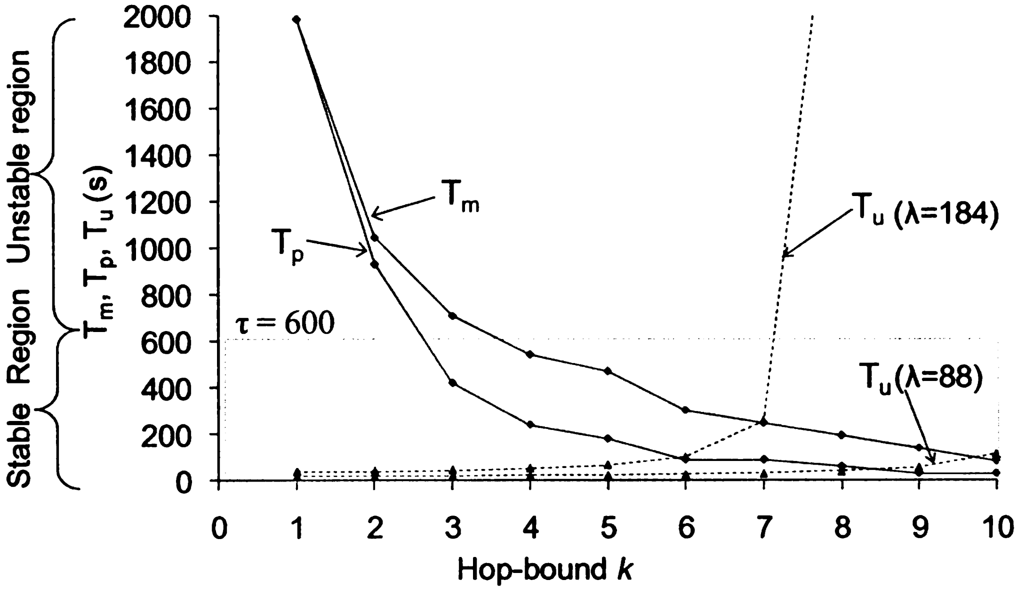


Figure A-3: Impact of *k* on T_m, T_p and T_u

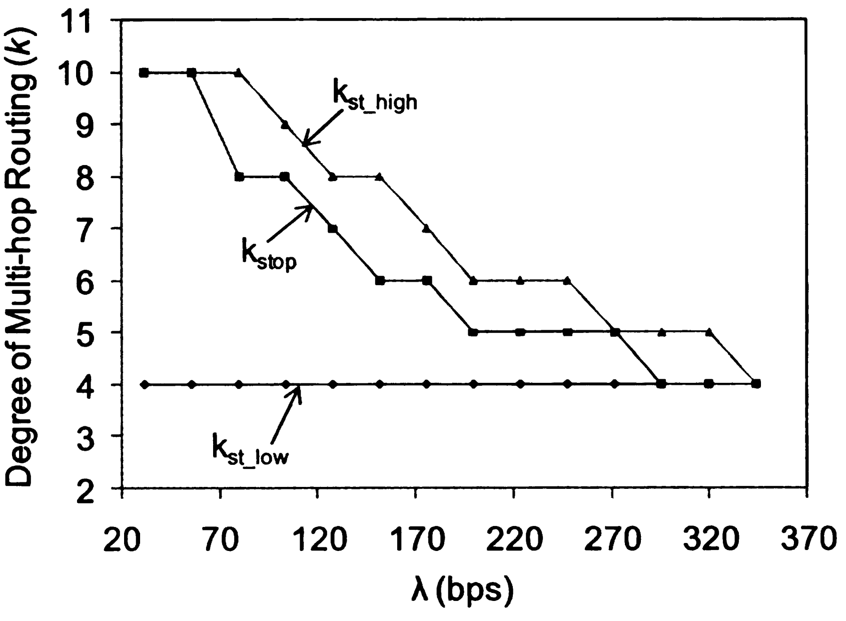


Figure A-4: Impact of data rate on thresholds

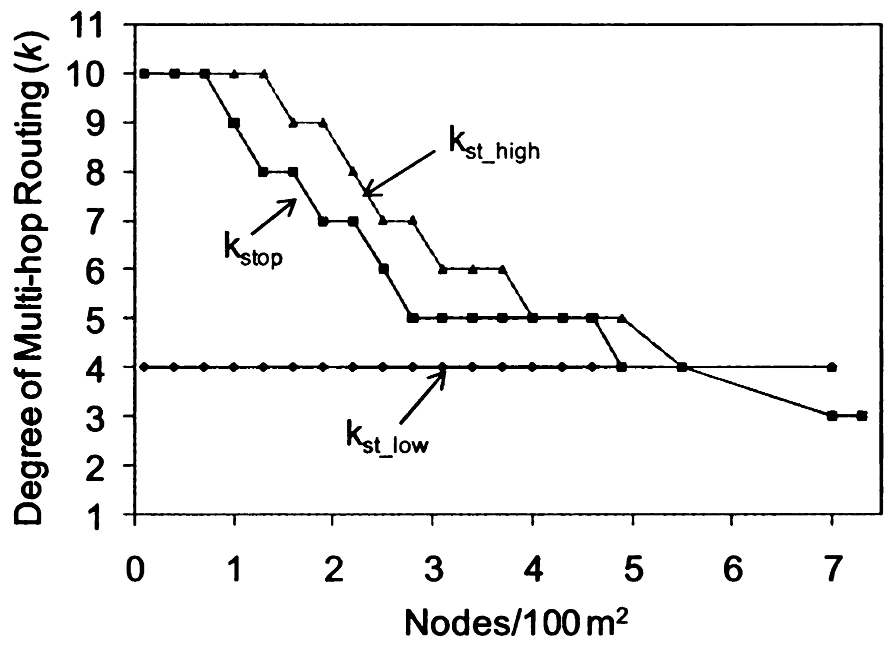


Figure A-5: Impact of density on thresholds

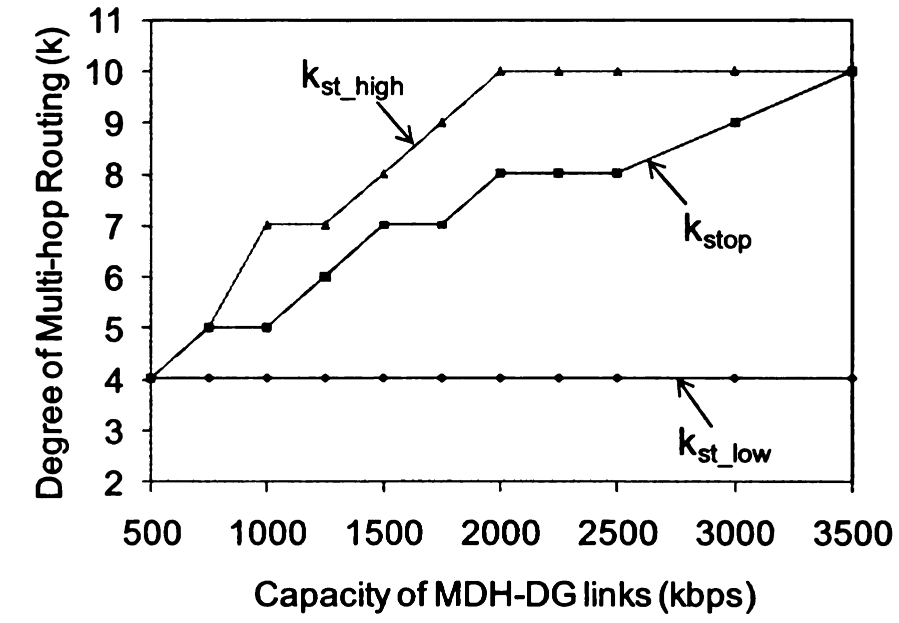


Figure A-6: Impact of MDH-DG link capacity on thresholds

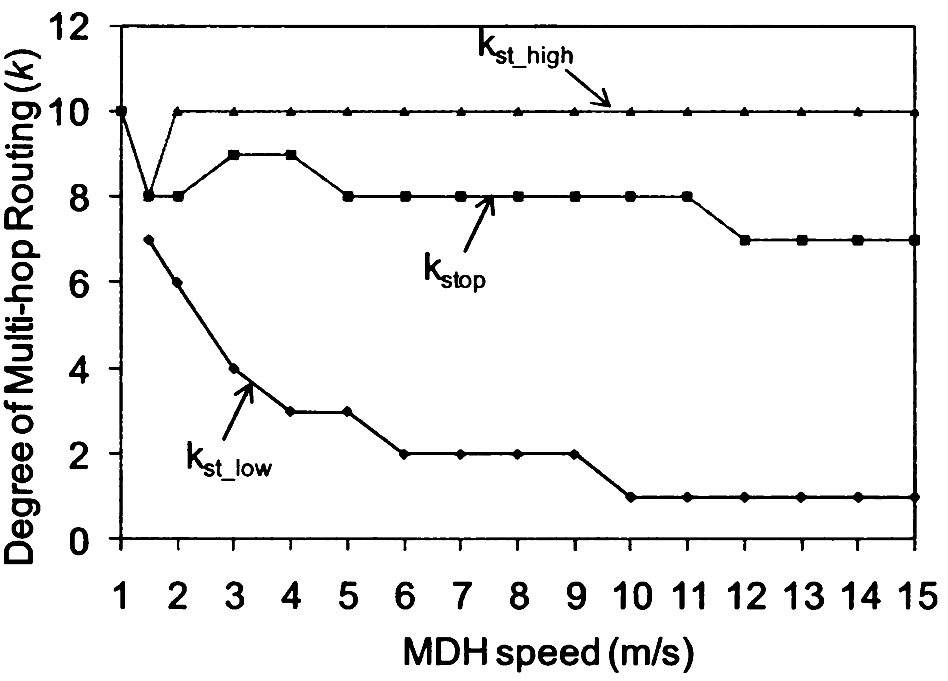


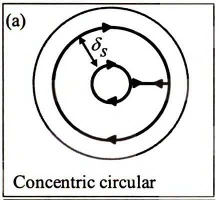
Figure A-7: Impact of speed on thresholds

A.4 Conclusions

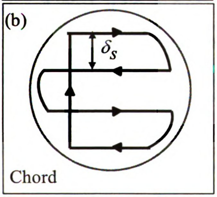
The results presented in this section show that although the absolute values are slightly different, the overall trends for the trip collection-time and the stability thresholds are the same as those observed in Chapter 10. Thus, irrespective of value of the coverage parameter δ_S, the analysis presented in Chapters 7, 9 and 10 hold.

