Mobile data collectors in wireless sensor networks

Waleed Al-Salih

A thesis submitted to the School of Computing
in conformity with the requirements
for the degree of Doctor of Philosophy

Queen's University
Kingston, Ontario, Canada
April, 2009

Your file Votre référence
ISBN: 978-0-494-65438-5
Our file Notre référence
ISBN: 978-0-494-65438-5

Abstract

Recent advances in wireless and sensing technologies have enabled the deployment of large scale Wireless Sensor Networks (WSNs) which have a wide range of scientific and commercial applications. However, due to the limited energy supply of sensor nodes, extending the network lifetime has become crucial for WSNs to deliver their promised benefits. Several proposals have aimed at this objective by designing energy efficient protocols at the physical, medium access, and network layers. While the proposed protocols achieve significant energy savings for individual sensor nodes, they fail to solve topology-related problems. An example of such problems is the bottlenecks around the sink, which is a direct result of multi-hop relaying: sensor nodes around the sink relay data generated all over the network which makes them deplete their energy much faster than other nodes.

A natural solution to this problem is to have multiple mobile data collectors so that the load is distributed evenly among all nodes. We investigate this promising direction for balancing the load and, hence, prolonging the lifetime of the network. We design optimization schemes for routing and placement of mobile data collectors in WSNs. We show, by theoretical analysis and simulations, that our approach has the potential to prolong the lifetime of the network significantly.

Acknowledgements

First of all, I thank ALLAH (God) for his mercy, help and guidance, without which this document would not be possible.

It has been a great honor for me to be supervised by Dr. Selim G. Akl and Dr. Hossam S. Hassanein. Their wide knowledge, guidance, enthusiasm in research, and nice personalities have been a great source of support for me throughout my research. Over the past four years, they have been always accessible and willing to help. I am grateful to Dr. Akl and Dr. Hassanein for the effort they spent to make me enjoy every minute I spent in research activities.

I owe my parents any success I make in my life. Their encouragement and support is a key factor in any achievement I have ever made. I am also indebted to my beloved wife, Nisren, for her continuous encouragement and patience during the course of my Ph.D. program. I also thank my children, Khaled and Salih, for their patience and for the happiness and motivation I always find in their smiles. I want to thank my brothers, Naeif, Abdullah, and Abdulrahman, and my sisters, Maha and Latifa, for their love and support.

I would like to thank all members of the School of Computing at Queens University for their support and friendship.

Special thanks to the Ministry of Higher Education in Saudi Arabia for their financial support.

Statement of Originality

I hereby certify that this Ph.D. thesis is original and that all ideas and inventions attributed to others have been properly referenced.

Contents

List of Figures

List of Tables

List of Acronyms

AUV	Autonomous Unmanned Vehicles
CPU	Central Processing Unit
CS	Carrier Sense
CTS	Clear To Send
DCP	Data Collector Placement
DCPR	Delay-Constrained Placement and Routing
DCR	Data Collector Record
DTPR	Delay-Tolerant Placement and Routing
GG	Gabriel Graph
GPS	Global Positioning System
GPSR	Greedy Perimeter Stateless Routing
HEED	Hybrid, Energy Efficient, Distributed
LEACH	Low Energy Adaptive Clustering Hierarchy
MAC	Medium Access Control
MDS	Minimum Dominating Set
MILP	Mixed Integer Linear Program
MM	Minimizing the Maximum energy
MOR	Maximal Overlapping Region
MR	Maximizing the minimum Residual energy
QoS	Quality of Service
RN	Relay Node
RNG	Relative Neighborhood Graph
RTS	Ready To Send
TDMA	Time Division Multiple Access
TNR	Trajectory Node Record
UCS	Unequal Clustering Size
WSN	Wireless Sensor Network

Chapter 1

Introduction

Advances in wireless communication and embedded microprocessors have brought about the development of small, low-cost sensor nodes, which are able to collect data from the surrounding environment and to communicate using a wireless medium. They report the data they collect to a central host, called the sink node. These tiny devices are battery-operated and, hence, untethered in terms of both power and communication. This enabled a new generation of large-scale networks of untethered, unattended sensor nodes suitable for a wide range of commercial, scientific, health, surveillance, and military applications. This rapidly evolving technology has the potential to revolutionize the way we interact with the physical environment and to facilitate collecting data which have never been available before [2]. However, the drastic energy constraints form a serious threat to the longevity of Wireless Sensor Networks (WSNs). Coping with this challenge and extending the lifetime of WSNs is the main focus of this research.

1.1 Motivations

Due to energy and size limitations, sensor nodes have a limited wireless transmission range. Therefore, sensor nodes, whose separation from the sink node is more than their

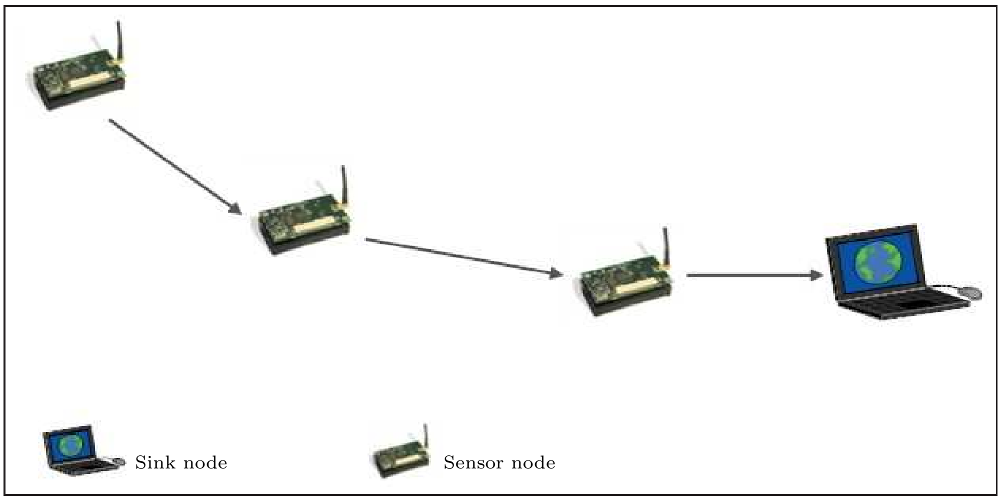


Figure 1.1: Multi-hop relaying.

transmission range, use multi-hop relaying to deliver their data to the sink node. Thereby, a sensor node may function as a router to forward data packets of other nodes to the sink node as shown in Fig. 1.1. However, data transmission is the dominant energy consuming operation in a sensor node; it has been stated in [3] that transmitting 1 Kb over a distance of 100 m would consume the same amount of energy as that of executing 3 million instructions by a local processor (i.e., the cost of sending 1 bit is equivalent to that of executing 3000 instructions). Since wireless communication is the major energy consumer in a sensor node, multi-hop relaying results in an unbalanced energy expenditure over different parts of the network: nodes near the sink become bottlenecks and deplete their energy reserves much faster than nodes distant from the sink node [4] [5] [6][7]. In fact, nodes which are one hop away from the sink node relay data generated all over the network to the sink node and, therefore, they run out of energy much faster than other nodes. Not only does this stop those nodes around the sink from functioning, but it also renders the sink unreachable by other nodes. Fig. 1.2 shows an example of a sensor network where the shaded sensor nodes are expected to have a higher data traffic than other nodes. While existing energy-aware protocols, at the physical, the Medium Access Control (MAC), and the network layers (see Section 2.2), achieve significant energy savings for individual sensor nodes, they fail to solve this topology-related problem.

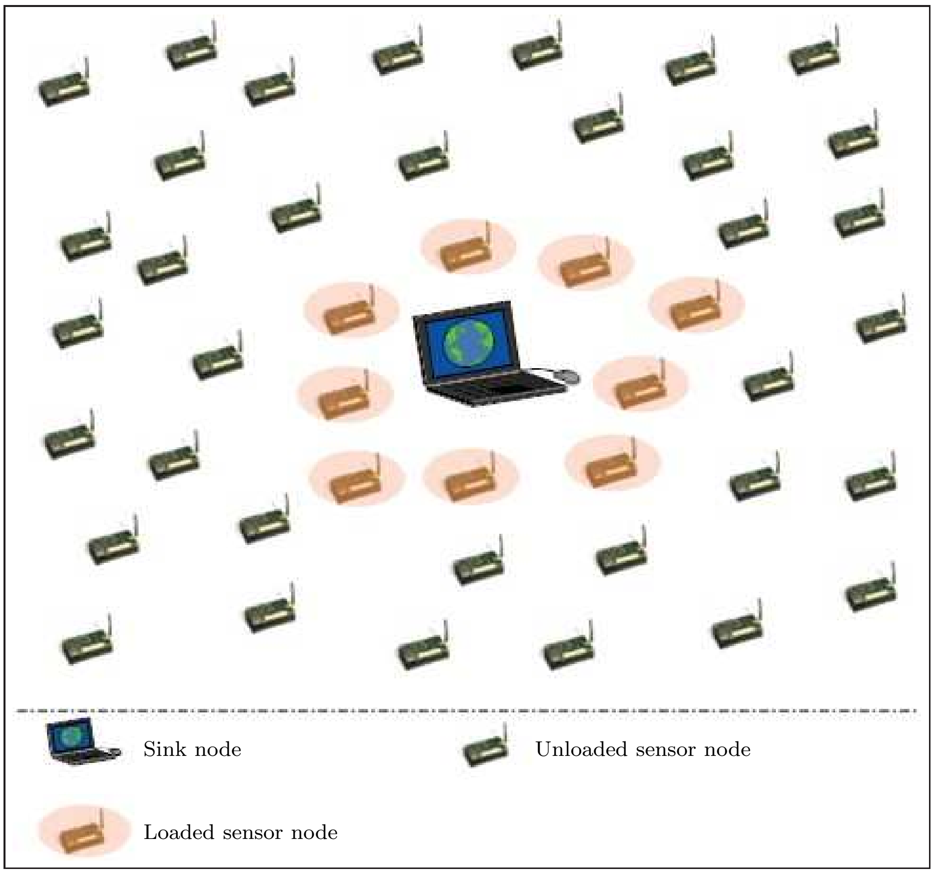


Figure 1.2: Bottlenecks around the sink.

To solve the problem of having a “hotspot” around a stationary sink, we argue for using multiple mobile sinks (which we call data collectors), which change their locations periodically. We address the problem of placing these data collectors in a way that balances the energy expenditure over different parts of the network and, hence, extends its lifetime. We also consider the problem of routing data from the stationary sensor nodes to the mobile data collectors.

With data collector mobility, the network can adapt to changes in data accumulation patterns. For example, a data collector should be located near areas where interesting events are taking place, so that data associated with these events travel over a small number of hops to save energy. The network can also adapt to changes in sensor nodes energy levels. Data collectors should be placed near nodes with relatively good amount of energy. These changes are usually driven by the application itself and by the nature of monitored events.

1.2 Thesis contributions

Our research aims primarily at prolonging the lifetime of a WSN by exploiting mobility of data collectors. We study the optimization problem of maximizing the network lifetime when mobile data collectors are in use. By having the data collectors mobile, the job of being a hotspot node is distributed over different parts of the network at different times and, hence, the relaying load disparity over the network is reduced. However, mobility of data collectors needs to be managed carefully in order to achieve that goal; and that is the focus of this research. To this end, we present novel schemes to place a set of mobile data collectors and to find the routing paths from the stationary sensor nodes to the mobile data collectors in WSNs. The main contributions of this thesis are the following:

1. In general, a data collector can be placed anywhere in the sensing field, which results in an infinite search space for data collectors locations. This has been a deterrent to full utilization of mobile data collectors [8]. We overcome this challenge by discretizing the search space without affecting the quality of the derived solutions.

2. To utilize mobility of data collectors, we divide the network lifetime into equal length rounds (e.g., hours, days, weeks, ... etc) and data collectors are moved to new locations at the beginning of each round. We define and solve the optimization problem of finding data collectors locations and routing paths from sensor nodes to data collectors for each round, such that the minimum residual energy at the end of the round is maximized. We formulate this problem as a Mixed Integer Linear Program (MILP).

3. We extend this optimization problem to an underwater 3D environment in which sensor nodes float at different depths and data collectors roam on the surface of the water.

4. We define and solve a delay-sensitive version of the previous optimization problem; we consider a delay constraint, which is an upper bound on the length of a path

between a sensor node and a data collector. This is useful for real-time applications such as a Tsunami warning system.

5. We propose a distributed routing scheme for WSNs with a mobile data collector moving along a globally known trajectory. The goal is to find routes to deliver delay-tolerant and delay-sensitive data to the data collector. While delay-tolerant data can wait for the data collector to come and pick them up, delay-sensitive data have to be sent to the data collector in its current location. We present a fully distributed routing scheme for such a configuration.

1.3 Document outline

The rest of this document is organized as follows. Chapter 2 presents the relevant background material and surveys previous related work. In Chapter 3, our optimization scheme for delay-tolerant placement and routing in terrestrial WSNs is presented. This includes discretizing the search space of data collectors locations and formulating the problem as a MILP. We present our optimization schemes for delay-tolerant and delay-sensitive placement and routing in underwater WSNs in Chapter 4. We show, in Chapter 4, how to extend the search space discretizing algorithm to handle multiple transmission ranges, delay attributes, and a 3D environment. We also show a linear programming formulation that accommodates delay constraints. In Chapter 5, we present our distributed scheme for routing to a mobile data collector. Finally, Chapter 6 concludes this document by highlighting the main issues in this thesis and outlining some future research directions.

Chapter 2

Background

This chapter presents the background material and surveys previous research related to the work in this thesis. It starts with an introduction to WSNs and their potential applications in Section 2.1. We overview some energy-aware protocols in WSNs in Section 2.2. In Section 2.3, previous work towards extending the network lifetime is classified into two main categories and the main schemes in each category are highlighted. The NP-completeness of the general data collector placement problem is shown in section 2.4. A brief summary is given in Section 2.5.

2.1 Wireless sensor networks

A WSN is composed of a large number of tiny sensor nodes and one or multiple more powerful sink nodes. Sensor nodes collect data from the surrounding environment and deliver the collected data to a sink node. Each sensor node has one or more sensors, a general-purpose Central Processing Unit (CPU) to perform arithmetic and logical operations, and a small amount of storage space. The power is supplied to these untethered sensor nodes through small, non-replenishable batteries. A sensor node has a wireless communication interface through which it can communicate with other nodes in its vicinity. Due to the scarcity of the power reservoir and due to the fact that communication is the dominant power consumer in a sensor node, the transmission range of these devices is limited for power efficiency purposes. Sensor nodes, which are spatially distant from the sink node, can report their data in a multi-hop fashion. A sink node is usually a more powerful device with a virtually unlimited power supply.

Crossbow MICAz motes [1], as an example of sensor nodes, have a low-power microcontroller that is in charge of regular processing, 128 KB of memory, and 512 KB of flash memory. MICAz motes have an IEEE 802.15.4 [9] compatible Radio frequency transceiver with a transmission rate of 250 kbps. A MICAz node has an outdoor transmission range of up to 100 m and an indoor transmission range of up to 30 m. Commercially available MICAz motes have 2X AA batteries which are enough for a week of full load [10]. MICAz mote is shown in Fig. 2.1.

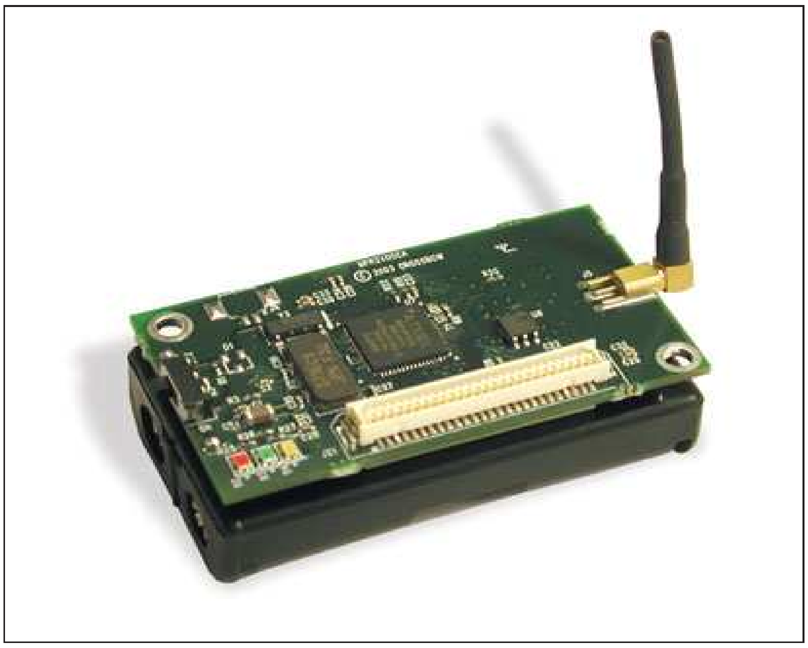


Figure 2.1: Crossbow MICAz node [1].

WSNs have a wide range of potential applications related to scientific, environmental, industrial, and military monitoring. WSNs have been deployed to monitor the seabird nesting environment and behavior in the Great Duck Island [11]. The ZebraNet system is a WSN deployed to support the research of biologists to monitor animal migration in Kenya [12]. WSNs have enabled a new generation of smart environments; an example of such an environment is the Gator Tech smart house in Florida, which is designed to assist elderly and disabled residents [13]. A prototype has been proposed in [14] for using WSNs to monitor drinking water quality. The design and implementation for using WSNs to monitor soil moisture is presented in [15]. An approach for using WSNs in monitoring the health of civil infrastructures (e.g., bridges and highways) is presented in [16]. These are just a few examples of how WSNs can be used to collect important data and facilitate significant services in real life.

2.2 Energy-aware protocols for WSNs

Energy awareness is an essential design issue for WSN communication protocols. In this section, we show how energy is conserved in some energy-aware MAC and routing protocols.

2.2.1 Energy-aware MAC protocols

In communications, a MAC protocol provides a mechanism to control the access of multiple stations (nodes) to a common physical communication channel. According to [17], the major causes of energy waste at the MAC level in WSNs are:

1. Collisions: a node receives more than one packet at the same time.
2. Overhearing: a node receives a packet which is not destined to it.
3. Idle listening: the receiver of a packet is active, yet no packets are being transmitted to it.
4. Overemitting: a node transmits a packet to a destination that is not ready.

Several energy-efficient MAC protocols have been proposed to alleviate the effect of these factors. Collisions are not specific to WSNs and can be dealt with using standard access control mechanisms. TRAMA [18] is a MAC protocol for WSNs that uses Time

Division Multiple Access (TDMA) to deal with collisions. WISEMAC [19] and S-MAC [20] are two protocols that use Carrier Sense (CS). Overhearing and idle listening are alleviated using sleep schedules which allow sensor nodes to turn off their wireless transceiver during sleep periods [21]. To avoid overemitting, a node wishing to communicate with another node initiates their dialogue by sending handshake control packets. The Request to Send / Clear to Send (RTS/CTS) are examples of such control packets which are used in the S-MAC protocol.