APPENDIX B: Length of Sample Covering Trajectories

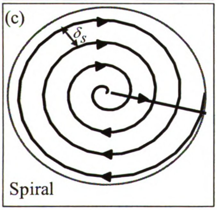
The trajectory length L depends on the covering trajectory and can be calculated as shown below for the sample trajectories discussed in Chapter 6.



$$L = 2\pi \sum_{i=1}^{R/2kr} \left[R - \left(kr(2i-1)\right)\right] + 2R_m$$



$$L = 2\sum_{i=1}^{R/\delta_S} \left\{ \sqrt{R^2 - \left[R - \delta_S(2i-1)\right]^2} - \delta_S \right\} + \frac{3\pi}{5}\left\{R - \frac{\delta_S}{2}\right\} + 2R_{m2}$$



$$L = \left(\frac{\pi}{\delta_S}\right)\left\{R^2 - \left(\frac{\delta_S}{2}\right)^2\right\} + 2R_{m3}$$

Figure B-1: Trajectory length computation for sample covering trajectories

References

[1] H. Karl and A. Willig, "A Short Survey of Wireless Sensor Networks," Technical Univeristy Berlin 2003.

[2] I. F. Akyildiz, W. Su, and Y. Sankarasubramaniam, "A Survey on Sensor Networks," in *IEEE Communications Magazine*. vol. 40, 2002, pp. 102-114.

[3] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless Sensor Network Survey," *Elsevier Computer Neworks,* vol. 52, pp. 2292-2330, August 2008.

[4] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," in *IEEE INFOCOM*, 2002, pp. 1567-1576.

[5] T. Arampatzis, J. Lygeros, and S. Manesis, "A survey of applications of wireless sensors and wireless sensor networks," in *Mediterranean conference on control and automation*, 2005.

[6] S. Singh and C. S. Raghavendra, "PAMAS: Power Aware Multi-access Protocol with Signaling for Ad Hoc Networks," *ACM Computer Communication Review,* vol. 28, pp. 5-26, July 1998.

[7] A. Somasundara, A. Kansal, D. Jea, D. Estrin, and M. B. Srivastava, "Controllably Mobile Infrastructure for Low Energy Embedded Networks," *IEEE Transactions on Mobile Computing,* vol. 5, 2006.

[8] Q. Li and D. Rus, "Communication in disconnected ad hoc networks using message relay," *J. Parallel Distrib. Comput.*, vol. 63, pp. 75-86, 2003.

[9] I. Solis and K. Obraczka, "Isolines: Energy-efficient Mapping in Sensor Networks," in *IEEE Symposium on Computers and Communication (ISCC)* Cartagena, Spain, 2005.

[10] K. Kotay, R. Peterson, and D. Ru, "Experiments with Robots and Sensor Networks for Mapping and Navigation," in *International Conference on Field and Service Robotics* Port Douglas, Australia, 2005.

[11] J. D. Nicoud and P. Machler, "Robots for Anti-personnel Mine Search," *Control Engineering Practice,* vol. 4, pp. 493-498, April 1996.

[12] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Oeh, and D. Rubenstein, "Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with Zebranet," in *Architectural Support for Programming Languages and Operating Systems (ASPLOS)* San Jose, CA, 202.

[13] T. Small and Z. Haas, "The Shared Wireless Infostation Model - A New Ad Hoc

Networking Paradigm (or Where There Is a Whale,There Is a Way)," in *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)* Annapolis, MD, 2003.

[14] A. Chakrabarti, A. Sabharwal, and B. Aazhang, "Using Predictable Observer Mobility for Power Efficient Design of Sensor Networks," in *International Conference on Information Processing in Sensor Networks* Palo Alto, CA, 2003.

[15] A. V. Savkin, "Coordinated collective motion of groups of autonomous mobile robots: analysis of Vicsek's model," *IEEE transactions on Automatic Control,* 2004.

[16] F. Amigoni, G. Fontana, and S. Mazzuca, "Robotic Sensor Networks: An Application to Monitoring Electro-Magnetic Fields," in *Emerging Artificial Intelligence Applications in Computer Engineering*, I. Maglogiannis, K. Karpouzis, M. Wallace, and J. Soldatos, Eds.: IOS Press, 2007.

[17] J. Reich and E. Sklar, "Toward Automatic Reconfiguration of Robot-Sensor Networks for Urban Search and Rescue," in *Agent Technology for Disaster Management (ATDM) Workshop at Autonomous Agents and Multi-Agent Systems (AAMAS)* 2006.

[18] J. Reich and E. Sklar, "Robot-sensor Networks for Search and Rescue," in *International Workshop on Safety, Security and Rescue Robotics (SSRR)*, 2006.

[19] M. Chen, T. Kwon, Y. Yuan, and V. C. M. Leung, "Mobile Agent Based Wireless Sensor Networks," *Journal of Computers,* vol. 1, pp. 14-21, April 2006.

[20] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less Low Cost Out Door Localization for Very Small Devices," Computer Science Department, USC 2000.

[21] A. Savvides, C.-C. Han, and M. Srivastava, "Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors," in *ACM MobiCom*, 2001.

[22] T. Small and Z. Haas, "The Shared Wireless Infostation Model - A New Ad Hoc Networking Paradigm (or Where There Is a Whale,There Is a Way)," in *Proc. ACM MobiHoc*, 2003.

[23] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein, "Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with Zebranet," in *Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2002.

[24] J. K. Hart and K. Martinez, "Environmental Sensor Networks: A revolution in the earth system science," *Earth-Science Reviews,* vol. 78, pp. 177-191, 2006.

[25] A. Kansal, W. Kaiser, G. J. Pottie, M. B. Srivastava, and G. Sukhatame, "Reconfiguration Methods for Mobile Sensor Networks," UCLA, 2006.

[26] R. Pon, M. Batalin, V. Chen, A. Kansal, D. Liu, M. Rahimi, L. Shirachi, and e. al., "Coordinated Static and Mobile Sensing for Environmental Monitoring," in *IEEE International Conference on Distributed Computing in Sensor Systems* Marina Del Rey, CA, 2005.

[27] B. Zhang and G. S. Sukhatme, "Adaptive Sampling for Estimating a Scalar Field using a Robotic Boat and a Sensor Network," in *IEEE International Conference on Robotics and Automation* Rome, Italy, 2007.

[28] G. S. Sukhatme, A. Dhariwal, B. Zhang, C. Oberg, B. Stauffer, and D. A. Caron, "The Design and Development of a Wireless Robotic Networked Aquatic Microbial Observing System," *Environmental Engineering Science,* vol. 24, pp. 205-215, 2006.

[29] T. He, S. Krishnamurthy, J. A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, and e. al., "Energy-Efficient Surveillance System Using Wireless Networks," in *International Conference on Mobile Systems, Applications and Services (MobiSys)* Boston, MA, 2004.

[30] Y. Tseng, Y. Wang, K. Cheng, and Y. Hsieh, "iMouse: An Integrated Mobile Surveillance and Wireless Sensor System," *IEEE Computer* vol. 40, pp. 60-66, 2008.

[31] T. He, P. Vicaire, T. Yan, L. Luo, L. Gu, G. Zhou, R. Stoleru, Q. Cao, J. A. Stankovic, and T. Abdelzaher, "Achieveing Real-Time Target Tracking Using Wireless Sensor Networks," in *IEEE Real-Time and Embedded Technology and Applications Symposium* San Jose, CA, 2006.

[32] D. P. Eickstedt and M. R. Benjamin, "Cooperative Target Tracking in a Distributed Autonomous Sensor Network," in *IEEE Oceans* Boston, MA, 2006.

[33] M. U. Ilyas and H. Radha, "Bounding A Statistical Measure Of Network Lifetime For Wireless Sensor Networks," in *Conference on Information Sciences & Systems (CISS'07)*, Baltimore, MD, 2007.

[34] "MICA wireless measurement system. Datasheet, Crossbow Technology Inc. http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA.pdf."

[35] V. Rajendran, K. Obraczka, and J.J.Garcia-Luna-Aceves, "Energy-Efficient, Collision-free Medium Access Control for Wireless Sensor Networks," in *ACM SenSys* Los Angeles, CA, 2003.

[36] A. Keshavarzian, H. Lee, and L. Venkataraman, "Wakeup Scheduling in Wireless Sensor Networks," in *ACM Internation Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)* Florence, Italy, 2006.

[37] S. Datta and S. Biswas, "Limiting carrier-sense energy consumption by low-power interface idling in 802.11 protocols," in *IEEE ICWN*, Phoenix, Arizona,

2004.

[38] I. Lazaridis and S. Mehrotra, "Capturing Sensor Generated Time Series with Quality Guarantees," in *IEEE International Conference on Data Engineering (ICDE)* Bangalore, India, 2003.

[39] Q. Han, S. Mehrotra, and N. Venkatasubramanian, "Energy Efficient Data Collection in Distributed Sensor Environments," in *International Conference on Distributed Computing Systems (ICDCS)* Tokyo, Japan, 2004.

[40] S. Park and R. Sivakumar, "Energy Efficient Correlated Data Aggregation for Wireless Sensor Networks," *International Journal of Distributed Sensor Networks,* vol. 4, pp. 13-27, 2008.

[41] R. Naik, S. Biswas, and S. Dutta, "Distributed Sleep-Scheduling Protocols for Energy Conservation in Wireless Networks," in *38th Annual Hawaii International Conference on System Sciences (HICSS'05)* Hawaii, 2005.

[42] J. H. Chang and L. Tassiulas, "Energy conserving routing in wireless ad hoc networks," in *IEEE INFOCOM*, 2000, pp. 22-31.

[43] R. C. Shah and J. M. Rabaey, "Energy Aware Routing for Low Energy Ad Hoc Sensor Networks," in *IEEE Wireless Communications Network Conference*, 2002, pp. 350-355.

[44] C. K. Toh, "Maximum battery life routing to support ubiquitous mobile computing in wireless ad hoc networks," in *IEEE Communications Magazine*, June ed. vol. 39, 2001, pp. 138-147.

[45] H. M. Ammari and S. K. Das, "Trade-off between energy savings and source-to-sink delay in data dissemination for wireless sensor networks," in *Proceedings of the 8th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems* Montr\&\#233;al, Quebec, Canada: ACM, 2005.

[46] X. Hong, M. Gerla, H. Wang, and L. Clare, "Load balanced energy-aware communications for mars sensor networks," in *IEEE Aerospace*, 2002.

[47] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks," in *Hawaii International Conference on System Sciences*, Hawaii, 2000.

[48] V. Rodoplu and T. H. Meng, "Minimum energy mobile wireless networks," *IEEE Journal of Selected Areas in Communications,* vol. 17, pp. 1333-1344, 1999.

[49] M. Devendar, D. Xiaojiang, D. Fei, and Y. Chao, "Load balance and energy efficient data gathering in wireless sensor networks." vol. 8: John Wiley and Sons Ltd., 2008, pp. 645-659.

[50] C. Li, M. Ye, G. Chen, and J. Wu, "An Energy Efficient Unequal Clustering Mechanism for Wireless Sensor Networks," in *IEEE International Conference on Mobile Ad Hoc and Sensor Systems* Washington, DC, 2005.

[51] S. A. Borbash and E. H. Jennings, "Distributed Topology Control Algorithm for Multi-hop Wireless Networks," in *World Congress on Computational Intelligence (WCCI)* Honolulu, Hawaii, 2002.