2.2.2 Energy-aware routing protocols

Routing is the process of finding paths between pairs of nodes in a network. In energy-aware routing, the quality of a path is estimated based on its effect on the energy reserves of nodes in the network. Tens of energy-aware routing protocols for WSNs have been reported during the last decade [22][23]. These protocols are classified into two categories: proactive and reactive. Proactive protocols are those in which paths are computed in advance. Reactive protocols are those in which a path is discovered only when it is needed. Because reactive protocols incurs high communication overhead for path discovery and setup, it is desirable to use proactive protocols in WSNs [22]. However, some geographic routing protocols (see Section 2.2.3), which are classified as reactive protocols, are able to find routing paths on-the-fly owing to the information nodes have about their geographic locations.

The majority of proactive routing protocols are based on constructing a tree rooted at the sink node [24][25][26]. Once such a tree is built, nodes deliver their data to the sink in a multi-hop fashion by passing them to their parents. The main idea of these protocols can be described as follows. At the deployment stage of the network, the sink initiates the tree construction process by broadcasting a tree construction packet. Each sensor node selects a sensor node, from which it has received the tree construction packet, to be its parent. When a node joins the tree, it rebroadcasts the tree construction packet to its neighbors. The selection of a parent can be made through different policies. An energy-efficient policy could be choosing the node that is closer to the root of the tree in terms of number of hops. Such a policy may reduce the number of transmissions needed to deliver data to the sink.

When the data generation rate of each sensor node is known, linear programming can be used to find the optimal routing in WSNs. In [27], the problem of finding the routing paths that maximize the network lifetime is formulated as a linear program. Such a formulation models a WSN as a flow network whose flow goes from sensor nodes to the sink node. The authors of [28] study the routing problem with splittable and unsplittable traffic. They present two linear programs for minimizing the total consumed energy in each problem.

Hierarchical routing is a class of routing protools in which sensor nodes are partitioned into clusters. Each cluster has a cluster head and several cluster members. Members of a cluster report their data to the cluster head where data is aggregated and then delivered to the sink. Processing data at cluster heads saves significant amount of energy by removing redundant data, compressing data by computing aggregate functions (e.g., average, maximum, minimum, ... etc), and, thereby, reducing the number of transmissions. The Low Energy Adaptive Clustering Hierarchy (LEACH) is a clustering protocol that utilizes randomized rotation of cluster heads to distribute the energy load among sensor nodes in the network [4]. In LEACH, the system operates in rounds, where each round begins with cluster heads selection followed by steady data transmission. During the cluster heads selection phase, each node n_i generates a threshold $T(i)$ as follows:

$$T(i) = \begin{cases} \dfrac{P}{1 - P * (r \mod \frac{1}{P})} & \text{if } i \in H \\ 0 & \text{otherwise} \end{cases}$$

where P is the desired percentage of cluster heads, r is the current round, and H is the set of sensor nodes that have not been chosen to be cluster heads during the last $\frac{1}{P}$ rounds. At the beginning of each round, each sensor node n_i generates a random number between 0 and 1, and elects itself to be a cluster head if this random number is less than $T(i)$. Cluster heads broadcast advertisement messages to all other nodes. Other nodes listen to advertisement messages, decide on the cluster they wish to belong to, and inform the head of the chosen cluster about their decision. Cluster heads then create transmission schedules to their clusters and start receiving data from members of their clusters. Once a cluster head receives data from all members in its cluster, it performs the required data aggregation and processing, and sends the compressed data directly to the sink. While LEACH represents a clustering protocol that evenly distributes the energy load amongst all nodes, it assumes that all nodes are able to communicate directly with the sink, which does not scale to large-size networks.

The Hybrid, Energy Efficient, Distributed (HEED) clustering [29] differs from LEACH in that it considers the residual energy of different nodes in the selection of cluster heads; nodes with higher residual energy are more likely to be chosen as cluster heads. Moreover, HEED makes use of multi-hop intercluster communication. HEED constructs a connected multi-hop cluster head graph to facilitate this intercluster communication.

2.2.3 Geographic routing

We use geographic routing in parts of the scheme we present in Chapter 5. In this subsection, we give the background material necessary to understand those parts. In geographic routing, it is assumed that sensor nodes know their geographic locations and locations of their immediate neighbors. A node can obtain its location using a Global Positioning System (GPS) or using one of the less expensive localization methods described in [30][31][32]. A routing node forwards a packet based on the geographic location of its destination and locations of immediate neighbors. Geographic routing algorithms can make efficient routing decisions based on local information, i.e., avoiding the overhead of maintaining global topology information. This makes geographic routing suitable for resource constrained WSNs.

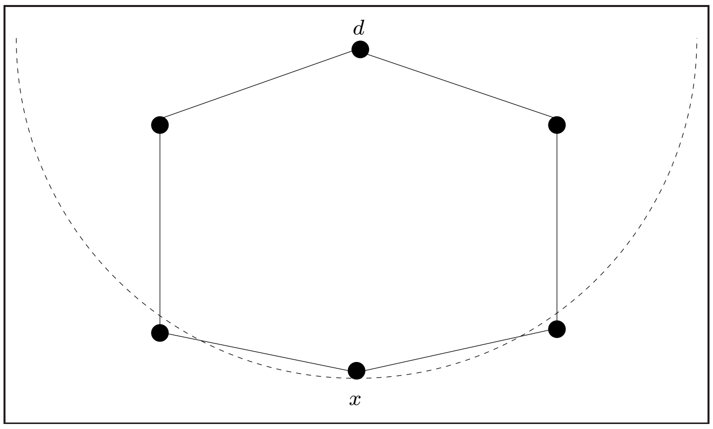


Figure 2.2: Using greedy distance routing, a packet destined to d may get stuck at x.

Greedy algorithms

Because maintaining global information is hard to achieve in WSNs, *locality* is a desired property for any distributed algorithm designed for WSNs. A distributed algorithm is localized if each node makes its decision based on information it has about itself, information about its immediate neighbors, and probably a constant size global information. *Greedy distance routing* [33] is a localized geographic routing protocol in which a routing node forwards a packet to its immediate neighbor which is the closest to the destination. A similar greedy algorithm is *compass routing* [34] in which a routing node x forwards the packet to its immediate neighbor y which minimizes the angle $\angle dxy$, where d is the destination node [35]. While simple, these two algorithms do not guarantee packet delivery; a packet may get stuck at a local minimum node at which no progress can be made towards the destination as shown in the example of Fig. 2.2. Only under global assumptions on the connectivity graph can these algorithms guarantee packet delivery. The greedy distance routing, for instance, guarantees packet delivery if the connectivity graph contains the *Delaunay triangulation* [36] of the sensor nodes. Compass routing guarantees delivery if the connectivity graph is the Delaunay triangulation of the sensor nodes. These assumptions are hard to meet in practice.

Planarization of the connectivity graph

Surprisingly, removing some edges from the connectivity graph is a condition for some routing algorithms to guarantee packet delivery, i.e., it does not get stuck at a local minimum node nor does it get trapped in a cycle. Some routing algorithms guarantee packet delivery if the connectivity graph is planar. The idea is to have a planar subgraph of the connectivity graph, and to apply one of these algorithms to provide guaranteed packet delivery. Fortunately, there are simple localized algorithms for constructing a planar subgraph from an arbitrary connectivity graph. The *Relative Neighborhood Graph* (RNG) [37] and the *Gabriel Graph* (GG) [38] are examples of planar graphs that can be constructed by localized algorithms. The RNG preserves an edge xy if the intersection of the two circles centered at x and y with a radius of the distance between x and y is free of other nodes. The GG preserves an edge xy if the circle, whose diameter is the line segment xy, is free of other nodes. *Perimeter routing* [34] is an example of a routing algorithm that guarantees delivery on planar connectivity graphs. In perimeter routing, when a packet destined to a node d is currently at a node s, the routing process starts with the face beyond s along the segment sd. Note that sd may intersect with one or more boundary segments of that face. The packet traverses the boundary segments of that face and gets back to s. The packet keeps with it the boundary segment kl that intersects with sd at the point furthest from s. When the packet gets back to s, it is sent to k. This process continues until a face that contains d is reached. If all faces of the planar graph are convex, sd intersects with only one boundary segment kl. So there is no need to traverse all boundary segments; the packet can, rather, walk along the boundaries (either clockwise or counterclockwise) until it reaches a segment that intersects with sd, at which point it switches to the next face as shown in Fig. 2.3.

Greedy perimeter stateless routing

Greedy geographic routing algorithms are simple but do not guarantee packet delivery. Other algorithms, such as perimeter routing, guarantee packet delivery but require a

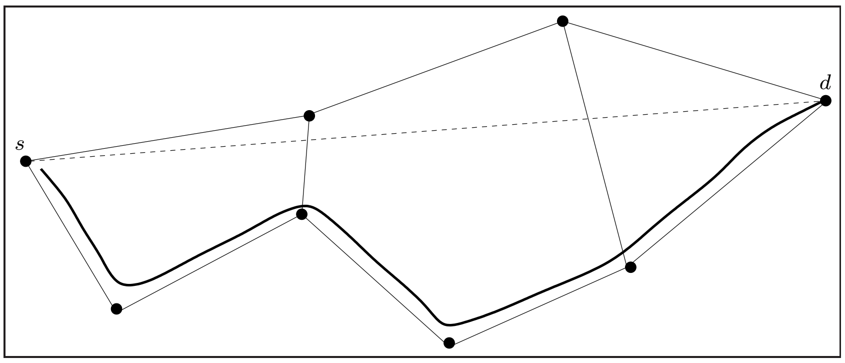


Figure 2.3: Perimeter routing on a convex-faces planar graph.

planar version of the connectivity graph, and use more complex routing algorithms. An interesting observation about WSNs is that they are anticipated to have a huge number of sensor nodes uniformly distributed over a sensing field. Dense deployment makes greedy algorithms work most of the time. However, there might be some holes in the topology of the network which results in some local minimum if a greedy method is used. *Greedy Perimeter Stateless Routing* (GPSR) [39] makes use of this interesting observation. GPSR combines a greedy method on the connectivity graph G and a perimeter method on a planar subgraph G'. GPSR starts with the greedy mode on G. During the greedy mode, each node forwards the packet to the neighbor which is the closest to the destination. If the packet arrives to a local minimum node, GPSR switches to the perimeter mode on G'. During the perimeter mode, the packet transitions to faces of G' closer to the destination. Eventually, the packet will reach a node closer to the destination than the node where the perimeter mode began, and will switch back to the greedy mode. This process continues until the packet reaches its destination. If there is no sensor node at the destination location, the packet will reach the face that contains that location and will be delivered to a sensor node along the perimeter of that face which is the closest to the destination location.

2.3 Device deployment and planning in WSNs

A critical performance measure for WSNs is the network lifetime. A significant amount of research has been done towards prolonging the lifetime of the network. The majority of such research targets energy-efficient routing and MAC protocols [21][20][17][4][23][22]. However, existing energy-aware routing and MAC protocols are not able to solve the topological problem of having bottlenecks around the sink and, hence, have a limited impact on the network lifetime. Accordingly, researchers in the field moved to device provisioning schemes in order to alleviate the effect of topological deficiencies in the network. Proposals in that direction can be classified into two streams: proposals with stationary devices and proposals with mobile devices. This section highlights proposals in these two streams.

2.3.1 Stationary devices

The use of stationary devices for load balancing in WSNs has been achieved through choosing the types, numbers, and locations of different devices in the network. One way to balance the load over the network is to deploy more nodes in areas closer to the sink. The authors in [40] proposed a nonuniform node distribution strategy that divides the sensing field into a number of coronas and gives the ratio in node densities between two consecutive coronas. The work in [41] is another attempt to alleviate the effect of having bottlenecks around the sink by making the density of sensor nodes in an area inversely proportional to its distance to the sink. One problem of this approach is the exponential growth of the total number of sensor nodes in the network. Moreover, with a fixed budget on the total number of sensor nodes, this approach results in a nonuniform sensing coverage over the sensing field.

The authors of [27] studied the problem of routing and placement of multiple sink nodes for maximizing the network lifetime and proposed an approximation algorithm to solve it. However, the time complexity of the proposed scheme is high; it involves solving a number of linear programming instances which is exponential in the number of sink nodes. The authors of [42] defined the following problem: given the power supply of each sensor node and a desired lifetime of the network, find the locations of R sink nodes and the routing paths from sensor nodes to sink nodes, such that the data rate is maximized. The data rate is the number of data units each sensor node sends to a sink node per time unit. When the possible locations of the sink nodes are limited to the exact locations of sensor nodes, the problem is shown to be NP-complete by a reduction from the problem of finding the minimum cardinality dominating set on unit disk graphs [43]. Two heuristic algorithms, a greedy algorithm and a local search algorithm, were proposed for this problem in [42]. It is important to notice that maximizing the data rate for a given lifetime is equivalent to maximizing the lifetime for a fixed data rate.

The work in [44] targets placing a single stationary sink node to collect data from a set of sensor nodes, with variable transmission ranges, when single-hop communication is used. It was shown that finding a sink location that maximizes the network lifetime is equivalent to finding the minimum radius disk that encloses all nodes. An optimal, linear-time algorithm for this problem was presented in [44].

A scheme for placing a number of data collectors to maximize the lifetime of the network and another scheme to find the minimum number of data collectors to meet a desired network lifetime are presented in [45]. Two genetic algorithms for placing a number of data collectors for the objective of minimizing the data collection latency are proposed in [46]. The work in [47] targets the problem of finding the minimum number of data collectors that can meet an upper bound on the maximum data collection latency. Another scheme for placing a number of data collectors in WSNs to achieve minimal latency is proposed in [48].

Some of the work in this literature uses a tier of Relay Nodes (RNs): a RN is a more powerful node that collects data from nearby sensor nodes and sends them to the sink. The Unequal Clustering Size (UCS) was proposed in [49]. In the UCS scheme, a number of RNs are deployed to serve as cluster heads. Each RN collects and aggregates data from sensor nodes in its cluster and sends the aggregated data to the sink. Sending data to the sink is made through multi-hop relaying; every RN forwards its aggregated data to a neighboring RN which is closer to the sink. As a result of this multi-hop relaying, RNs closer to the sink are assigned higher intercluster load than other RNs. The UCS scheme deals with this problem by making the cluster size proportional to the distance to the sink. Thereby, RNs with higher intercluster traffic are serving smaller clusters and, hence, are having a lower intracluster load.

The authors in [50] proposed an energy provisioning approach. The unbalanced load is alleviated by assigning RNs different amounts of energy and by deploying more RNs, if necessary. A heuristic based algorithm is presented to maximize the network lifetime under a certain energy and extra RNs budget. The work in [51] and [52] targets another variation of the problem: finding the minimum number of RNs and their locations to guarantee a desired network lifetime. The scheme presented in [51] is composed of two phases. In the first phase, some RNs are deployed to guarantee the connectivity of sensor nodes. In the second phase, extra RNs are deployed to connect the first phase RNs to the sink and to make all RNs meet the desired network lifetime. In [52], another best effort algorithm to place the second phase RNs is introduced and a theoretical lower bound on the number of the second phase RNs is derived. However, the use of RNs suffers a critical drawback: when sensor nodes use multi-hop relaying to reach the RNs, there will be a load balancing problem among sensor nodes within single clusters; and to make sensor nodes use single-hop transmission, a large number of RNs will be needed and a load balancing problem among RNs will arise.

2.3.2 Mobile devices

The main topological problem in WSNs stems from the fact that the load on a particular sensor node is determined by its distance to the sink: with multi-hop relaying, nodes that are one hop away from the sink relay data generated all over the network, and with single-hop communication, nodes distant from the sink deplete their energy faster than nodes near the sink. Such a problem is hard to alleviate when both the sink and the sensor nodes are stationary. While sensor nodes, which are envisioned to be small in size and low in cost, are hard to be made mobile, it is plausible to have a reasonable number of mobile data collectors to balance the load in the network and to prolong its lifetime. Recently, only few schemes that follow that stream have been proposed.

The work in [5] and [53] assumes the existence of a number of predefined spots where data collectors can be placed; the network lifetime is divided into equal length rounds and data collectors are moved to new locations at the beginning of each round. In [5], the network is modeled as a flow network and the problem of finding the optimal data collectors locations is formulated as a MILP whose objective is either minimizing the maximum energy spent by a single sensor node or minimizing the total consumed energy during the round. We argue that these two objective functions are not really suitable for the placement of mobile data collectors. This is because the optimal solution towards such objectives will not change over time and, hence, locations of data collectors will not be changed. A heuristic scheme, that considers residual energy, is proposed in [53]. However, locations of data collectors are chosen according to local information only: the decision of whether or not a data collector is placed at a given location is made based on the residual energy of sensor nodes that are one hop away from that location. While very simple, this scheme may obtain solutions which are not even close to the optimal one.

The authors in [8] proposed a heuristic for repositioning a data collector to extend the longevity of the network and to improve the data gathering timeliness. That work strives to answer questions about when and how the position of the data collector is changed. A metric of the traffic density and power consumption is used to monitor the status of each sensor node in the network. The decisions of whether or not to move the data collector and to which location it is moved are made based on the status of sensor nodes in the vicinity of the data collector. When such a relocation seems to be beneficial to the network, a local search is made to find a better position. Since the data collector is not moved far from its current location, changes to the routing paths are limited, which reduces the overhead of such a repositioning. In [8], another motive to change the location of the data collector is to improve the timeliness of the data gathering process for real-time applications. Data collector repositioning could be beneficial to meet delay-based Quality of Service (QoS) requirements; for example, getting closer to areas where real-time events are taking place could reduce the delay between the sources of these events and the data collector.

Recently, linear programming has been used to find the optimal routing and placement of a single mobile data collector [54][55]. The problem is defined as follows: given a set of predefined points where the data collector can be located, find the time the data collector stays at each location (i.e., the sojourn time) and the multi-hop routing paths from sensor nodes to the data collector at each location. This important result was found through a nice observation that the order different locations are being visited by the data collector does not affect the network lifetime; what really affects it is the sojourn time and the routing strategy associated with each location. However, this observation does not hold when more than one data collector are available and, hence, this scheme can not be extended to multiple data collectors.

The authors in [56] investigate the problem of uneven relaying load in WSNs. They compare three approaches: nonuniform node densities, multiple sink nodes, and a mobile sink node. Their results indicated that using multiple sink nodes is the most effective solution. Yet, it is the most expensive one. Repositioning the sink node has shown a good performance at a lower cost. However, nonuniform node densities was not recommended as it demands deploying a significant number of extra sensor nodes.

The use of mobile data collectors has been also mentioned in the context of underwater WSNs. The work in [57] involves the hardware and networking software design for using mobile data collectors in underwater WSNs. According to [57], each sensor node collects data and waits for the mobile data collector to become close enough to receive data in a single hop. The authors of [58] try to find the relationship between the number of data collectors and the time needed to harvest all data when data collectors follow random trajectories. Both the work in [57] and that in [58] assume single-hop communication and do not consider the problem of finding a path for the data collector. Moreover, both schemes are not suitable for delay sensitive applications.