[52] P. K. Sahoo, J.-P. Sheu, and K.-Y. Hsieh, "Power control based topology construction for the distributed wireless sensor networks," vol. 30, pp. 2774-2785, 2007.

[53] N. Li and J. Hou, "Topology Control in Heterogeneous Wireless Networks: Problems and Solutions," in *IEEE Conference on Computer Communications (INFOCOM)* Hong Kong, China, 2004.

[54] R. Ramanathan and R. Rosales-Hain, "Topology Control of Multi-hop Wireless Networks using Transmit Power Adjustment," in *IEEE Conference on Computer Communications (INFOCOM)* Tel-Aviv, Israel, 2000.

[55] Y. U. Cao and A. S. Fukunaga, "Cooperative Mobile Robotics: Antecedents and Directions," *Autonomous Robots,* vol. 4, pp. 1-23, 1997.

[56] G. Wang, G. Cao, and T. L. Porta, "Movement assisted sensor deployment," in *IEEE Conference on Computer Communications (INFOCOM)* Hong Kong, 2004.

[57] P. Basu and J. Redi, "Movement control algorithms for realization of fault-tolerant ad hoc robot networks," in *IEEE Network.* vol. July, 2004.

[58] S. Jain, R. C. Shah, G. Borriello, W. Brunette, and S. Roy, "Exploiting Mobility for Energy Efficient Data Collection in Sensor Networks," in *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, 2004.

[59] Q. Li and D. Rus, "Sending Messages to mobile users in disconnected ad-hoc wireless networks," in *ACM MobiCom*, 2000.

[60] D. Puccinelli, M. Brennan, and M. Haenggi, "Reactive Sink Mobility in Wireless Sensor Networks," in *MobiOpp* San Juan, Puerto Rico: ACM, 2007.

[61] G. Cao and G. Kesidis, *Purposeful mobility in tactical sensor networks*: IEEE Press, 2005.

[62] M. H. Rahimi, H. Shah, G. S. Sukhatame, J. Heidemann, and D. Estrin, "Energy harvesting in mobile sensor networks " in *IEEE international Conference on Robotics and Automation*, Taipei, Taiwan, 2003.

[63] M. Lisovich, S. Bermudez, and S. Wicker, "Reconfiguration in Heterogeneous Mobile Wireless Sensor Networks," in *ISWPC*, 2008.

[64] Z.Vincze, D. Vass, R. Vida, A. Vidács, and A. Telcs, "Adaptive sink mobility in event-driven multi-hop wireless sensor networks," in *Proceedings of the first international conference on Integrated internet ad hoc and sensor networks* Nice, France 2006.

[65] C. Frank and K. Romer, "Algorithms for Generic Role Assignment in Wireless Sensor Networks," in *ACM Conference on Embedded Networked Sensor Systems (SenSys)* San Diego, CA, 2005.

[66] Y. Yu, L. J. Rittle, V. Bhandari, and J.B.LeBrun, "Supporting Concurrent Applications in Wireless Sensor Networks," in *4th International Conference on Embedded Networked Sensor Systems (SenSys)*, Boulder, CO, 2006.

[67] M. L. Sichitiu and V. Ramadurai, "Localization of Wireless Sensor Networks with a Mobile Beacon," in *IEEE Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, 2004.

[68] W. Zhao, M. Ammar, and E. Zegura, "A Message Ferrying Approach for Data Delivery in Sparse Mobile Ad Hoc Networks," in *The ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)* Tokyo, Japan, 2004.

[69] R. C. Shah, S. Roy, S. Jain, and W. Brunette, "Data MULEs: Modeling a Three-tier Architecture for Sparse Sensor Networks," in *IEEE SNPA Workshop* Achorage, Alaska, 2003.

[70] M. M. B. Tariq, M. Ammar, and E. Zegura, "Message ferry route design for sparse ad hoc networks with mobile nodes," in *Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing* Florence, Italy: ACM, 2006.

[71] J. Lou and J. P. Hubaux, "Joint mobility and routing for lifetime elongation in wireless sensor networks," in *IEEE INFOCOM*, 2005.

[72] W. Wang, V. Srinivasan, and K. Chua, "Using Mobile Relays to Prolong the Lifetime of Wireless Sensor Networks," in *ACM International Conference on Mobile Computing and Networking (MobiCom)* Cologne, Germany 2005.

[73] A. Kansal, A. Somasundara, D. Jea, M. Srivastava, and D. Estrin, "Intelligent Fluid Infrastructure For Embedded Networks," in *International Conference on Mobile Systems, Applications and Services (MobiSys)* Boston, MA, 2004.

[74] S. Eidenbenz, L. Kroc, and J. P. Smith, "Maneuverable Relays to Improve Energy Efficiency in Sensor Networks," in *IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom)* Kauai, Hawaii, 2005.

[75] J. Ma, C. Chen, and J. P. Salomaa, "mWSN for Large Scale Mobile Sensing," *Journal of Signal Processing Systems*, vol. 51, pp. 195-206, 2008.

[76] E. Ekici, Y. Gu, and D. Bozdag, "Mobility-based Communication in Wireless Sensor Networks," in *IEEE Communications Magazine*, July ed. vol. 44, 2006, pp. 56-62.

[77] Y. Guo and Z. Qu, "Coverage Control for a Mobile Robot Patrolling a Dynamic and Uncertain Environment," in *Fifth World Congress on Intelligent Control and Automation* Hangzhou, China, 2004.

[78] M. Jian, C. Canfeng, and P. S. Jyri, "mWSN for Large Scale Mobile Sensing." vol. 51: Kluwer Academic Publishers, 2008, pp. 195-206.

[79] M. Ma and Y. Yang, "Data Gathering in Wireless Sensor Networks with Mobile Collectors," in *IEEE International Conference on Parallel and Distributed Processing (IPDPS)* Rome, Italy, 2008.

[80] W. Shaw, Y. He, and I. Lee, "Mobile Sink to Track Multiple Targets in Wireless Virtual Sensor Networks," in *International Symposium on Ubiquitous Multimedia Computing*, Hobart, Australia, 2008.

[81] S. R. Gandham, M. Dawande, R. Prakash, and S. Venkatesan, "Energy Efficient Schemes for Wireless Sensor Networks with Mutiple Mobile Base Stations," in *IEEE GlobeCom* San Francisco, 2003.

[82] D. Johnson, T. Stack, R. Fish, D. Flickinger, L. Stoller, R. Ricci, and J. Lepreau, "Mobile Emulab: A Robotic Wireless and Sensor Network Testbed," in *IEEE INFOCOM* 2006.

[83] G. Sibley, M. Rahimi, and G. Sukhatame, "Robomote: A Tiny Mobile Robot Platform for Large-Scale Ad-hoc Sensor Networks," in *International Conference on Robotics and Automation* Washington, D.C, 2002.

[84] D. Goldenberg, J. Lin, A. S. Morse, B. E. Rosen, and Y. R. Yang, "Towards Mobility as a Network Control Primitive," in *MobiHoc*, 2004.

[85] C. Tang and P. K. McKinley, "Energy Optimization under Informed Mobility," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, 2006.

[86] J. In and J. Kim, "Novel Sink-Oriented Approach for Efficient Sink Mobility Support in Wireless Sensor Networks " in *International Conference on Mobile Sensor Networks (MSN'06)* Hong Kong, 2006.

[87] J. Luo and J. Panchard, "MobiRoute: Routing towards a Mobile Sink for Improving Lifetime in Sensor Networks," in *International Conference on Distributed Computing in Sensor Systems (DCOSS '06)*, San Francisco, 2006.

[88] K. Akkaya and M. Younis, "Energy-aware Routing to a Mobile Gateway in Wireless Sensor Networks," in *IEEE Global Telecommunications Conference (GLOBECOM)* Dallas, TX, 2004.

[89] P. Baruah, R. Urgaonkar, and B. Krishnamachari, "Learning Enforced Time Domain Routing to Mobile Sinks in Wireless Sensor Fields " in *First IEEE Workshop on Embedded Networked Sensors (EmNetS-I), held in conjunction with IEEE LCN* Tampa, FL, 2004.

[90] R. W. N. Pazzi and A. Boukerche, "Mobile data collector strategy for delay-sensitive applications over wireless sensor networks." vol. 31: Butterworth-Heinemann, 2008, pp. 1028-1039.

[91] M. Wang, S. Basagni, and e. al, "Exploiting Sink Mobility for Maximizing Sensor Networks Lifetime " in *Hawaii International Conference on System Sciences 2005 (HICSS-38)* Hawaii, 2005.

[92] S. Basagni, A. Carosi, and e. al, "A New MILP Formulation and Distributed Protocols for Wireless Networks Lifetime Maximization," in *International Conference on Communications (ICC'06)* Istanbul, Turkey, 2006.

[93] I. Papadimitriou and L. Georgiadis, "Energy-aware Routing to Maximize Lifetime in Wireless Sensor Networks with Mobile Sink," in *International Conference on Software, Telecommunications and Computer Networks (SoftCOM)* Split, Croatia, 2005.

[94] S. Basagni, A. Carosi, E. Melachrinoudis, C. Petrioli, and Z. M. Wang, "Controlled Sink Mobility for Prolonging Wireless Sensor Networks Lifetime," *ACM/Springer Wireless Networks Journal,* vol. 14, pp. 831-858, 2008.

[95] Y. Bi, L. Sun, J. Ma, N. Li, I. A. Khan, and C. Chen, "HUMS: An Autonomous Moving Strategy for Mobile Sinks in Data-Gathering Sensor Networks," *EURASIP Journal on Wireless Communications and Networking,* 2007.

[96] S. Bill and G. Grant, "Small UGV Platforms for Unattended Sensors " in *SPIE,* Bruges, Belgium, 2005, pp. 101-110.

[97] A. Somasundara, A. Ramamoorthy, and M. B. Srivastava, "Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines," in *IEEE Real-Time Systems Symposium (RTSS)* Lisbon, Portugal, 2004.

[98] Y. Gu, D. Bozdag, E. Ekici, F. Ozguner, and C. G. Lee, "Partitioning based mobile element scheduling in wireless sensor networks," in *Sensor and Ad Hoc Communication and Networks (SECON)* Santa Clara, CA, 2005.

[99] W. Aioffi, G. Mateus, and F. Quintao, "Optimization issues and algorithms for wireless sensor networks with mobile sink," in *International network optimization conference* Belgium, 2007.

[100] Y. Gu, D. Bozdag, R. W. Brewer, and E. Ekici, "Data Harvesting with Mobile Elements in Wireless Sensor Networks," *Computer Networks,* vol. 50, pp.

3449-3465, 2006.

[101] M. Zhao, M. Ma, and Y. Yang, "Mobile Data Gathering with Space-Division Multiple Access in Wireless Sensor Networks," in *IEEE Conference on Computer Communications (INFOCOM)* Phoenix, AZ, 2008.