2.4 NP-completeness of the data collectors placement problem

For the sake of completeness we show the NP-completeness proof presented in [42] with a minor modification to reflect the objective of maximizing the network lifetime rather than maximizing the data rate. We first make the following definitions:

Definition 1 *For a graph $G = (V, E)$, where V is the set of vertices and E is the set of edges, a dominating set is a subset D of V, such that for each vertex $v \in V$ either $v \in D$ or there exists a vertex $u \in D$ that shares an edge with v.*

Definition 2 *A graph G is a unit disk graph if and only if it has the following property: There is an edge between a pair of vertices if and only if their separation is at most 1 distance unit.*

Definition 3 *The Minimum Dominating Set (MDS) problem on unit disk graphs:*
Input: A unit disk graph G.
Question: Is there a dominating set of size d.

Definition 4 *The Data Collector Placement (DCP) problem:*
Input: Locations of sensor nodes loc_0, loc_1, ..., loc_{N-1}; energy supplies of sensor nodes E_0, E_1, ..., E_{N-1}; a data rate ρ data units/time unit from every sensor node; a transmission range r distance units; and a number of data collectors R.
Question: Is there a placement of R data collectors at sensor nodes locations and a multi-hop routing strategy that keep all sensor nodes alive for at least k time units.

Two sensor nodes are said to be neighbors if their separation is at most r distance units. Sending a data unit consumes one energy unit, and a sensor node is considered to be alive as long as its residual energy is greater than zero.

Now we state the following theorem.

Theorem 1 *The DCP problem is NP-complete.*

Proof

First, DCP is in NP:

A solution to the DCP problem is in the form of a subset $\mathcal{P}$ of R sensor nodes whose locations are chosen to place the data collectors at, and an assignment of flow values between neighboring sensor nodes. f_{ij}, which denotes the flow from a sensor node n_i to a neighboring sensor node n_j, is the total number of data units to be sent from n_i to n_j. Let F_i^+ denote the sum of the outgoing flow from sensor node n_i, and let F_i^- denote the sum of the incoming flow to sensor node n_i. To certify that a solution is feasible and has a lifetime of at least k time units, one needs to check the following conditions:

1. $\mathcal{P}$ has at most R sensor nodes.

2. If a sensor node n_i is in $\mathcal{P}$, $F_i^+/\rho \geq k$ and $F_i^+ \leq E_i$.

3. If a sensor node n_i is not in $\mathcal{P}$, $(F_i^+ - F_i^-)/\rho \geq k$ and $F_i^+ \leq E_i$.

Since these conditions can be checked in a polynomial time, the DCP is in NP.

Second, DCP is NP-hard:

The NP-hardness of the DCP problem can be shown through a reduction from the MDS problem, which has been shown to be NP-hard in [43]. An instance A of the MDS problem, on a graph G, can be reduced to an instance B of the DCP problem in which there is a sensor node for each vertex in G, every sensor node has a residual energy of 1 energy unit, r is set to 1 distance unit, R is set to d, and ρ is set to 1 data unit/time unit. There exists a dominating set of size at most d for A if and only if there is a placement for the data collectors and a routing strategy with a lifetime of at least 1 time unit for B. This can be shown as follows.

If there is a dominating set of size at most d for A, we can place a data collector at the location of each vertex in the dominating set. Then, by the definition of a dominating set, each sensor node will have a data collector either in its location or at the location of one of its neighbors. Thus, each sensor node can send one data unit directly to a data collector which results in a lifetime of 1 time unit for B.

Now assume that there is a placement for the d data collectors and a routing strategy with a lifetime of at least 1 time unit for B. The data collectors are placed at the locations of a set $\mathcal{P}$ of d sensor nodes. Since each sensor node has only 1 energy unit (which is just enough to send its own data), no sensor node is able to relay a single data unit for any other sensor nodes. Therefore, each sensor node will have a data collector placed not more than 1 distance unit away from it. Since data collectors are placed at sensor nodes locations only, each sensor node must have a data collector placed at its own location or at the location of one of its neighbors. Thus, the set $\mathcal{P}$ form a dominating set for the underlying connectivity graph of the sensor network and, hence, a dominating set of size d for B.

Therefore, the DCP problem is NP-complete. $\square$

When the data collectors are allowed to be placed at any point, rather than at the locations of sensor nodes only, the problem seems to be harder. However, we are not aware of any formal proof for the NP-hardness of that problem.

2.5 Summary

This chapter surveys existing proposals for handling the bottleneck problems in WSNs. These proposals are classified into two categories: proposals with stationary devices and proposals with mobile devices. Proposals in the former category use nonuniform node distribution [40][41], multiple RNs with and without variable energy provisioning

	Data collectors					
	Multiple	Mobile	Objective	Optimal	Multi-hop relaying	Technique
[27]	Yes	No	Lifetime	No	Yes	Linear programming
[45]	Yes	No	Lifetime	No	Yes	Clustering
[42]	Yes	No	Maximum data rate	No	Yes	Greedy algorithm
[44]	No	No	Lifetime	Yes	No	Computational geometry
[5]	Yes	Yes	Lifetime	No	Yes	Linear programming
[53]	Yes	Yes	Lifetime	No	Yes	Heuristic
[8]	No	Yes	Lifetime/ timeliness	No	Yes	Local search
[54][55]	No	Yes	Lifetime	Yes	Yes	Linear programming

Table 2.1: Data collector placement schemes.

[49][50][51][52], or multiple stationary data collectors [27][42]. Proposals in the later category exploit mobility of data collectors to distribute the load over the network and alleviate the bottleneck problem [5][53][8][54][55]. Table 2.1 gives a comparison between different data collector placement schemes discussed in this chapter. Optimal polynomial-time schemes exist only for special settings and assumptions: there is a linear-time optimal algorithm to place a single stationary data collector in a single-hop network with a variable transmission range [44], and there is a linear programming approach to find the optimal routing and placement of a single mobile data collector in a multi-hop network [54][55]. The problem of placing more than one mobile data collectors in a multi-hop network is proven to be NP-complete when data collector locations are limited to the exact locations of sensor nodes [42]. The problem does not seem to be less hard when data collectors can be placed anywhere in the sensing field, yet no formal NP-hardness proof exists in the literature.

Chapter 3

Mobile data collectors in terrestrial WSNs

A major challenge affecting the lifetime of WSNs comes from the unbalanced energy consumption over different parts of the network. In this chapter, we present a mobile data collector placement scheme for extending the lifetime of the network. The lifetime of the network is divided into rounds and data collectors are moved to new locations at the beginning of each round. We define and solve two problems: the on-track placement where data collectors can be placed only along predefined tracks (roads) spanning the sensing field, and the general placement where data collectors may be placed at any point in the sensing field.

We formulate the problems as MILPs and use a linear programming solver (with a constant time limit) to find near-optimal placements of the data collectors and to find routing paths to deliver data to data collectors. Our experiments show that our approach makes a significant extension to the lifetime of the network.

The rest of this chapter is organized as follows. Section 3.1 outlines the proposed approach and highlights its contributions. Section 3.2 describes the model of the system and gives a formal problem definition. In Section 3.3, we present our placement schemes. Section 3.4 shows the experimental results. Finally, Section 3.5 concludes this chapter with a brief summary.

3.1 Schemes outline and contributions

We argue for using multiple mobile data collectors and propose an approach for placing these data collectors in a way that balances the energy expenditure and increases the lifetime of the network. Our approach divides the lifetime of the network into fixed length rounds (e.g., hours, days, or weeks) and moves the data collectors to new locations at the beginning of each round. Our policy of maximizing the lifetime is to maximize the minimum residual energy at the end of each round. Some recently proposed schemes have addressed the issue of mobile sinks. However, some of that work assumes the existence of a set of predefined spots (i.e., points) where data collectors may be placed [5], and some is limited to placing data collectors at the boundaries of the field [53]. To this end, the novel contribution of this work is twofold:

1. We define and solve two placement and routing problems. The first one assumes the existence of predefined tracks (e.g., a road network) spanning the sensing field, and data collectors can be moved over and placed at any point along these tracks. This would be practical in a situation where data collectors are carried on Autonomous Unmanned Vehicles (AUVs) or robots that move along paved roads only. In the second one, a data collector can be placed anywhere in the sensing field.

2. We discretize the search space of data collector locations without affecting the quality of the derived solutions: we devise an algorithm that finds a finite set of relatively small number of points, and we prove the existence of an optimal placement in which each data collector is placed at a point in that set. Since the problem is modeled as a MILP, making the cardinality of this set as small as possible would significantly improve the efficiency and the solution quality.

By formulating the problems as a MILP, an optimal solution can be found. However, this may require an exponential time in the worst case [59]. Therefore, we impose a time

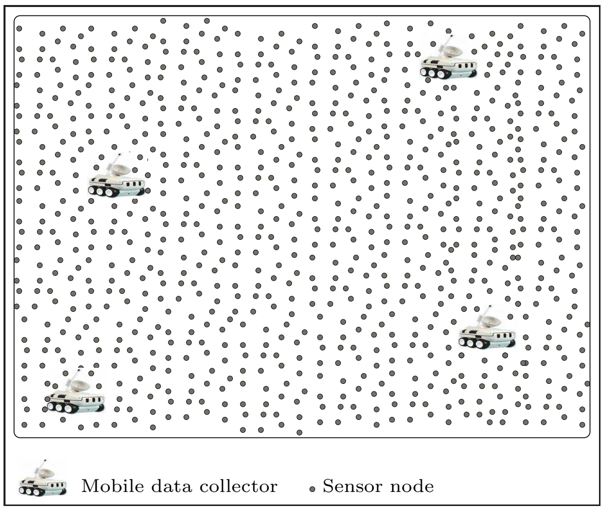


Figure 3.1: A WSN with four mobile data collectors.

limit on a branch-and-bound solver in order to find near-optimal solutions in reasonable time.

3.2 System model and problem definition

We consider a WSN consisting of N sensor nodes and R data collectors. Fig. 3.1 shows an example of such a network. Each sensor node collects data from the surrounding environment and sends the collected data to one of the data collectors. The transmission range, which is modeled as a disk, of all sensor nodes is fixed to r distance units. The topology of the network is modeled as a graph $G = (V, E)$, where $V = \{n_0, n_1, ..., n_{N-1}\}$ is the set of N sensor nodes, and $(i, j) \in E$, if sensor nodes n_i and n_j are within the transmission range of each other (we say n_i and n_j are neighbors). Each sensor node n_i has a data generation rate G_i; G_i is the number of data units generated by node n_i per time unit (without loss of generality, we assume that the round is our time unit).

Our schemes are independent of the underlying MAC protocol. We assume a capacity limit for each sensor node which limits the number of data units that can be transmitted by a sensor node during one round. The capacity of a sensor node n_i is denoted by C_i. This parameter can be adjusted to comply with any constraint imposed by any MAC protocol.

We define the lifetime of the network as the time until the first sensor node dies. Yet other definitions (e.g., the time until a particular proportion of the sensors die) can be equally used in our approach.

We also consider a set of tracks in the sensing field; these tracks are modeled as a set of L line segments $\{l_0, l_1, ..., l_{L-1}\}$.

Assumptions

1. For each sensor node n_i, its location loc_i, its residual energy E_i, and its data generation rate G_i are known. The location of a sensor node can be known a priori during a deterministic deployment or can be obtained after deployment using GPS or other GPS-less schemes [31][60]. The values of E_i and G_i, which may be estimated at the node itself, can be exchanged through control packets or piggybacked with data packets.

2. With the fixed transmission range, the energy consumed to send one data unit and that consumed to receive one data unit are constants. In other words, the exact distance between two neighboring nodes has no impact on the energy consumed for exchanging data between the two.

3. Data collectors are not energy constrained (as they can be recharged easily).

4. Every generated data unit is sent to any data collector (not to all of them).

Problem definition

The placement and routing problem can be described as follows:

> The lifetime of the network is divided into equal length rounds. At the beginning of each round, find the optimal locations of R data collectors together

with the routing paths to deliver the generated data to the data collectors, which maximize the minimum residual energy at the end of the round.

In the on-track placement problem, data collectors can be placed only at points along the tracks. In the general placement problem, data collectors may be placed anywhere in the sensing field.

3.3 Data collector placement scheme

A major challenge in this placement problem is the infinite search space for data collector locations. The first step in our approach is to make the search space finite without affecting the quality of the solution. We first make the following definition.

Definition 5 *A finite set of points $\mathcal{P}$ is complete if and only if it satisfies the following property:*
There exists an optimal[1] *placement of data collectors in which each data collector is placed at a point in $\mathcal{P}$.*

Finding a complete set would make the placement problem a discrete optimization problem rather than a continuous one. In the following two subsections we show how to find complete sets for the on-track and the general placement problems.

3.3.1 A complete set for the on-track placement problem

For the on-track placement problem, a data collector may be placed at any point on any track as long as it is within the transmission range of at least one sensor node. This results in an infinite number of possible locations for each data collector and, hence, an infinite search space. Recall that a track is modeled as a line segment; we will use the terms line segment and track interchangeably. To explain our method of discretizing the search space for the on-track problem, we make the following definitions.

[1]Optimality can be defined according to any energy-based objective function (e.g., total consumed energy, maximum amount of energy consumed by a single node, network lifetime, ... etc).

Definition 6 *For any line segment l, connecting two points s and t, a critical point on l is either s, t, or a point where the line segment l and the boundary of the transmission disk of a sensor node intersect.*

Definition 7 *For any line segment l, an overlapping segment α is either:*

- *a line segment l' connecting two consecutive critical points on l, or*

- *an intersection point of l and a transmission disk for which l is a tangent.*

Let $S(\alpha)$ denote the subset of sensor nodes whose transmission disks overlap with l at α.

In Fig. 3.2, the overlapping segments of the track st with respect to sensor nodes n_1, n_2, n_3, n_4, and n_5 are sa, ab, bc, cd, de, ef, ff (i.e., the point f), fg, and gt. $S(sa) = \{n_1\}$, $S(ab) = \phi$, $S(bc) = \{n_2\}$, $S(cd) = \{n_2, n_3\}$, $S(de) = \{n_3\}$, $S(ef) = \phi$, $S(ff) = \{n_4\}$, $S(fg) = \phi$, and $S(gt) = \{n_5\}$.

Definition 8 *For any line segment l, an overlapping segment α is maximal if there is no overlapping segment β on l, where $S(\alpha) \subset S(\beta)$.*

In Fig. 3.2, the overlapping segments sa, cd, ff, and gt are maximal. Now we state the following lemma.

Lemma 1 *For every overlapping segment β, there exists a maximal overlapping segment α, such that $S(\beta) \subseteq S(\alpha)$.*

Proof If β is maximal, we choose α to be β itself. If β is not maximal then, by definition, there exists an overlapping segment α_1 such that $S(\beta) \subset S(\alpha_1)$. If α_1 is maximal, we choose α to be α_1, and if α_1 is not maximal then, by definition, there exists an overlapping segment α_2 such that $S(\alpha_1) \subset S(\alpha_2)$. This process continues until a maximal overlapping segment α_z is found; we choose α to be α_z. It is obvious that $|S(\alpha_z)| \leq N$, and $|S(\beta)| < |S(\alpha_1)| < |S(\alpha_2)| < ... < |S(\alpha_z)| \leq N$. Therefore, the process of finding the maximal overlapping segment α_z takes a finite number of steps which is at most $N - 1$. $\square$

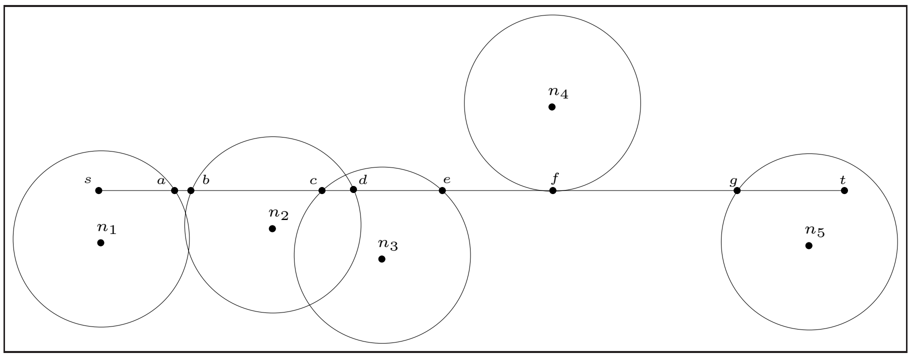


Figure 3.2: Intersection points on a line segment.

Then, we deduce the following theorem.

Theorem 2 *A set $\mathcal{P}$ that contains one point from every maximal overlapping segment is complete.*

Proof To prove this theorem, it suffices to show that for any arbitrary placement $\aleph$ we can construct an equivalent placement $\aleph^*$ in which every data collector is placed at a point in $\mathcal{P}$. Let us say that in $\aleph$, a data collector B is placed such that it is within the transmission range of a subset of sensor nodes H. It is obvious that there exists an overlapping segment β, such that $H \subseteq S(\beta)$. From the previous lemma, there exists a maximal overlapping segment α, such that $S(\beta) \subseteq S(\alpha)$. In $\aleph^*$, we place B at the point in $\mathcal{P}$ that belongs to α, so that B is placed at a point in $\mathcal{P}$ and is still within the transmission range of all sensor nodes in H. Repeating for all data collectors, we construct a placement $\aleph^*$ which is equivalent to the placement $\aleph$. $\square$

Before we proceed to the algorithm that finds all maximal overlapping segments, we define the notations of *entry* points, *exit* points, and *tangent* points. Each line segment (i.e., track) has a set of intersection points; an intersection point is a point where the line segment intersects with the boundary of the transmission disk of a sensor node. In Fig. 3.2, the line segment st has 7 intersection points: a, b, c, d, e, f, and g. By walking over a line segment from left to right (and from down to up if the line segment is parallel to the y-axis), intersection points incident to that line segment can be classified into entry points, exit points, and tangent points. An entry point is one at which the intersection

Algorithm 1: Finding all maximal overlapping segments.

```
Procedure FindMaximalOverlappingSegments( )
Output: A set $\mathcal{P}$ that contains one point from every maximal overlapping
        segment.
$\mathcal{P}$:=$\phi$ ;
foreach track $l$ do
    Find all intersection points incident to $l$;
    Sort all intersection points incident to $l$ in a non-decreasing lexicographic order
    according to their ($x$-coordinate,$y$-coordinate);
    \\ i.e., $(x_1, y_1) \leq (x_2, y_2)$ if and only if $x_1 < x_2$ or
    \\ ($x_1 = x_2$ and $y_1 \leq y_2$).
    Active:=True;
    foreach intersection point $p$ incident to $l$ do
        if (Active AND $p$ is an exit point) OR $p$ is a tangent point then
            $\mathcal{P}$:=$\mathcal{P} \cup \{p\}$;
            Active=False;
        end
        if $p$ is an entry point then
            Active=True;
        end
    end
    if Active then
        $p$:= the last intersection point incident to $l$;
        $\mathcal{P}$:=$\mathcal{P} \cup \{p\}$;
    end
end
```

of a line segment and a transmission disk begins. An exit point is one at which the intersection of a line segment and a transmission disk ends. A tangent point occurs if the line segment and a transmission disk intersect at exactly one point. In Fig. 3.2, b, c, and g are entry points; a, d, and e are exit points; and f is a tangent point. Now we state the following obvious observation without a proof.