[102] R. Sugihara and R. K. Gupta, "Improving the Data Delivery Latency in Sensor Networks with Controlled Mobility," in *International Conference on Distributed Computing in Sensor Systems (DCOSS)* Marina Del Rey, CA, 2008.

[103] E. M. Saad, M. H. Awadalla, and R. R. Darwish, "Adaptive Energy-Aware Gathering Stratey for Wireless Sensor Networks," *International Journal of Computers,* vol. 2, pp. 148-157, 2008.

[104] E. M. Saad, M. H. Awadalla, M. A. Saleh, H. Keshk, and R. R. Darwish, "A data gathering algorithm for a mobile sink in largescale sensor networks," in *10th WSEAS International Conference on Mathematical Methods and Computational Techniques in Electrical Engineering* Sofia, Bulgaria, 2008.

[105] L. Lima and J. Barros, "Random walks in sensor networks," in *International Symposium on Modeling and Optimization in Mobile Ad Hoc and Wireless Netowrks (WiOpt)* Cyprus, 2007.

[106] I. Chatzigiannakis, A. Kinalis, and S. Nikoletseas, "Sink mobility protocols for data collection in wireless sensor networks," in *ACM International Symposium on Mobility Management and Wireless Access (MOBIWAC)* Spain, 2006.

[107] D. Jea, A. Somasundara, and M. B. Srivastava, "Multiple Controlled Mobile Elements (Data Mules) for Data Collection in Sensor Networks," in *IEEE/ACM International Conference on Distributed Computing in Sensor Systems (DCOSS)* Marina del Rey, California, 2005.

[108] G. Xing, T. Wang, Z. Xei, and W. Jia, "Rendezvous Planning in Wireless Sensor Networks with Mobile Elements," *IEEE Transactions on Mobile Computing* vol. 7, December 2008.

[109] O. Tekdas, J. H. Lim, A. Terzis, and V. Isler, "Using Mobile Robots to Harvest Data from Sensor Fields," in *IEEE Wireless Communications*, February ed, 2009.

[110] R. C. Shah, S. Roy, and S. Jain, "Data MULEs: Modeling a Three-Tier Architecture for Sparse Sensor Networks," *Elsevier Ad Hoc Networks,* vol. 1, pp. 215-233, 2003.

[111] W. Zhao, M. Ammar, and E. Zegura, "Controlling the Mobility of Multiple Data Transport Ferries in a Delay-Tolerant Network," in *IEEE Conference on Computer Communications (INFOCOM)* Miami, FL, 2005.

[112] B. Wang, D. Xie, C. Chen, J. Ma, and S. Cheng, "Deploying Multiple Mobile

Sinks in Event-Driven WSNs," in *International Conference on Communications (ICC)* Beijing, China, 2008.

[113] A. Bogdanov, E. Maneva, and S. Riesenfeld, "Power-aware Base Station Positioning for Sensor Networks," in *IEEE INFOCOM*, 2004.

[114] A. P. Azad and A. Chockalingam, "Mobile Base Stations Placement and Energy Aware Routing in Sensor Networks," in *IEEE Wireless Communications and Networking Conference (WCNC)* Las Vegas, NV, 2006.

[115] J. Pan, L. Cai, and e. al, "Optimal Base Station Locations in Two Tiered Wireless Sensor Networks," *IEEE Transactions on Mobile Computing,* vol. 4, 2005.

[116] Y. Shi, Y.T.Hou, and A. Efrat, "Algorithm Design for Base Station Placement Problems in Sensor Networks," in *QShine'06* Waterloo, Canada, 2006.

[117] A. Mainwaring, J. Polastre, R. Szewczyk, and D. Culler, "Wireless sensor networks for habitat monitoring," in *First ACM International Workshop on Wireless Sensor Networks and Applications*, Atlanta, GA, 2002.

[118] S. D. Hanford, L. N. Long, and J. F. Horn, "A Small Semi-Autonomous Rotary-Wing Unmanned Air Vehicle (UAV)," in *American Institute of Aeronautics and Astronautics Aerospace Conference*, 2007.

[119] "Emperor penguin crèches and altruism. http://www.dmclf.org/articles/creches.html.."

[120] "Website of the Lego Mindstorms Robotics Inventions System 2.0 http://mindstorms.lego.com/eng/default.asp."

[121] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao, "Habitat monitoring: Application driver for wireless communications technology," in *ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean*, 2001.

[122] J. N. Karaki, "Data Aggregation in Wireless Sensor Networks - Exact and Approximate Algorithms," in *IEEE Workshop on High Performance Switching and Routing* Phoenix, AZ, 2004.

[123] R. Rozovsky and P. Kumar, "SEEDEX: A MAC Protocol for Ad Hoc Networks," in *ACM MobiHoc*, 2001.

[124] Z. Chen and A. Khokhar, "Self organization and energy efficient TDMA MAC protocol by Wake up for wireless sensor networks," in *IEEE Conference on Sensors and Adhoc Communication and Networks (SECON)*, 2004.

[125] M. Sichitiu, "Cross Layer Scheduling for Power Efficiency in Wireless Sensor Networks," in *IEEE Confernece on Computer Communications (INFOCOM)* Hong Kong, 2004.

[126] D. Bertsekas and R. Gallager, *Data Networks*: Englewood Cliffs, NJ: Prentice-Hall, 1992.

[127] W. Liao, Y. Tseng, and J. Sheu, "GRID: A Fully Location-Aware Routing Protocol for Mobile Ad Hoc Networks," *Telecommunication Systems,* vol. 18, pp. 37-60, 2001.