Observation 1 A maximal overlapping segment on a line segment st can only take one of the following forms:

1. A tangent point (e.g., the point f in Fig. 3.2),
2. If an entry point is followed by an exit point, the line segment between the two is a maximal overlapping segment (e.g., the line segment cd in Fig. 3.2),

3. If the first intersection point is an exit point, the line segment between s and the first intersection point is a maximal overlapping segment (e.g., the line segment sa in Fig. 3.2), or

4. If the last intersection point is an entry point, the line segment between the last intersection point and t is a maximal overlapping segment (e.g., the segment gt in Fig. 3.2).

Algorithm 1 uses Observation 1 to construct a complete set $\mathcal{P}$ that contains exactly one point from every maximal overlapping segment. This algorithm runs in $O(L\ N \log N)$ time, where L is the number of tracks and N is the number of sensor nodes.

3.3.2 A complete set for the general placement problem

In the general placement problem a data collector can be placed at any point in the sensing field as long as it is within the transmission range of at least one sensor node. To show our method of constructing a complete set for the general placement problem, we use some notations similar to those used in the previous subsection. These notations are derived from the arrangement of transmission disks which partitions the plane into one open face and several closed faces (see Fig. 3.3). Closed faces are bounded by portions (arcs) of the boundaries of one or more transmission disks.

Definition 9 *An overlapping region α is either:*

- *a tangent point*[2] *or*

- *a closed face in the arrangement of transmission disks.*

Let $S(\alpha)$ denote the subset of sensor nodes whose transmission disks overlap at α.

Definition 10 *An overlapping region α is maximal if there is no overlapping region β, where $S(\alpha) \subset S(\beta)$.*

[2]When two transmission disks overlap at a single point only, this point is called a tangent point.

Fig. 3.3 shows six sensor nodes and their maximal overlapping regions (MORs). In Fig. 3.3, there are 15 overlapping regions (closed faces) and 4 of them are maximal.

We next show that a complete set can be derived from the set of MORs. It is obvious that Lemma 1 and Theorem 2 can be applied to the general placement problem as follows.

Lemma 2 *For every overlapping region β, there exists a MOR α, such that $S(\beta) \subseteq S(\alpha)$.*

Theorem 3 *A set $\mathcal{P}$ that contains one point from every MOR is complete.*

The proofs of the above lemma and theorem can be directly derived from those of Lemma 1 and Theorem 2. One just needs to observe the analogy between maximal overlapping regions here and maximal overlapping segments in Lemma 1 and Theorem 2.

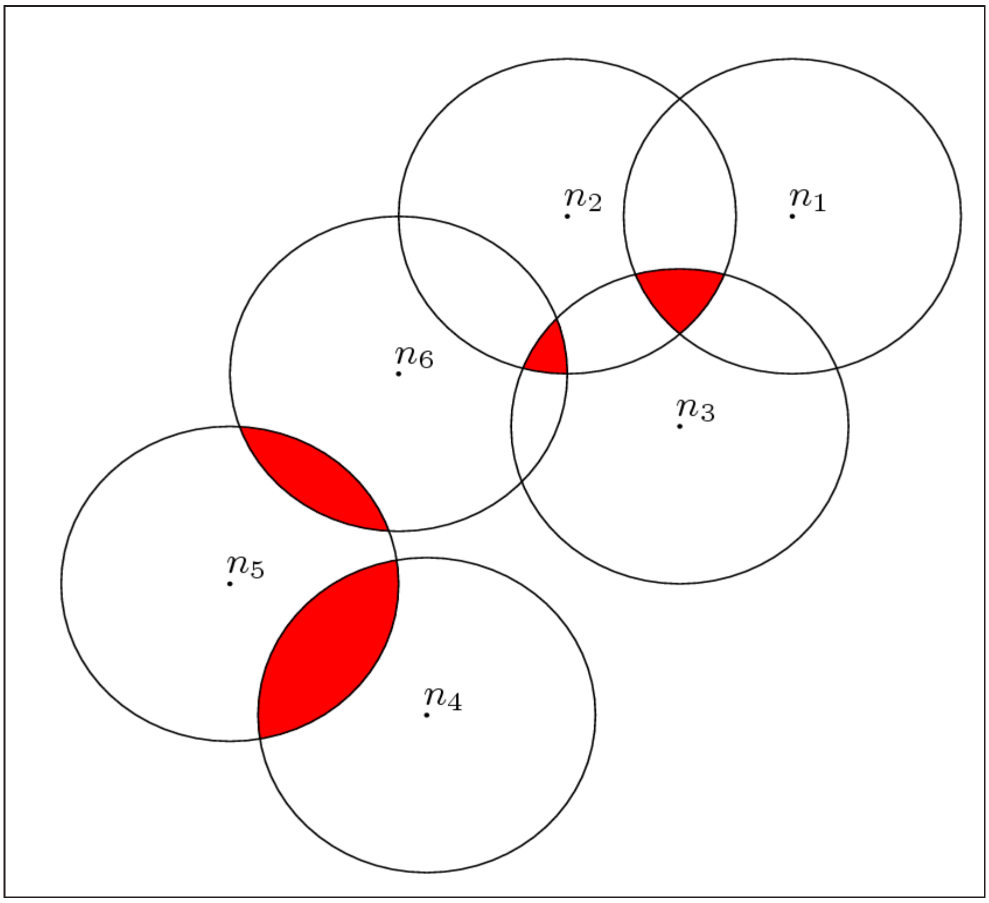


Figure 3.3: Maximal overlapping regions.

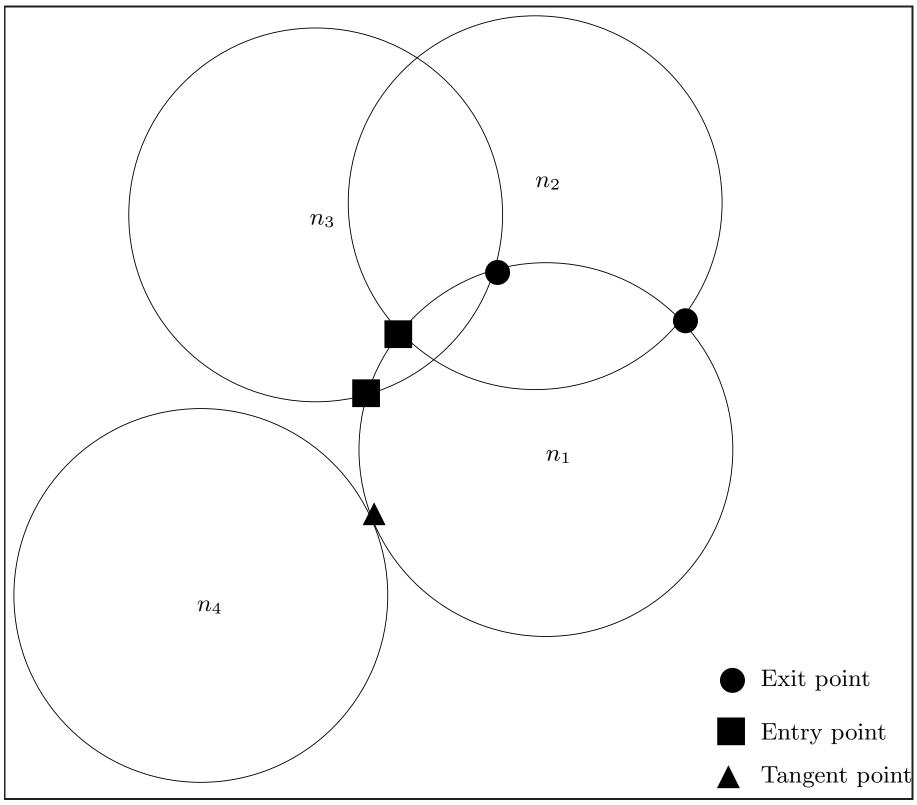
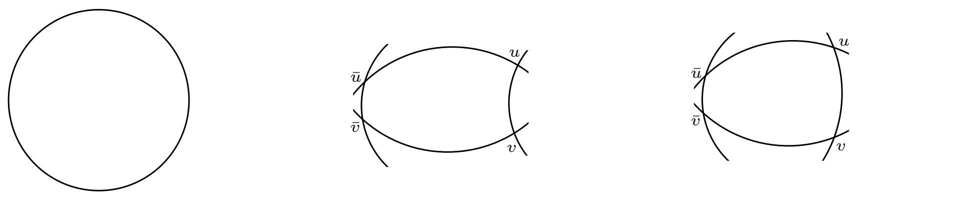


Figure 3.4: Entry points, exit points, and tangent points.

In order to find all MORs, we need to find the arrangement of transmission disks. Let $D(i)$ denote the transmission disk of sensor node n_i, and if p is an intersection point of the boundaries of $D(i)$ and $D(j)$, let $succ_i(p)$ ($succ_j(p)$) denote the intersection point incident to $D(i)$ ($D(j)$) that comes right after p according to a clockwise order, and let $other_i(p) = j$ and $other_j(p) = i$. The arrangement of disks is a data structure by which for any intersection point p of $D(i)$ and $D(j)$, we can get $succ_i(p)$ and $other_i(p)$ in $O(1)$ time. Such a data structure can be constructed by Algorithm 2, which runs in $O(N^2 \log N)$ time.

By walking over the boundary of a transmission disk $D(i)$ in a clockwise direction, intersection points incident to $D(i)$ can be classified into three groups: *entry* points, *exit* points, and *tangent* points. An entry point is one at which we enter the transmission disk of another sensor node. An exit point is one at which we leave the transmission disk of another sensor node. A tangent point occurs if two transmission disks intersect at exactly one point. Note that an intersection point of two disks is either a tangent point



(a) Single disk. (b) Concave overlapping region. (c) Convex overlapping region.

Figure 3.5: Different types of closed faces.

on both disks or an entry point with respect to one disk and an exit point with respect to the other disk. Fig. 3.4 gives an example of four sensor nodes (n_1,n_2, n_3, and n_4) where the intersection points incident to $D(1)$ are classified with respect to $D(1)$.

The arrangement of transmission disks divides the plane into one open face and several closed faces. Closed faces are of three types:

1. A disk that does not overlap with any other disks (see part (a) of Fig. 3.5).
2. A concave overlapping region, i.e., it has at least one concave arc in its boundaries (see part (b) of Fig. 3.5).
3. A convex ovelapping region, i.e., all arcs in its boundaries are convex (see part (c) of Fig. 3.5).

Now, it is straightforward to realize the following observation.

Observation 2 An overlapping region β is maximal if and only if it satisfies one of the following properties:

1. β is a tangent point.
2. β is a disk that does not overlap with any other disks.
3. β is a convex overlapping region whose boundaries do not have any tangent points.

Proof To prove this observation, we need to show that any of these three properties is a sufficient condition for an overlapping region to be maximal (i.e., the if direction) and that neglecting all of them is a sufficient condition for an overlapping region to be non-maximal (i.e., the only if direction).

The if direction:

First: Since the tangent point is the overlapping region of two disks that do not overlap anywhere else, a tangent point is a MOR.

Second: A disk that does not overlap with any other disks is a MOR. If it was not a MOR, that disk would overlap with at least one other disk which makes a contradiction.

Third: In order to show that β, which is a convex overlapping region whose boundaries do not have any tangent points, is maximal, it suffices to show that there is no point outside β that lies inside all disks in $S(\beta)$. Note that since there is no tangent points on the boundaries of β, points inside β and points on the boundaries of β are within the transmission range of the same subset of sensor nodes. So we just need to check points inside β against points outside it as follows. Assume, for the sake of contradiction, that there exists a point u outside β which is inside all disks in $S(\beta)$. Let us take another point v inside β. Since β is convex, the line uv must intersect with the boundaries of β at exactly one point. Let the line uv intersect with the boundaries of β at a point on an arc that is part of the perimeter of a disk $D(i)$. Now, either v is outside $D(i)$ which contradicts with the fact that β is convex or v is inside $D(i)$ and, thus, u is outside $D(i)$, which contradicts with the fact that u is inside all disks in $S(\beta)$ because $i \in S(\beta)$. Therefore, there is no point outside β which is inside all disks in $S(\beta)$ and, hence, β is maximal.

The only if direction:

When none of these three properties is satisfied, we end up with either a concave overlapping region or a convex overlapping region with a tangent point on its boundaries. We prove that both cases result in a non-maximal overlapping region. First, let β be a concave overlapping region whose concave arc belongs to a disk $D(i)$, and let α be the

Algorithm 2: Arrangement of transmission disks.

```
Procedure FindArrangement( )
Find all intersection points;
foreach sensor node $n_i$ do
    sort all intersection points incident to $D(i)$ in a clockwise order;
end
```

overlapping region on the other side of that concave arc. Obviously, $S(\alpha) = S(\beta) + \{i\}$ and, hence, $S(\beta) \subset S(\alpha)$ which makes β non-maximal. Second, let β be a convex overlapping region with a tangent point u on its boundaries. Then, u must be the intersection point of a disk $D(i)$ and another disk $D(j)$, such that $i \in S(\beta)$ and $j \notin S(\beta)$. Therefore, $S(u) = S(\beta) + \{j\}$ and, hence, $S(\beta) \subset S(u)$ which makes β non-maximal. □

To check whether an overlapping region β is convex or concave, we sort the intersection points on the boundaries of β in a clockwise order. Let uv be an arc on the boundaries of β as shown in Fig. 3.5 (b) or (c), let $\bar{u}$ be the point that comes before u on the boundaries of β according to a clockwise order, and let $\bar{v}$ be the point that comes after v on the boundaries of β according to a clockwise order. Let $\bar{u}u$ be a convex arc belonging to a disk $D(i)$. Obviously, the arc uv is convex if u is an exit point with respect to $D(i)$, and it is concave if u is an entry point with respect to $D(i)$. By applying this test to all arcs that make the boundaries of β, we can tell whether β is convex or concave.

Algorithm 3 uses Observation 2 to test whether an overlapping region is maximal or not. Algorithm 4 uses Algorithms 2 and 3 to find all MORs. Note that the Boolean array $Flag(p, i)$ is used to guarantee that we do not check the same overlapping region more than once. The overall complexity of Algorithm 4 is $O(N^2 \log N)$.

3.3.3 MILP formulation

When a complete set $\mathcal{P} = \{p_0, p_1, ..., p_{M-1}\}$ is found, the problem of finding the optimal locations of R data collectors and the flow paths from sensor nodes to data collectors can be formulated as a MILP. Note that once we obtain a complete set, the on-track and the general placement problems become the same; that is because in the two problems,

Algorithm 3: Testing whether an overlapping region is maximal or not.

```
**Function Maximal**($n_i$: a sensor node, $p$: a point)
**Input**: A sensor node $n_i$ and an intersection point $p$ incident to $D(i)$.
**Output**: True if a MOR is found, and False otherwise.
if $Flag(p,i) = 1$ OR $p$ is an entry point with respect to $D(i)$ then
    return False ;
end
if $p$ is a tangent point then
    Flag($p$,$other_i(p)$) := 1;
    return True ;
end
Flag($p$,$i$) := 1;
$q$:=$p$;
$j$:=$other_i(p)$;
$p$:=$succ_j(p)$;
$i$:=$j$;
while $p \neq q$ do
    if $Flag(p,i) = 1$ OR $p$ is an entry point with respect to $D(i)$ OR $p$ is a tangent
    point then
        return False ;
    end
    Flag($p$,$i$) := 1;
    $j$:=$other_i(p)$;
    $p$:=$succ_j(p)$;
    $i$:=$j$;
end
return True ;
```

data collectors are to be placed only at points in the complete set. Therefore, the linear programming formulation we present here is applicable to the two problems. The two problems differ, however, in the algorithm that finds complete sets as we have seen in the previous two subsections.

Now, we define the following constants and variables.

Constants:

G_i	is the data generation rate of sensor node n_i (i.e., the number of data units generated by node n_i per round).
$N(i)$	is a set of indices such that $j \in N(i)$ if n_j is within the transmission range of n_i (i.e., n_j is a neighbor of n_i).

Algorithm 4: Finding all MORs.

```
Procedure FindMORs( )
Output: A set $\mathcal{P}$ that contains one point from every MOR.
$\mathcal{P}$:=$\phi$ ;
FindArrangement( );
foreach intersection point $p$ which is incident to two transmission disks $D(i)$ and
$D(j)$ do
    Flag($p$,$i$):=0;
    Flag($p$,$j$):=0;
end
foreach sensor node $n_i$ do
    if $D(i)$ has no intersection points then
        $\mathcal{P}$:=$\mathcal{P} \cup \{loc_i\}$;
    else
        foreach intersection point $p$ incident to $D(i)$ do
            if Maximal($n_i$,$p$) then
                $\mathcal{P}$:=$\mathcal{P} \cup \{p\}$;
            end
        end
    end
end
```

$M(i)$	is a set of indices such that $j \in M(i)$ if the point p_j, which belongs to a point in the complete set, is within the transmission range of n_i.
$B(j)$	is a set of indices such that $i \in B(j)$ if the point p_j, which belongs to a point in the complete set, is within the transmission range of n_i.
N	is the number of sensor nodes.
R	is the number of data collectors.
M	is the size of the complete set (i.e., $\lvert\mathcal{P}\rvert$).
E_i	is the residual energy of sensor node n_i at the beginning of the round.
E_{Tr}	is the energy consumed to send one data unit.
E_{Rc}	is the energy consumed to receive one data unit.
C_i	is the traffic capacity of sensor node n_i (i.e., the maximum number of data units that can be relayed by n_i per round).
α	is the weight assigned to the minimum residual energy.
β	is the weight assigned to the total consumed energy.

Variables:

d_i	$d_i = 1$ if a data collector is located at p_i, which belongs to a point in the complete set, and $d_i = 0$ otherwise.
f_{ij}	If $j \in N(i)$, f_{ij} is the flow from sensor node n_i to sensor node n_j (i.e., the number of data units to be sent from n_i to n_j per round).
h_{ij}	If $j \in M(i)$, h_{ij} is the flow from sensor node n_i to the data collector at p_j (if no data collector is placed at p_j, h_{ij} will be set to 0).
E_{min}	is the minimum residual energy over all sensor nodes at the end of the round.
E_{total}	is the total consumed energy during the round.

Our policy of maximizing the lifetime is to maximize the minimum residual energy (i.e., E_{min}) at the end of each round. However, it is easy to see that more than one solution may have the same optimal value for E_{min} but possibly different values for E_{total}. Amongst those solutions for which E_{min} is maximized, we want to pick the one with the minimum E_{total}. In order to do so, we have the following objective function:

$$\alpha\ E_{min} - \beta\ E_{total},$$

which is composed of a linear combination of E_{min} and E_{total} with a much higher weight given to E_{min} (i.e., $\alpha \gg \beta$). We use

$$\alpha = 1 \text{ and } \beta = \frac{1}{\sum\limits_{0 \le i < N} E_i}$$

(note that E_i is constant, $0 \le i < N$). So that $0 < \beta E_{total} < 1$ and, hence, any increase to E_{min} dominates any decrease to E_{total}.

While other objective functions can be easily integrated in our MILP, we claim that residual energy must be taken into account in order to balance the energy levels over the network and, hence, prolong its lifetime. In fact, nodes with low energy supply should be relieved from relaying data for other nodes.

A similar linear programming formulation was introduced in [5]. Yet the authors in [5] assume the existence of a set of feasible locations (i.e., the set $\mathcal{P}$ which we construct in the previous subsections). Furthermore, our objective function is different from the one in [5]. Our scheme prolongs the lifetime by maximizing the minimum residual energy over all nodes at the end of each round. Including the work in [5], several existing proposals for different routing and placement problems aim at prolonging the lifetime by minimizing the total consumed energy or minimizing the maximum amount of energy consumed by a single node. We argue that these schemes are not really suitable for the placement of mobile data collectors. This is because the optimal solution towards such objectives will not change over time and, thus, the locations of data collectors will not be changed. Residual energy must be taken into account for any mobile data collector scheme as it is the only attribute that changes over time, and the only factor that makes such mobility desirable. In our scheme a MILP that reflects our policy is to be solved at the beginning of each round in order to move data collectors to new locations.

The MILP is shown in Fig. 3.6. Inequality (3.1) satisfies the traffic capacity constraints and equation (3.2) guarantees the flow balance. Inequality (3.3) makes E_{min} the minimum residual energy over all sensor nodes (note that we maximize E_{min}). Equation (3.4) sets E_{total} to the total energy consumption. Inequality (3.5) guarantees that the energy expenditure of any sensor node is not more than its current residual energy. Inequality (3.6) guarantees that if no data collector is located at p_j (i.e., d_j=0), no flow is sent to p_j. Equation (3.7) satisfies the constraint that only R data collectors are available. This MILP is supposed to be solved by the beginning of each round in order to move data collectors to new locations.

The time complexity of our approach is $O(N^2 \log N)$ time to find a complete set plus the time to solve the MILP. While MILPs are NP-hard in general, there exist several advanced algorithms (e.g., branch-and-bound, branch-and-cut, and branch-and-price),

Maximize $\alpha\, E_{min} - \beta\, E_{total}$
s.t.,

$$\sum_{j \in N(i)} f_{ij} + \sum_{j \in M(i)} h_{ij} \leq C_i, 0 \leq i < N \tag{3.1}$$

$$\sum_{j \in N(i)} f_{ij} + \sum_{j \in M(i)} h_{ij} - \sum_{j \in N(i)} f_{ji} = G_i, 0 \leq i < N \tag{3.2}$$

$$\begin{aligned} E_i \quad - \quad & E_{Tr}(\sum_{j \in N(i)} f_{ij} + \sum_{j \in M(i)} h_{ij}) \\ - \quad & E_{Rc} \sum_{j \in N(i)} f_{ji} \geq E_{min} \quad , 0 \leq i < N \end{aligned} \tag{3.3}$$

$$\begin{aligned} \sum_{0 \leq i < N} (\sum_{j \in N(i)} E_{Rc} f_{ji} + \sum_{j \in N(i)} E_{Tr}\, f_{ij} \\ + \sum_{j \in M(i)} E_{Tr} h_{ij}) = E_{total} \end{aligned} \tag{3.4}$$

$$E_{min} \geq 0 \tag{3.5}$$

$$\sum_{i \in B(j)} h_{ij} \leq d_j \sum_{0 \leq i < N} G_i, 0 \leq j < M \tag{3.6}$$

$$\sum_{0 \leq i < M} d_i = R \tag{3.7}$$

$$d_i \in \{0, 1\}, 0 \leq i < M$$

Figure 3.6: MILP formulation.

which can be used to find near-optimal solutions [59]. For small MILPs, these algorithms are able to find the optimal solution within a reasonable time. For relatively large MILPs, one can use any of these algorithms, let it run and search for better solutions for a given amount of time, and then take the best solution it has found. In our experiments, we were able to obtain very good solutions (as compared with the best existing schemes) within reasonable time (10-20 minutes) for networks of up to 200 sensor nodes and five data collectors. With the fact that these MILPs are to be solved once every round (and the round lasts for at least a couple of hours), 20 minutes is not too long; during the last

20 minutes of a round, the MILP of the next round is solved.

3.4 Experimental results

We compare our proposed schemes with two other schemes: a stationary scheme where data collectors are stationary and a mobile scheme that ignores the residual energy of different sensor nodes. In the stationary scheme, data collectors are placed randomly in the sensing field, and we use a similar linear program to find near-optimal flow paths. While our schemes have the objective of Maximizing the minimum Residual energy (MR), the mobile scheme we compare with Minimizes the Maximum energy (MM) consumed by a single sensor node (this objective has been used in several routing and placement problems, viz the work in [5]).

We use the general energy consumption model presented in [4] which can be described as follows.

$$E_{Tr}(r, b) = b \times (e_{elec} + e_{amp} \times r^{\gamma}) \tag{3.8}$$

$$E_{Rc}(b) = b \times e_{elec} \tag{3.9}$$

where $E_{Tr}(r, b)$ is the energy consumed to send b bits over r m, $E_{Rc}(b)$ is the energy consumed to receive b bits, e_{elec} is the energy consumed by the transmitter (receiver) to send (receive) one bit, e_{amp} is the energy consumed by the transmission amplifier for one bit, and γ is the path-loss exponent. In our simulation, r is set to 50 m, e_{elec} is set to 50 nJ/bit, e_{amp} is set to 0.1 nJ/bit/m^2, and γ is set to 2. The packet size is 512 bits. Every sensor node has an initial energy of 6 J. Data generation rates are uniformly distributed between 100 and 200 packets/round.

Our simulations involve networks of 200 sensor nodes randomly deployed in a 300 m x 300 m field. To add tracks to the sensing field, we generate 10 points uniformly distributed over the sensing field, and we construct the *relative neighborhood graph* (RNG) of these points [61]. The edges of the resulting RNG are used as tracks spanning the field. In each network, we tested different scenarios of one, three, and five data collectors. To solve the MILP, we use *lp_solve* 5.5 [62] with a timeout of 20 minutes.

Fig. 3.7 shows the lifetime comparison between our MR scheme, the stationary scheme, and the mobile MM scheme for the on-track placement problem. Fig. 3.8 shows a comparison of the average energy consumed per bit between the three schemes for the on-track placement problem.

Fig. 3.9 shows the lifetime comparison between the three schemes for the general placement problem. Fig. 3.10 shows a comparison of the average energy consumed per bit between the three schemes for the general placement problem.

It can be observed that the three schemes have similar energy consumption per bit, yet our MR scheme has a much longer network lifetime. The lifetime improvement of our MR scheme over the stationary scheme is in the order of 250%. This is due to the load balancing effect that is achieved by exploiting data collectors mobility. The MR scheme also makes about 40% lifetime improvement over the MM scheme. This is a direct result of the considerations made for residual energy at single nodes. One of the interesting results that we have is that the MR scheme may consume a little more energy than the MM scheme (as shown in Fig. 3.8 and Fig. 3.10), yet the MR scheme makes a longer network lifetime (as shown in Fig. 3.7 and Fig. 3.9). The reason is that the MR scheme may choose a route or a data collector location that spends more energy for the sake of avoiding and reducing the load on sensor nodes with low residual energy. An energy-efficient route or data collector location that puts extra load on a sensor node with a critical energy level is not preferred in the MR scheme.

The standard deviations of all averaged results in our simulations in this chapter were within 10.8% of the average value. Using different values for the MILP solver timeout, different sensor densities, and different number of data collectors have shown similar trends.

Figures 3.11 - 3.18 show an example of how energy is depleted in a network of 64 nodes and 2 data collectors using the MR and the MM schemes with a general placement. Each

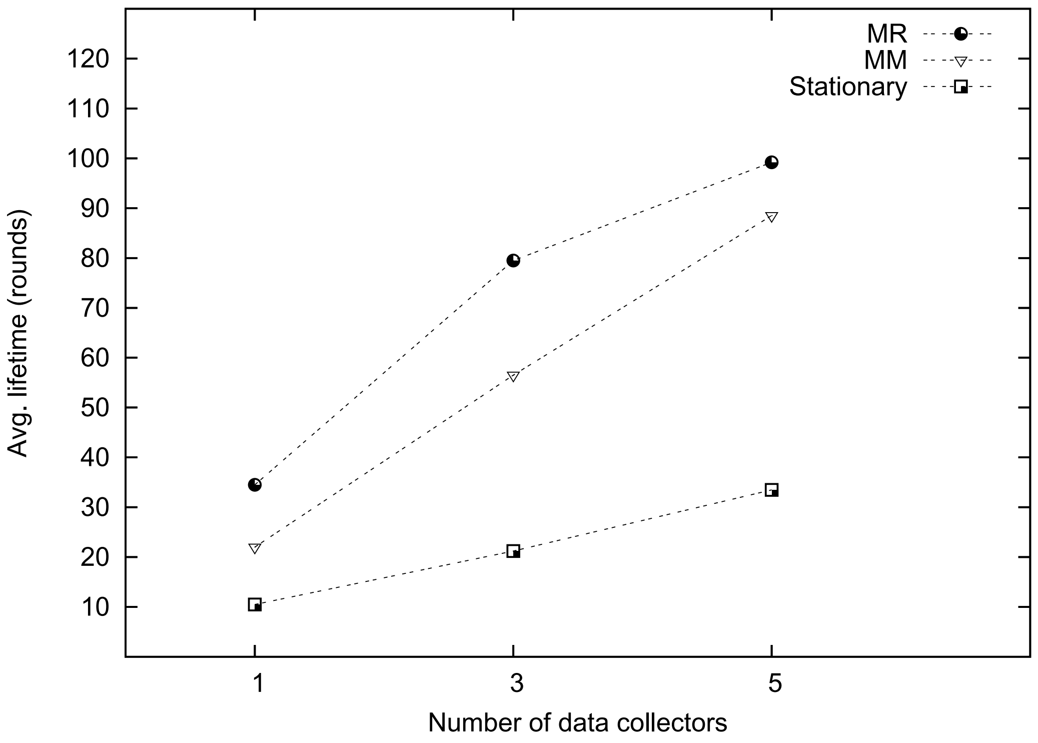


Figure 3.7: Lifetime comparison for the on-track placement problem.

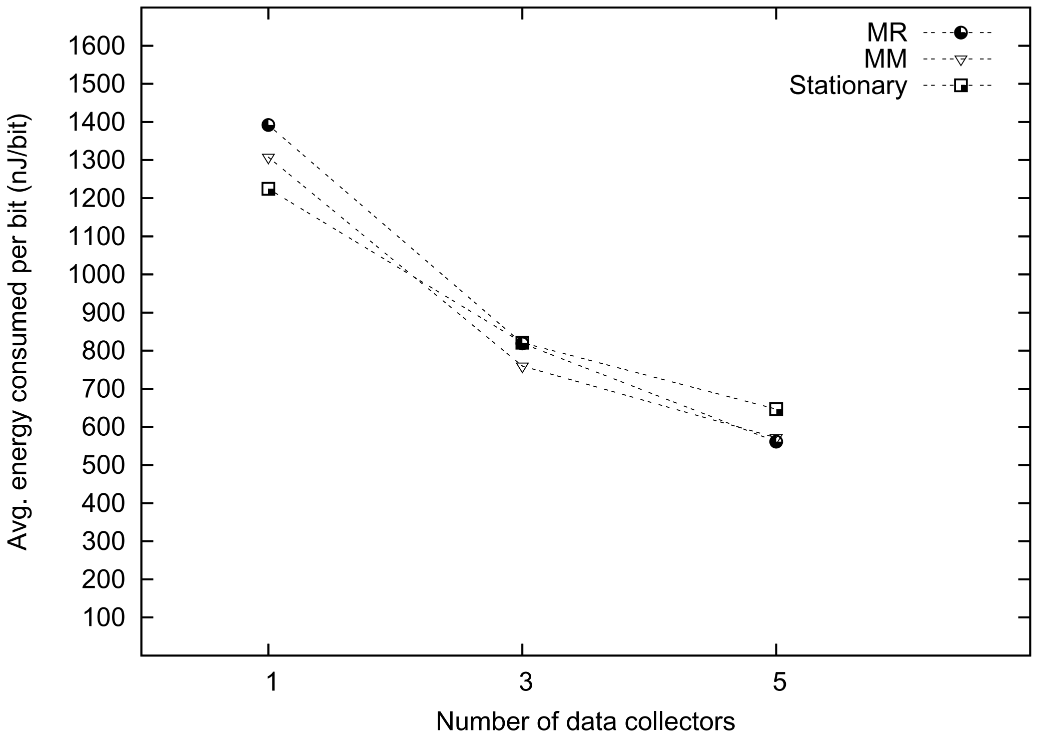


Figure 3.8: Energy consumption comparison for the on-track placement problem.

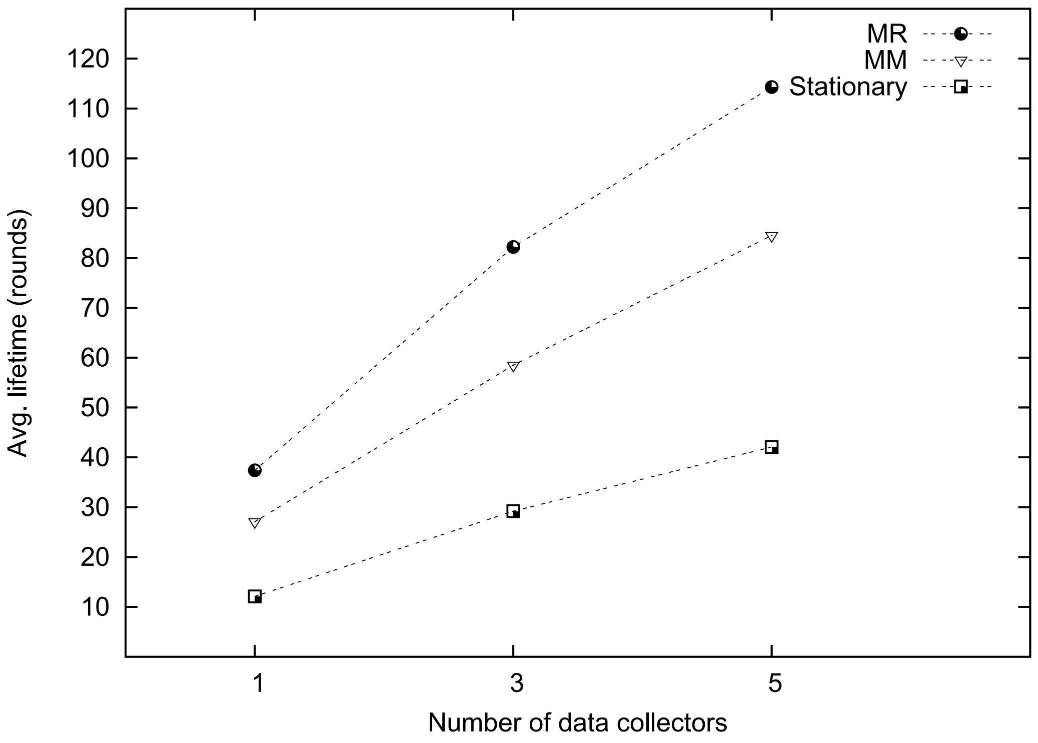


Figure 3.9: Lifetime comparison for the general placement problem.

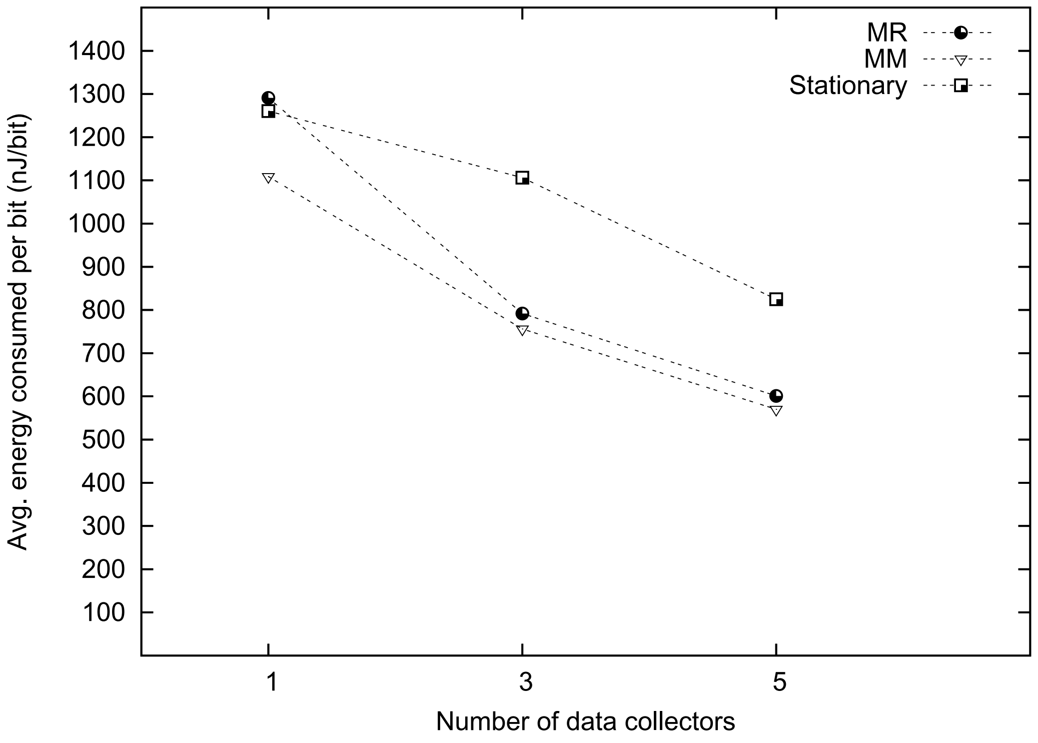


Figure 3.10: Energy consumption comparison for the general placement problem.

plot shows the residual energy of every sensor node for each scheme at the end of a round. In each plot, every sensor node is represented by a square and a disk: the height of the square (disk) reflects the residual energy of the corresponding sensor node at the end of the round using the MM (MR) scheme. Note that the MR scheme tries to maintain a bound on the minimum residual energy which helps in balancing the residual energy amongst all sensor nodes and, hence, prolonging the lifetime of the network. For example, at the end of the fifth round, the lowest energy level using the MR scheme is 261 mJ and the lowest energy level using the MM scheme is 138 mJ. Furthermore, at the end of the last round, the lowest energy level using the MR scheme is 116 mJ and 37.5% of the nodes using the MM scheme have energy levels lower than 116 mJ.