[128] I. Stojmenovic and X. Lin, "GEDIR: Loop-Free Location Based Routing in Wireless Networks," in *International Conference on Parallel and Distributed Computing and Systems* Boston, MA, 1999.

[129] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed Energy Conservation for Ad-hoc Routing," in *ACM/IEEE International Conference on Mobile Computing and Networking*, 2001.

[130] C. Schurgers and M. B. Srivastava, "Energy Efficient Routing in Wireless Sensor Networks," in *Military Communications (MILCOM)* Washington D. C, 2001.

[131] M. Youssef, M. Younis, and K. Arisha, "A Constrained Shortest-path energy-aware Routing Algorithm for Wireless Sensor Networks," in *IEEE Wireless Communications and Network Conference (WCNC)* Orlando, FL, 2002.

[132] C. Intanagonwiwat and R. Govindan, "Directed Diffusion: A scalable and robust communication paradigm for sensor networks," in *Sixth International Conference on Mobile Computing and Networking (MobiCOM'00)*, Boston, 2000.

[133] W. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive Protocols for Information Dissemination in Wireless Sensor Networks," in *ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)* Seattle, WA, 1999.

[134] Y. Wen, R. Wolski, and C. Krintz, "Online Prediction of Battery Lifetime for Embedded and Mobile Devices," in *Special Issue on Embedded Systems: Springer-Verlag Heidelberg Lecture Notes in Computer Science*, 2004.

[135] "Webpage for Lego Technic Motors http://www.philohome.com/motors/motorcomp.htm."

[136] W. Stallings, "IEEE 802.11: Moving Closer to Practical Wireless LANs," *IT Professional,* vol. 3, pp. 17-23, May-June 2001.

[137] G Held, "The ABCs of IEEE 802.11," *IT Professional,* vol. 3, pp. 49-52, Nov-Dec 2001.

[138] S. Biswas, S. Gupta, F. Yu, and T. Wu, "Collaborative Multi-target Tracking using Networked Robotic Vehicles," in *SPIE Defense and Security Symposium* Orlando, Florida, 2007.

[139] C. Tang and P. K. McKinley, " iMobif: An Informed Mobility Framework for Energy Optimization in Wireless Ad Hoc Networks," in *International Conference on Distributed Computing Systems (ICDCS) Workshops* 2005.

[140] W. Ren, "On Constrained Nonlinear Tracking Control of a Small Fixed-wing UAV," *Journal of Intelligent Robot Systems,* vol. 48, pp. 525-537, 2007.

[141] M. Quigley, M. A. Goodrich, and R. W. Beard, "Semi-Autonomous Human-UAV Interfaces for Fixed-wing Mini-UAVs," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2004.

[142] F. Nocetti and J. Gonzalez, "Connectivity Based k-hop Clustering in Wireless Networks," *Telecommunication Systems,* vol. 22, pp. 1-4, 2003.

[143] E. Tunstel, G. Anderson, and E. Wilson, "Motion Trajectories for Wide-Area Surveying with a Rover-based Distributed Spectrometer," in *World Automation Congress (WAC)* Budapest, Hungary, 2006.

[144] Y. Mei, Y.-H. Lu, Y. C. Hu, and C. S. G. Lee, "Energy-efficient Motion Planning for Mobile Robots " in *IEEE International Conference on Robotics and Automation (ICRA)* New Orleans, LA, 2004.

[145] M. R. Garey and D. S. Johnson, *A guide to the Theory of NP-Completeness*. San Francisco: W. H. Freeman, 1979.

[146] B. Han, H. Fu, L. Lin, and W. Jia, "Efficient construction of connected dominating set in wireless ad hoc networks," in *IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)* Fort Lauderdale, FL, 2004.

[147] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm intelligence*: Oxford university press, 1999.

[148] S. Chandran, *Advances in Direction-of-Arrival Estimation*: Artech House, 2006.

[149] "IEEE Standard for Wireless LAN-Medium Access Control and Physical Layer Specification," November 1997.

[150] T. Spiez, J. Bange, M. Buschmann, and P. Vorsmann, "First Application of the Meteorological Mini-UAV M^{2}AV," *Meteorologische Zeitschrift,* vol. 16, pp. 159-169, 2007.

[151] K. R. S. Santini, "An Adaptive Strategy for Quality-based Data Reduction in Wireless Sensor Networks," in *International Conference on Networked Sensing Systems (INSS)* Chicago, IL, 2006.

[152] A. Jain and E. Y. Chang, "Adaptive Sampling for Sensor Networks," in *International Workshop on Data Management for Sensor Networks* New York, NY: ACM, 2004.

[153] M. Marta and M. Cardei, "Improved Sensor Network Lifetime with Multiple Mobile Sinks," *Elsevier Pervasive and Mobile Computing,* vol. In press, 2009.

[154] I. F. Akyildiz, D. Pompili, and T. Melodia, "Underwater Acoustic Sensor Networks: Research Challenges," *Elsevier Ad Hoc Networks,* vol. 3, pp. 257-279, May 2005.

[155] D. Kedar and S. Arnon, "A Distributed Sensor System for Detection of Contaminants in the Ocean," in *SPIE, the International Society for Optical Engineering* Stockholm, Sweden, 2006.

[156] J. Heidemann, Y. Li, A. Syed, J. Wills, and W. Ye, "Research Challenges and Applications for Underwater Sensor Networking," in *IEEE Wireless Communications and Networking Conference (WCNC)* Las Vegas, Nevada, 2006.