3.5 Summary

This chapter addresses the problem of unbalanced energy expenditure in WSNs resulting from using a stationary sink and multi-hop relaying. To alleviate the effect of this problem, we argue for using multiple mobile data collectors, and propose a scheme for finding near-optimal placement of mobile data collectors together with the routing paths to deliver data to data collectors. The novelty of our scheme stems from:

1. Solving a general problem in which a data collector can be placed anywhere in the sensing field and another problem in which data collectors can move along and be placed on tracks spanning the sensing field.
2. Finding a complete, finite search space for data collector locations for each problem.

Linear programming is used to find near-optimal solutions from the obtained search space. We use an objective function that takes into account both the current residual energy and future energy expenditure of each sensor node. Experimental results show that our scheme has the potential to prolong the lifetime of a WSN significantly as compared with another stationary scheme and a mobile scheme that does not consider the residual energy of nodes.

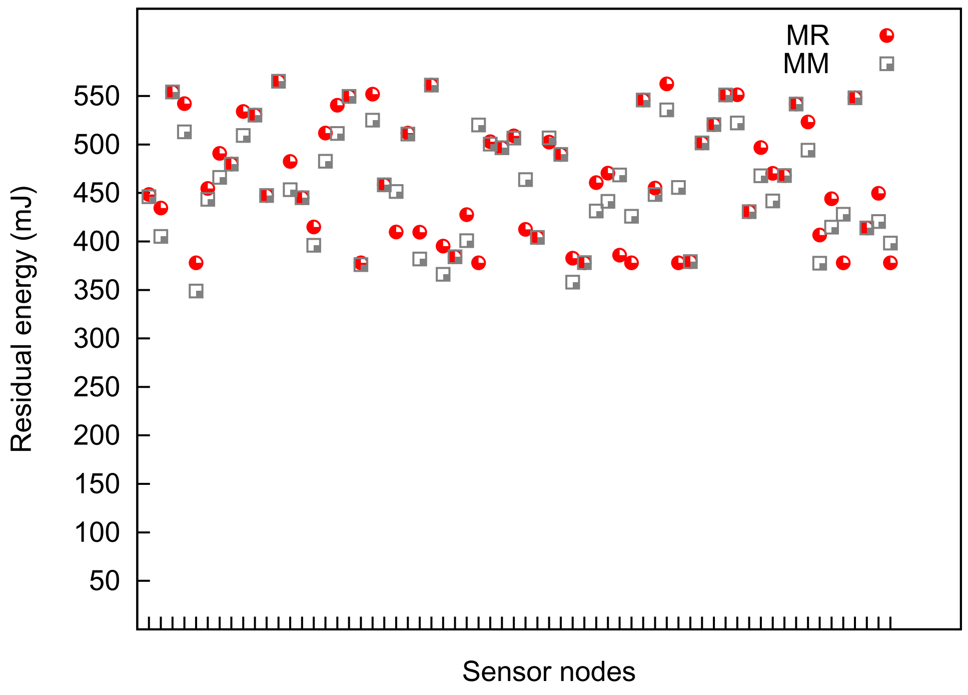


Figure 3.11: Round no. 1.

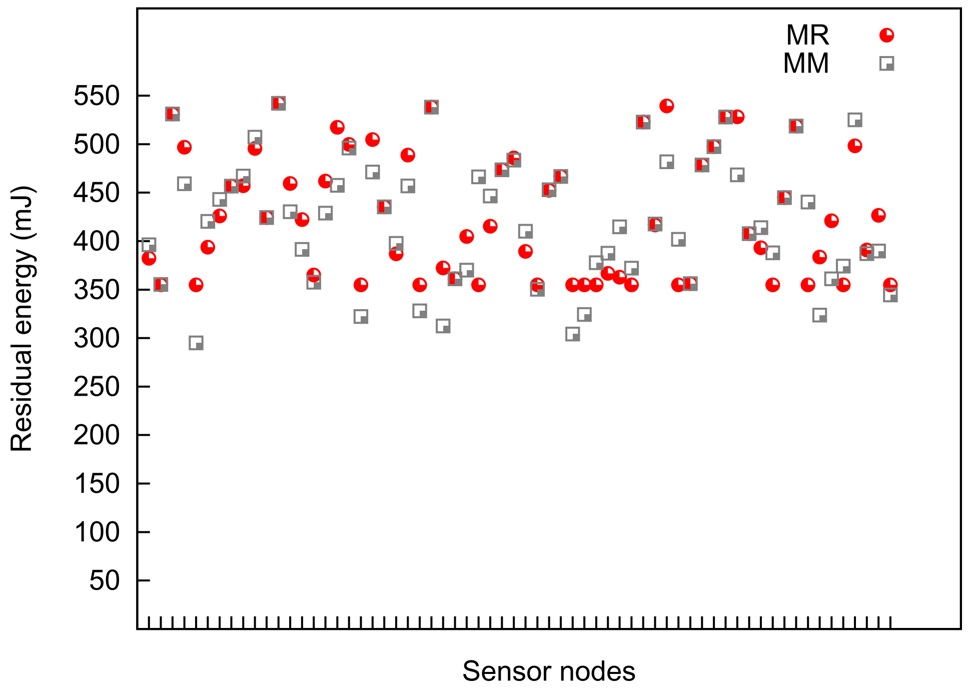


Figure 3.12: Round no. 2.

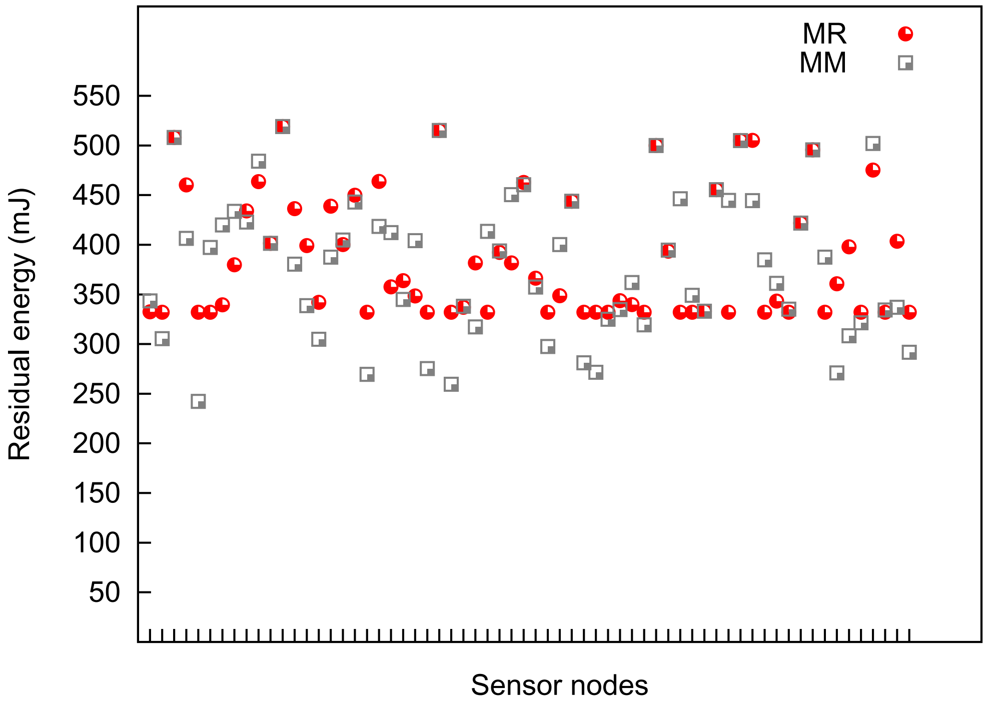


Figure 3.13: Round no. 3.

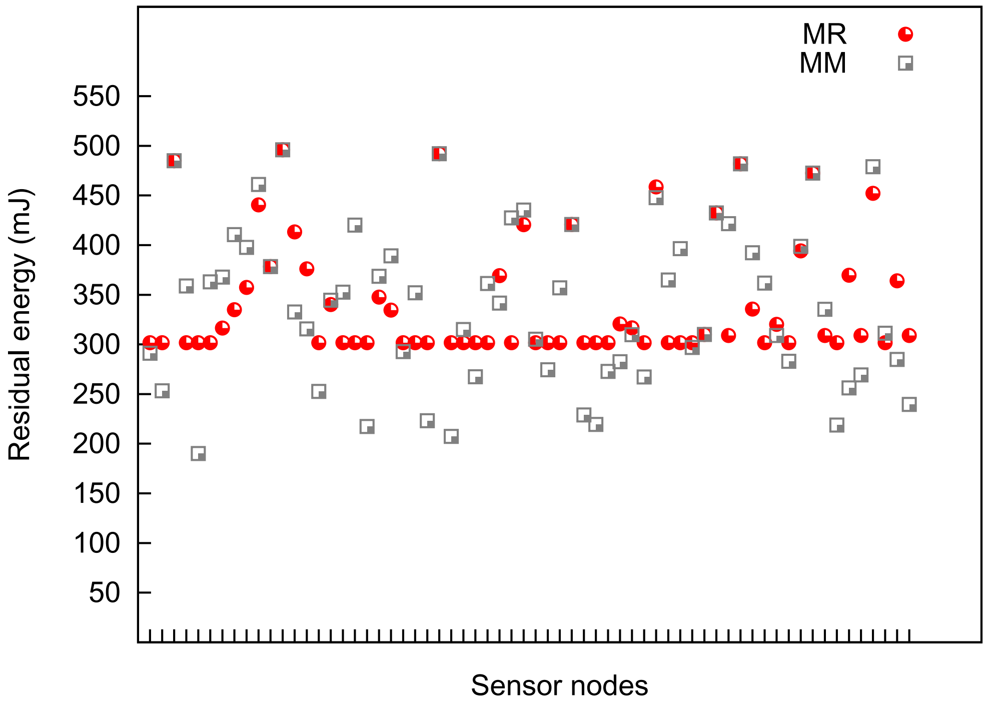


Figure 3.14: Round no. 4.

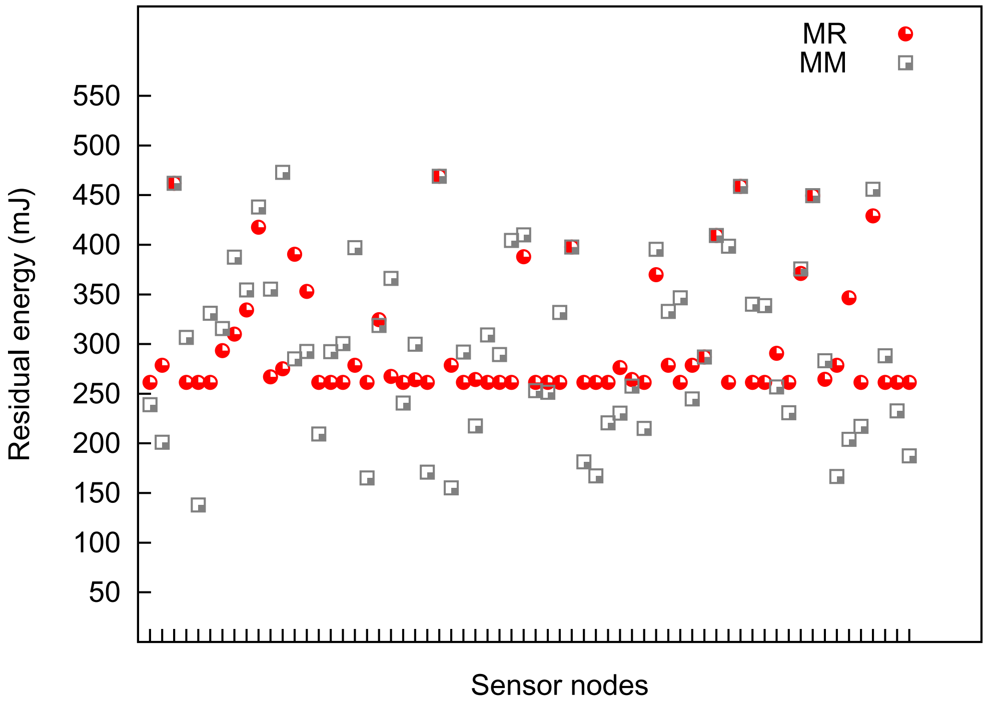


Figure 3.15: Round no. 5.

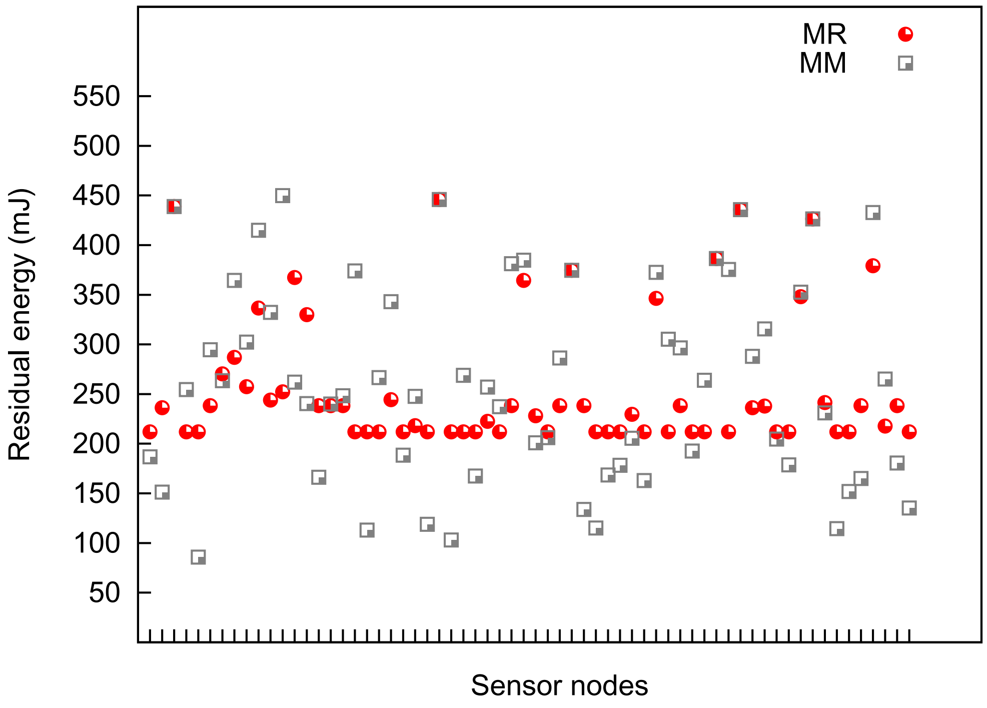


Figure 3.16: Round no. 6.

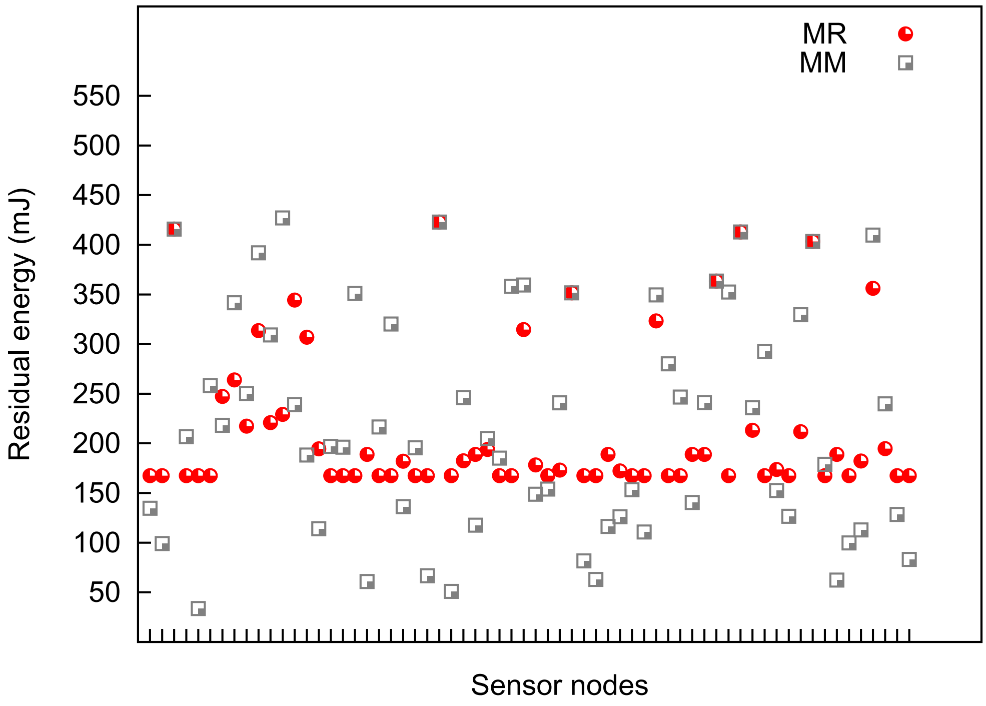


Figure 3.17: Round no. 7.

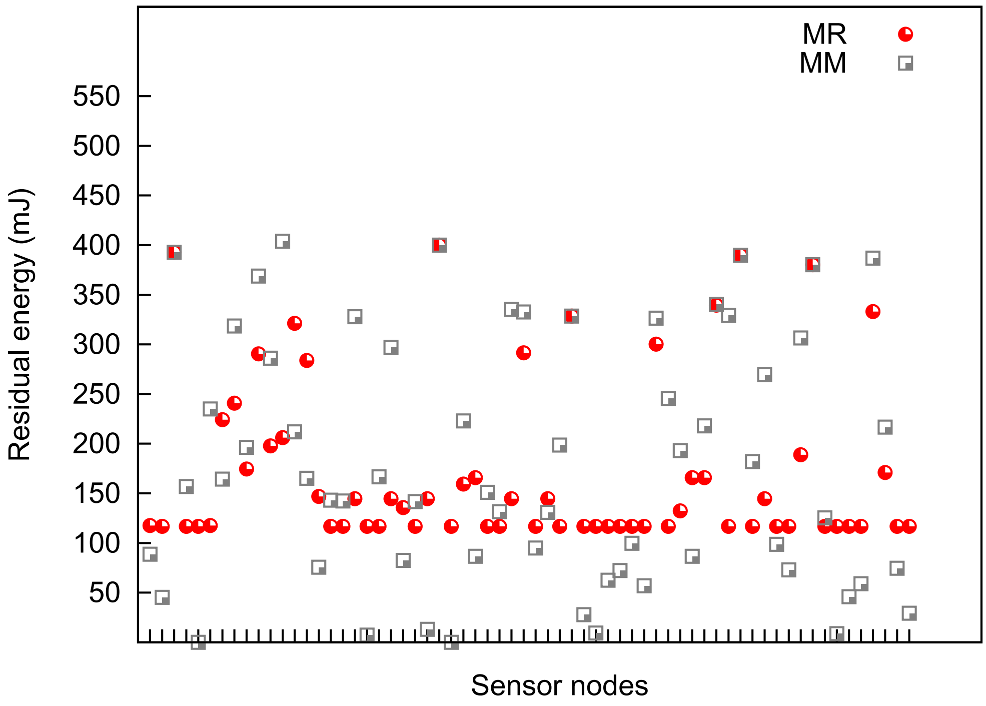


Figure 3.18: Round no. 8.

Chapter 4

Mobile data collectors in underwater WSNs

Underwater sensing has a prominent potential for environmental monitoring and oceanographic data gathering. A network of underwater sensors can provide unprecedented ability to monitor oceans in our planet and to facilitate a wide range of major applications. This includes providing synoptic sampling of the ocean, monitoring and tracking events (e.g., pollution), preventing natural disasters (e.g., a Tsunami warning system), monitoring underwater equipments and infrastructure (e.g., cables), and supporting ocean industry (e.g., fishery and transportation) [63]. While wired networks can provide reliable communication and a continuous power supply, they demand high cost and significant engineering effort. Neptune is a project led by a couple of universities in North America to build the world's largest wired ocean observatory. It involves laying a total of 3000 km of powered fiber-optic cables over a 200,000 km^2 region off the coasts of British Columbia, Washington, and Oregon. The cost of only laying the cables down on the seabed is about 33 million dollars [64]. Recently, attention has been directed to wireless networks because of their potential to provide low-cost, distributed underwater monitoring [65] [66] [67]. However, wireless communication underwater is much more challenging than that in terrestrial environments: underwater environments are 3D, underwater wireless communication consumes much more energy, and the speed of wireless signals underwater is significantly lower than that in terrestrial environments (see Section 4.1). The great potentials of underwater WSNs and the challenges in their communications motivated us to extend our work to this 3D underwater environment.

The remainder of this chapter is organized as follows. Section 4.1 describes the contribution of this work and its motives. The system model is given in Section 4.2. In Section 4.3, we present our Delay-Tolerant Placement and Routing (DTPR) scheme. In Section 4.4, we present our Delay-Constrained Placement and Routing (DCPR) scheme. Section 4.5 shows the experimental results. Finally, in Section 4.6, we conclude by summarizing the main issues in this chapter.

4.1 Motivations and contributions

Acoustic signals are known to be the typical wireless medium for underwater communication. Radio waves require large antennae and demand high power to propagate underwater. Optical waves can propagate long distances underwater but require high precision in pointing the laser beams. However, the speed of acoustic signals underwater is around 1500 m/s; that is not even comparable to the speed of terrestrial radio signals, which is around 3×10^8 m/s. Furthermore, the energy consumption of an acoustic modem is higher than that of a terrestrial radio frequency modem [68]. To have a better control on the energy consumption of acoustic modems, their transmission range is usually tunable to a number of discrete levels (i.e., variable, discrete transmission range).

In this chapter, we generalize the placement and routing problem in three aspects. First, rather than having sensor nodes and data collectors on the same surface, sensor nodes float at different depths in the ocean and data collectors stay on the surface of the water. Fig. 4.1 depicts this architecture. Second, sensor nodes have variable, discrete transmission ranges rather than a fixed transmission range. Third, we consider an upper bound on the delay between the sensor nodes and the data collectors.

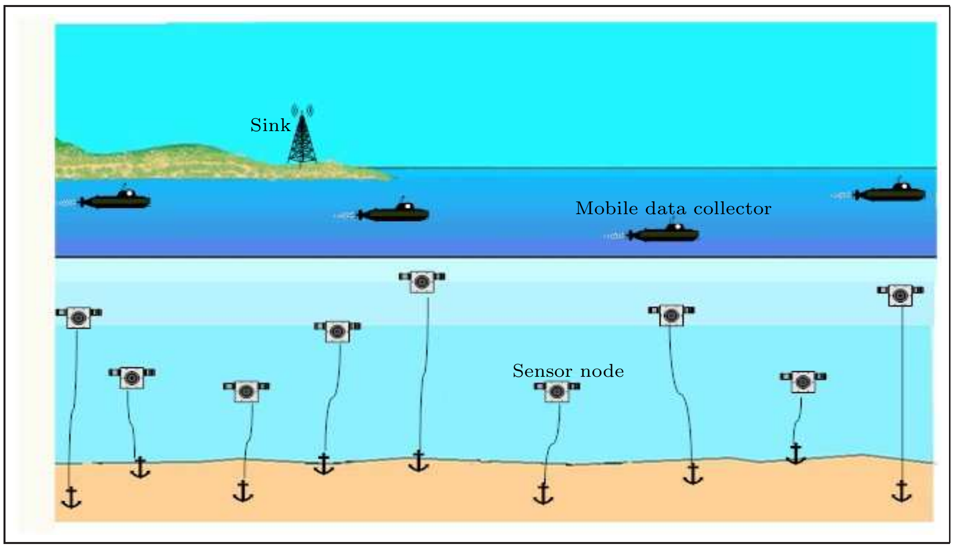


Figure 4.1: A 3D architecture for underwater WSNs.

4.2 System model

We consider an underwater WSN consisting of N sensor nodes and R data collectors. Each sensor node collects data from the surrounding environment and sends the collected data to one of the collectors either directly or through other nodes (i.e., multi-hop communication). We follow the three-dimensional architecture proposed in [65]. In this architecture, data collectors are equipped with a radio transceiver to communicate with an on-shore sink, and an acoustic transceiver to communicate with underwater sensors. While data collectors stay on the surface to keep the link with the sink, sensor nodes, which are equipped with acoustic transceivers, float at different depths to carry out the sensing mission.

We assume that the transmission range of an acoustic transceiver can be adjusted to a constant number of levels (i.e., variable, discrete transmission ranges). Thereby, when a sensor node wants to send data to another node, it uses the smallest transmission range that can reach the destination. The topology of the network is modeled as a weighted graph $G = (V, E)$, where $V = \{n_0, n_1, ..., n_{N-1}\}$ is the set of N sensor nodes, and $(i, j) \in E$, if sensor nodes n_i and n_j can communicate with each other directly (i.e., their separation is not more than the maximum transmission range). W_{ij} denotes the weight of the link (i, j) which is the Euclidean distance between senor nodes n_i and n_j; the energy consumption and the delay over the link (i, j) are functions of W_{ij}. Each sensor node n_i has a data generation rate G_i. G_i is the number of data units generated by node n_i per time unit.

We adopt a discrete delay model which can be tuned to meet any desired accuracy. This delay model is described in details in Section 4.4. Moreover, due to the relatively low speed of sound in water (approximately 1500 m/second), the propagation delay dominates the transmission and queueing delays. Therefore, we assume that transmission and queueing delays are negligible.

Our approach is independent of the underlying MAC protocol. We assume a capacity limit for each sensor node which limits the number of data units that can be transmitted by a sensor node during one round. The capacity of a sensor node n_i is denoted by C_i. While this capacity constraint is applicable to communication links, to simplify the presentation, we will limit ourselves to capacity constraints on individual nodes.

We assume that data generation rates, residual energy, and locations of all sensor nodes are known in advance. We also assume that data collectors are not energy constrained as they can be easily recharged [57]. Without loss of generality, we define the lifetime of the network as the time until the first sensor node dies. Yet other definitions (e.g., the time until a particular proportion of the sensors die) can be equally used in our schemes.

4.3 Delay-tolerant placement and routing

In this section, we present a scheme for routing and placement of mobile data collectors that handles variable transmission ranges in a 3D underwater environment. The objective of this DTPR scheme is to maximize the lifetime of the network with no delay constraints; there is no bound on the length of any path between a sensor node and a data collector. The DTPR problem can be defined as follows:

The lifetime of the network is divided into equal length rounds. At the beginning of each round, find the optimal locations of R data collectors together with the routing paths to deliver the generated data from all sensor nodes to data collectors. The objective is to maximize the minimum residual energy at the end of the round. Data collectors can be placed anywhere on the surface of the water.

In this 3D environment, a data collector may be placed anywhere on the surface of the water as long as it is within the acoustic transmission range of at least one sensor node. Since each sensor node has a discrete number of transmission levels, its transmission range can be viewed as a set of transmission spheres: one sphere for each transmission level. Further, since data collectors are placed on the surface of the water, we should find the intersection of these spheres and the surface of the water. Without loss of generality, we let the plane $z = 0$ represent the surface of the water (where the $z-$ axis represents the depth). Then, the intersection of a sphere centered at a point (a, b, c) with a radius of r and the plane $z = 0$ is:

- ϕ, the empty set, if $r^2 - c^2 < 0$.
- A disk centered at a point $(a, b, 0)$ with a radius of $\sqrt{r^2 - c^2}$, if $r^2 - c^2 \geq 0$. We call such a disk a *surface disk*.

Fig. 4.2 shows an example of the intersection of a sphere and a plane.

Each sensor node will have at most one surface disk for each transmission level. And since data collectors are placed on the surface of the water, we can focus on these surface disks rather than the transmission spheres, i.e., the problem becomes two-dimensional. So let us say each sensor node n_i has γ_i surface disks $\{\zeta_i^0, \zeta_i^1, ..., \zeta_i^{\gamma_i-1}\}$.

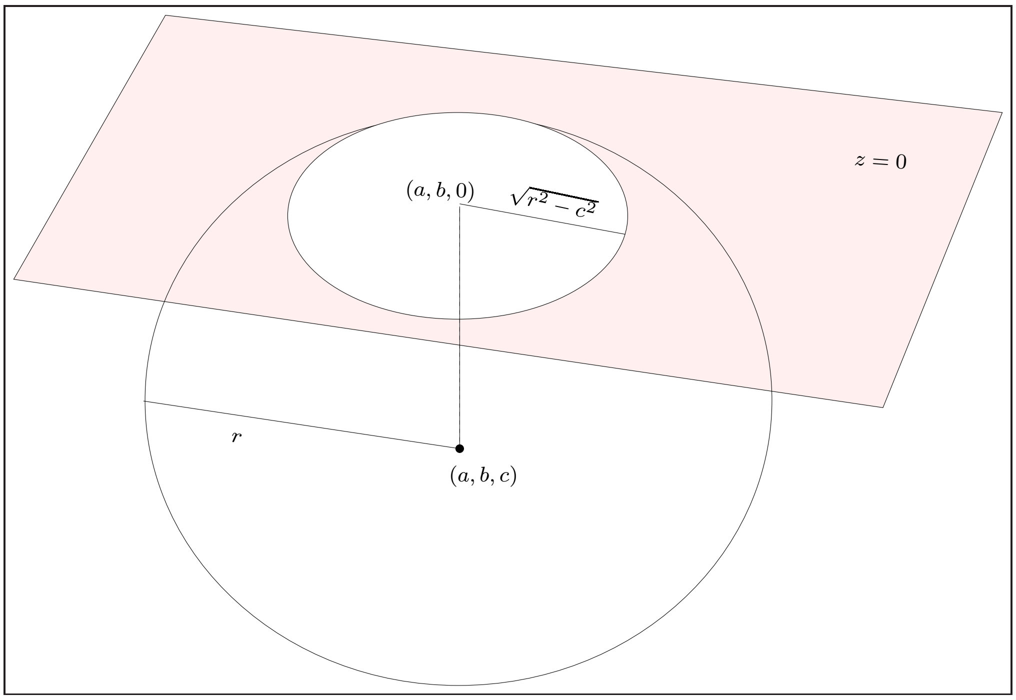


Figure 4.2: An example of the intersection of a sphere and the plane $z = 0$.

4.3.1 Finding a complete set for the DTPR problem

In order to find a complete set for the DTPR problem, we extend the method presented in the previous chapter for finding all MORs to handle multiple transmission levels. Observation 2 can be generalized as follows.

Observation 3 An overlapping region β is maximal if and only if it satisfies one of the following properties:

1. β is a tangent point.
2. β is a disk whose perimeter does not have any intersection points AND no disk lies entirely inside β.
3. β is a convex overlapping region whose boundaries do not have any tangent points AND no disk lies entirely inside β.

Observation 3 takes into account the case of a disk lying entirely inside another disk, which is not possible when all sensor nodes are deployed on a 2D space and have the same single transmission range. To check whether a disk lies entirely inside an overlapping region, we add *defect points* to the arrangement of disks. A disk may add at most one defect point. To find the defect point of a disk ζ_i^k centered at a point (x_i, y_i), we find points that:

1. lie at the intersection of the line $y = y_i$ and another disk ζ_j^l.

2. are outside and to the right of ζ_i^k.

Amongst those points that satisfy these two conditions, we take the point p which is the closest to (x_i, y_i). Let us say p lies at the boundary of a disk ζ_j^l. Then, if (x_i, y_i) is inside ζ_j^l, p is added as a defect point on ζ_j^l. Note that such a point does not necessarily exist for all surface disks, i.e., some disks may not add any defect point. Fig. 4.3 shows an example of an overlapping region which is not maximal because it is concave (a), a MOR (b), and an overlapping region which is not maximal because of a defect point (c). Note that the whole area of the small disk in Fig. 4.3 (c) is a MOR.

Since a sensor node may have multiple surface disks, we need to redefine the functions *other* and *succ* as follows:
When p is an intersection point of the boundaries of ζ_i^k and ζ_j^l, $succ_i^k(p)$ ($succ_j^l(p)$) denotes the intersection point incident to ζ_i^k (ζ_j^l) that comes right after p according to a clockwise order. $other_i^k(p) = (j, l)$ and $other_j^l(p) = (i, k)$.

In what follows, Algorithm 5 finds the arrangement of surface disks together with the defect points; it runs in $O(N^2 \log N)$ time. Algorithm 6 tests whether an overlapping region is maximal or not. Algorithm 7 uses Algorithms 5 and 6 to construct a complete set $\mathcal{P}$ by finding all MORs. Note that the Boolean array $Flag(p, i)$ is used to guarantee that we do not check the same overlapping region more than once. The overall complexity of Algorithm 7 is $O(N^2 \log N)$.

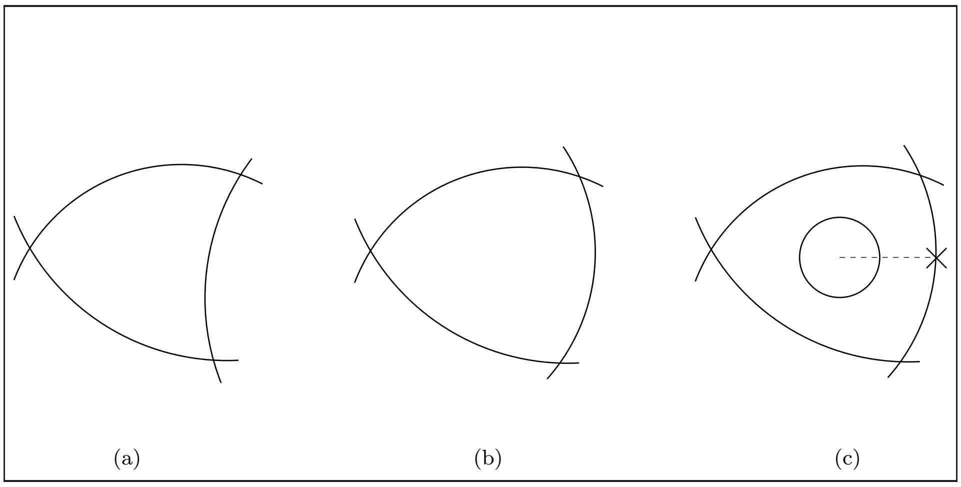


Figure 4.3: Overlapping regions, MORs, and defect points.

Algorithm 5: Arrangement of surface disks for the DTPR problem.

```
Procedure FindArrangement( )
foreach sensor node $n_i$ do
    Find all surface disks corresponding to $n_i$;
end
Find all points where the boundaries of two surface disks intersect;
Find all defect points;
foreach surface disk $\zeta_i^k$ do
    Sort all intersection points and defect points incident to $\zeta_i^k$ in a clockwise order;
end
```

Algorithm 6: Testing whether an overlapping region is maximal or not.

```
Function Maximal($\zeta_i^k$: a disk, $p$: a point)
Input:  A surface disk $\zeta_i^k$ and a point $p$, which could be an intersection point or a
        defect point, incident to $\zeta_i^k$.
Output: True if a MOR is found, and False otherwise.
if Flag(p,i) $= 1$ OR $p$ is an entry point with respect to $\zeta_i^k$ OR $p$ is a defect point
then
    return False ;
end
if $p$ is a tangent point then
    Flag($p$,$other_i^k(p)$) := 1;
    return True ;
end
Flag($p$,$i$) := 1;
$q$:=$p$;
$(j, l)$:=$other_i^k(p)$;
$p$:=$succ_j^l(p)$;
$i$:=$j$;
$k$:=$l$;
while $p \neq q$ do
    if Flag(p,i) $= 1$ OR $p$ is an entry point with respect to $\zeta_i^k$ OR $p$ is a defect
    point OR $p$ is a tangent point then
        return False ;
    end
    Flag($p$,$i$) := 1;
    $(j, l)$:=$other_i^k(p)$;
    $p$:=$succ_j^l(p)$;
    $i$:=$j$;
    $k$:=$l$;
end
return True ;
```

Algorithm 7: Finding all MORs.

```
Procedure FindMORs( )
Output: A set $\mathcal{P}$ that contains one point from every MOR.
$\mathcal{P}$:=$\phi$ ;
FindArrangement( );
foreach intersection point p which is incident to two surface disks $\zeta_i^k$ and $\zeta_j^l$ do
    Flag($p$,$i$):=0;
    Flag($p$,$j$):=0;
end
foreach surface disk $\zeta_i^k$ do
    if $\zeta_i^k$ has no intersection points and no defect points then
        $\mathcal{P}$:=$\mathcal{P} \cup \{loc_i\}$;
    else
        foreach intersection point p incident to $\zeta_i^k$ do
            if Maximal($\zeta_i^k$,p) then
                $\mathcal{P}$:=$\mathcal{P} \cup \{p\}$;
            end
        end
    end
end
```

4.3.2 MILP formulation for the DTPR

Once we obtain the set $\mathcal{P} = \{p_0, p_1, ..., p_{M-1}\}$, which contains a point from each MOR, the problem of finding the optimal locations of R data collectors and the routing paths from sensor nodes to data collectors can be formulated as a MILP. This MILP is similar to the one presented in the previous chapter with a minor modification that reflects variable transmission ranges. We define the following constants and variables.

Constants:

G_i	is the data generation rate of sensor node n_i.
$N(i)$	is a set of indices such that $j \in N(i)$ if n_j is within the transmission range of n_i (i.e., n_j is a neighbor of n_i).
$M(i)$	is a set of indices such that $j \in M(i)$ if the point p_j, which belongs to a MOR, is within the transmission range of n_i.
$B(j)$	is a set of indices such that $i \in B(j)$ if the point p_j, which belongs to a MOR, is within the transmission range of n_i.

N	is the number of sensor nodes.
R	is the number of data collectors.
M	is the size of the complete set.
E_i	is the residual energy of sensor node n_i.
$E_{Tr}(d)$	is the energy consumed to send one data unit to a destination which is d units away from the source.
E_{Rc}	is the energy consumed to receive one data unit.
W_{ij}	is the Euclidean distance between sensor nodes n_i and n_j.
D_{ij}	is the Euclidean distance between a sensor node n_i and a point p_j, which belongs to a MOR.
C_i	is the traffic capacity of sensor node n_i (i.e., the maximum number of data units that can be relayed by n_i per round).
α	is the weight assigned to the minimum residual energy.
β	is the weight assigned to the total consumed energy.

Variables:

l_i	$l_i = 1$ if a data collector is placed at p_i, and $l_i = 0$ otherwise.
f_{ij}	If $j \in N(i)$, f_{ij} is the flow from sensor node n_i to sensor node n_j (i.e., the number of data units to be sent from n_i to n_j per round).
h_{ij}	If $j \in M(i)$, h_{ij} is the flow from sensor node n_i to the data collector at p_j (if no data collector is placed at p_j, h_{ij} will be set to 0).
E_{min}	is the minimum residual energy over all sensor nodes at the end of the round.
E_{total}	is the total consumed energy during the round.

The objective is to maximize the minimum residual energy (i.e., E_{min}) at the end of each round. Amongst those solutions for which E_{min} is maximized, we pick the one with the minimum E_{total}.

In our scheme a MILP that reflects our policy is to be solved at the beginning of each round in order to move data collectors to new locations. The MILP is shown in Fig. 4.4. Inequality (4.1) satisfies the traffic capacity constraints and Equation (4.2) guarantees the flow balance. Inequality (4.3) makes E_{min} the minimum residual energy over all sensor nodes (note that we maximize E_{min}). Equation (4.4) sets E_{total} to the total energy consumption. Inequality (4.5) guarantees that the energy expenditure of any sensor node is not more than its current residual energy. Inequality (4.6) guarantees that if no data collector is placed at p_j (i.e., l_j=0), no flow is sent to p_j. Equation (4.7) satisfies the constraint that only R data collectors are available.

It is important to note that this MILP can be modified to handle more complex capacity constraints (e.g., giving different capacities to different links incident to a single node). We show a general case here to simplify the presentation.

4.4 Delay-constrained placement and routing

Even though energy is a critical resource in underwater WSNs, there should be a balance between energy saving and the provisioned QoS. Delay is an example of a QoS metric that can be of significant importance in many applications, especially real-time ones. In the DTPR scheme, which is described in the previous section, a data unit might be sent through a longer path in order to avoid sensor nodes with lower residual energy. However, while the DCPR and the DTPR seek the same goal in that both of them maximize the lifetime of the network, the DCPR has a bound on how much time a data unit may spend in its way to a data collector.

4.4.1 Delay model and DCPR problem definition

Delay has a continuous nature; even if two nodes are placed in a bounded area, there is an infinite number of possible values for the delay between them. That is simply because

Maximize $\alpha\, E_{min} - \beta\, E_{total}$
s.t.,

$$\sum_{j \in N(i)} f_{ij} + \sum_{j \in M(i)} h_{ij} \leq C_i \quad, 0 \leq i < N \tag{4.1}$$

$$\sum_{j \in N(i)} f_{ij} + \sum_{j \in M(i)} h_{ij} - \sum_{j \in N(i)} f_{ji} = G_i \quad, 0 \leq i < N \tag{4.2}$$

$$\begin{aligned} E_i \quad &- \sum_{j \in N(i)} E_{Tr}(W_{ij})\, f_{ij} - \sum_{j \in M(i)} E_{Tr}(D_{ij})\, h_{ij} \\ &- \sum_{j \in N(i)} E_{Rc}\, f_{ji} \geq E_{min} \quad, 0 \leq i < N \end{aligned} \tag{4.3}$$

$$\begin{aligned} \sum_{0 \leq i < N} (\sum_{j \in N(i)} E_{Rc} f_{ji} + \sum_{j \in N(i)} E_{Tr}(W_{ij})\, f_{ij} \\ + \sum_{j \in M(i)} E_{Tr}(D_{ij}) h_{ij}) = E_{total} \end{aligned} \tag{4.4}$$

$$E_{min} \geq 0 \tag{4.5}$$

$$\sum_{i \in B(j)} h_{ij} \leq l_j \sum_{0 \leq i < N} G_i, 0 \leq j < M \tag{4.6}$$

$$\sum_{0 \leq i < M} l_i = R \tag{4.7}$$

$$l_i \in \{0, 1\}, 0 \leq i < M$$

Figure 4.4: MILP formulation of the DTPR.

the delay can take any real value between two real numbers, which is an infinite set. In order to keep the optimization problem discrete, we adopt a discrete, flexible delay model that meets the following two properties:

1. It can be tuned to measure the delay in terms of the number of hops.
2. It can be tuned to measure the delay in terms of the propagation delay with any desired accuracy (i.e., even though it is a discrete model, it is still able to meet any desired accuracy).

This model is described as follows. We use a delay step Δ which could be defined as the distance an acoustic signal would travel in the water in one time unit. Then, the discrete delay over a single-hop link (i, j) is $\lceil \frac{W_{ij}}{\Delta} \rceil$. The discrete delay over a multi-hop path is the sum of the discrete delays of single-hop links that constitute that path. In what follows we use W^*_{ij} to denote the discrete delay over a single-hop link between two sensor nodes n_i and n_j, and D^*_{ij} to denote the discrete delay over a single-hop link between a sensor node n_i and a data collector placed at a point p_j.

Obviously, this discrete model can be tuned to reflect the number of hops by setting $\Delta = MaxRange$, where $MaxRange$ is the maximum transmission range of a sensor node. It is also obvious that Δ (and the time unit) can be made as small as needed to measure the propagation delay with any desired accuracy.

Now, we can define the DCPR problem as follows:

> The lifetime of the network is divided into equal length rounds. At the beginning of each round, find the optimal locations of R data collectors together with the routing paths to deliver the generated data from all sensor nodes to data collectors, such that the discrete delay of a path from a sensor node to a data collector is at most Γ. The objective is to maximize the minimum residual energy at the end of the round. Data collectors can be placed anywhere on the surface of the water.

4.4.2 Finding a complete set for the DCPR problem

In Section 4.3 we defined the complete set as a set of points on the surface of the water, such that there exists an optimal placement in which each data collector is placed at a point in that set. However, a set of points $\mathcal{P}$, which is complete for the DTPR problem, is not necessarily complete for the DCPR problem on the same network. That is because the optimal DTPR placement embedded in $\mathcal{P}$ may be infeasible for the DCPR probelm possibly because it involves a path that exceeds the maximum delay limit. Therefore, we need to modify our algorithms in Section 4.3 to accommodate the delay constraints.

In Section 4.3 we find a complete set using the MORs of the surface disks. These surface disks are obtained from the transmission spheres of all sensor nodes. The transmission spheres of a sensor node n_i divide the space around n_i into layers such that sending a data unit from n_i to any node in the same layer would consume the same amount of energy. However, sending a data unit from n_i to two nodes in the same layer may result in different delay values. What we really need, for the DCPR problem, is to divide the space around n_i into a finite number of layers such that sending a data unit to any two nodes in the same layer would consume the same amount of energy and encounter the same discrete delay. In order to do so, for each sensor node, we add $\lfloor \frac{MaxRange}{\Delta} \rfloor$ virtual delay spheres. When these delay spheres are indexed from 1 to $\lfloor \frac{MaxRange}{\Delta} \rfloor$, the radius of the j^{th} sphere is $j\Delta$. Thereby, a layer, which is the space between two consecutive transmission or delay spheres (see Fig. 4.5), has the same delay and energy properties, i.e., sending a data unit to any node inside the same layer would consume the same amount of energy and encounter the same discrete delay. Thus, we just need to find the MORs of the surface disks of all transmission and delay spheres in order to construct a complete set for the DCPR problem. Algorithm 8 is a modified version of Algorithm 5, which considers delay spheres as well as transmission spheres.

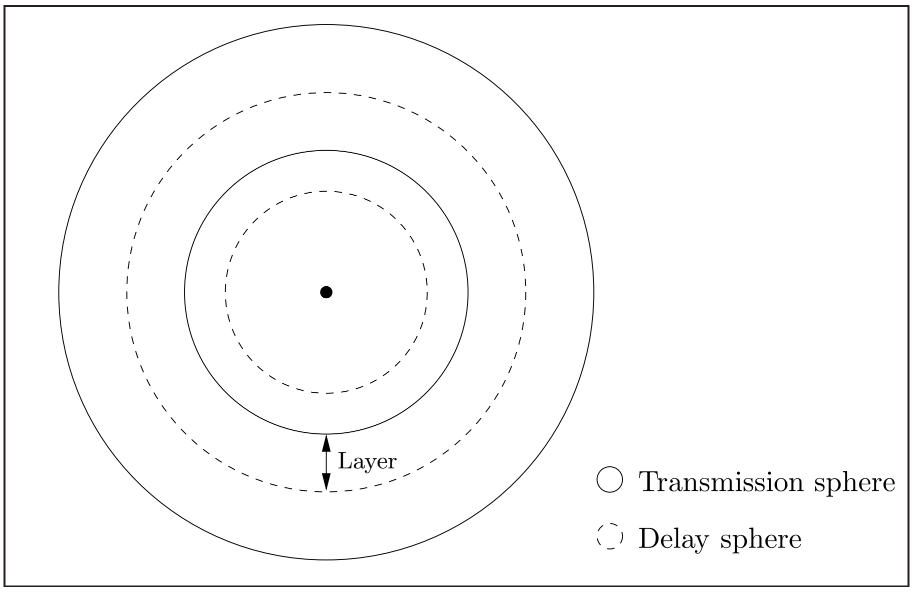


Figure 4.5: A sensor node with four layers.

Algorithm 8: Arrangement of surface disks for the DCPR problem.

```
Procedure FindArrangement( )
foreach sensor node $n_i$ do
    for $j = 1..\lfloor \frac{MaxRange}{\Delta} \rfloor$ do
        Add a delay sphere centered at $n_i$ with a radius of $j\Delta$;
    end
    Find all surface disks derived from all delay and transmission spheres of $n_i$;
end
Find all points where the boundaries of two surface disks intersect;
Find all defect points;
foreach surface disk $\zeta_i^k$ do
    Sort all intersection points and defect points incident to $\zeta_i^k$ in a clockwise order;
end
```

4.4.3 MILP formulation for the DCPR

In order to accommodate the delay in the MILP formulation, the flow from a node n_i to another node n_j, i.e., f_{ij}, is decomposed into several sub-flows according to the encountered delay. f_{ij}^d denotes the flow arriving to node n_j through node n_i, for which the discrete delay from the source of the flow to node n_j is d. In other words, by the time the flow f_{ij}^d arrives to node n_j, it will have encountered d delay steps. Similarly, h_{ij}^d denotes the flow arriving to a data collector placed at a point p_j through node n_i, for which the discrete delay from the source of the flow to the data collector is d.

When Γ is the upper bound on the discrete delay, the flow from a sensor node n_i to another node n_j is decomposed into $\Gamma - W^*_{ij}$ sub-flows: $\{f_{ij}^{W^*_{ij}}, f_{ij}^{W^*_{ij}+1}, \ldots, f_{ij}^{\Gamma-1}\}$. And the flow from a sensor node n_i to a data collector placed at a point p_j is decomposed into $\Gamma - D^*_{ij} + 1$ sub-flows: $\{h_{ij}^{D^*_{ij}}, h_{ij}^{D^*_{ij}+1}, \ldots, h_{ij}^{\Gamma}\}$.

The MILP is shown in Fig. 4.6. Inequality (4.8) satisfies the traffic capacity constraints. Equation (4.9) guarantees the flow balance and makes the appropriate flow decomposition for different delay values; if some data arrived to a node n_i over the flow f_{ki}^{d} and is to be relayed to a node n_j, it should go on the flow $f_{ij}^{d+W^*_{ij}}$ (see Fig. 4.7); and if it is to be relayed to a data collector at a point p_j, it should go on the flow $h_{ij}^{d+D^*_{ij}}$. Equation (4.10) guarantees that each sensor node pushes its own data to its neighbors on the appropriate subflows; data generated at node n_i and sent to node n_j will encounter W^*_{ij} delay steps, and data generated at node n_i and sent to a data collector at a point p_j will encounter D^*_{ij} delay steps. Inequality (4.11) makes E_{min} the minimum residual energy over all sensor nodes (note that we maximize E_{min}). Equation (4.12) sets E_{total} to the total energy consumption. Inequality (4.13) guarantees that the energy expenditure of any sensor node is not more than its current residual energy. Inequality (4.14) guarantees that if no data collector is placed at p_j (i.e., $l_j = 0$), no flow is sent to p_j. Equation (4.15) satisfies the constraint that only R data collectors are available.

When the number of nodes is large, we will have a MILP with too many variables; this increases the complexity of the problem and, as a result, a significant amount of time is required to find near optimal solutions. Such a scalability problem can be dealt with by partitioning nodes and data collectors into smaller groups, which results in a set of smaller problems that can be solved separately. However, such a partitioning is out of the scope of this work.

Maximize $\alpha\, E_{min} - \beta\, E_{total}$
s.t.,

$$\sum_{j \in N(i)} \sum_{W^*_{ij} \le d < \Gamma} f^d_{ij} + \sum_{j \in M(i)} \sum_{D^*_{ij} \le d \le \Gamma} h^d_{ij} \le C_i, 0 \le i < N \tag{4.8}$$

$$\sum_{\substack{j \in N(i),\\ d \ge W^*_{ji}}} f^d_{ji} = \sum_{\substack{j \in N(i),\\ d < \Gamma - W^*_{ij}}} f^{d+W^*_{ij}}_{ij} + \sum_{\substack{j \in M(i),\\ d \le \Gamma - D^*_{ij}}} h^{d+D^*_{ij}}_{ij}, 0 \le i < N, 1 \le d < \Gamma \tag{4.9}$$

$$G_i = \sum_{\substack{j \in N(i),\\ W^*_{ij} < \Gamma}} f^{W^*_{ij}}_{ij} + \sum_{\substack{j \in M(i),\\ D^*_{ij} \le \Gamma}} h^{D^*_{ij}}_{ij}, 0 \le i < N \tag{4.10}$$

$$\begin{aligned} E_i \quad & - \sum_{j \in N(i)} \sum_{W^*_{ij} \le d < \Gamma} E_{Tr}(W_{ij})\, f^d_{ij} - \sum_{j \in M(i)} \sum_{D^*_{ij} \le d \le \Gamma} E_{Tr}(D_{ij})\, h^d_{ij} \\ & - \sum_{j \in N(i)} \sum_{W^*_{ji} \le d < \Gamma} E_{Rc}\, f^d_{ji} \ge E_{min} \quad , 0 \le i < N \end{aligned} \tag{4.11}$$

$$\begin{aligned} \sum_{0 \le i < N} \quad & (\sum_{j \in N(i)} \sum_{W^*_{ji} \le d < \Gamma} E_{Rc} f^d_{ji} + \sum_{j \in N(i)} \sum_{W^*_{ij} \le d < \Gamma} E_{Tr}(W_{ij})\, f^d_{ij} \\ & + \sum_{j \in M(i)} \sum_{D^*_{ij} \le d \le \Gamma} E_{Tr}(D_{ij}) h^d_{ij}) = E_{total} \end{aligned} \tag{4.12}$$

$$E_{min} \ge 0 \tag{4.13}$$

$$\sum_{i \in B(j)} \sum_{D^*_{ij} \le d \le \Gamma} h^d_{ij} \le l_j \sum_{0 \le i < N} G_i, 0 \le j < M \tag{4.14}$$

$$\sum_{0 \le i < M} l_i = R \tag{4.15}$$

$$l_i \in \{0, 1\}, 0 \le i < M$$

Figure 4.6: MILP formulation of the DCPR.

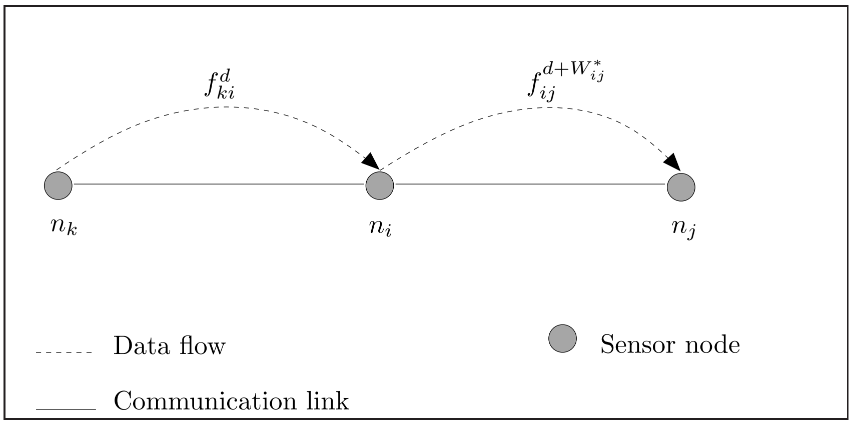


Figure 4.7: An example of a flow going from n_k to n_j through n_i.

4.5 Experimental results

We carried out two sets of experiments to study the performance of our schemes. The first set of experiments is devoted to the DTPR scheme. The novelty of our DTPR scheme comes from the mobility of the data collectors and from considering the residual energy in the objective function. Therefore, in the first set of experiments, we compare the DTPR scheme with a scheme in which data collectors are stationary, and with a mobile scheme that ignores the residual energy of different sensor nodes. The second set of experiments are devoted to the DCPR scheme. However, we are not aware of any mobile data collector scheme that prolongs the lifetime of the network under some delay constraints. So we compare the DCPR scheme with the DTPR one to see how a delay constraint would affect the lifetime and to measure the delay of the DTPR scheme.

4.5.1 DTPR experiments

We conducted some experiments to compare the DTPR scheme with two other schemes. The first one is the stationary scheme in which data collectors are stationary and are placed randomly on the surface of the water. In the stationary scheme, we use a similar linear program to find the optimal routing for a given placement. The second scheme we compare with is a mobile one in which the objective is to Minimize the Maximum (MM) amount of energy consumed by one node.

In our simulations we assume three transmission levels: 5 km, 2.5 km, and 1 km. The WHOI modem [69] transmits over 5 km at 10W with a data rate of 220 bits/sec. Other acoustic modems have slightly different energy consumption settings [57]. For simulation purposes, we set our energy consumption parameters to be compatible with those of the WHOI modem and with the general energy consumption model described in [67]. So we set $E_{Tr}(5\text{ km}) = 20$ mJ/bit, $E_{Tr}(2.5\text{ km}) = 5$ mJ/bit, $E_{Tr}(1\text{ km}) = 2$ mJ/bit, and $E_{Rc} = 1$ mJ/bit. Every sensor node has an initial energy of 2x10^5 J, and generates 64 bytes/hour. The round length is one day. The capacity of sensor nodes is uniformly distributed between 1000 and 2000 bytes/hour. Our simulations involve networks of 100 sensors randomly deployed in a 20 x 20 x 2 km^3 volume (the maximum depth is 2 km). In each network, we tested different scenarios of one, two, three, four, and five data collectors. For each scenario, we generate 20 random samples and take the average. The standard deviations of all averaged results in our simulations in this chapter were within 13.4% of the average value. To solve the MILP, we use *lp_solve* 5.5 [62] with a timeout of 20 minutes (the MILP of a particular round is solved during the last 20 minutes of the previous round).

Fig. 4.8 shows the lifetime comparison between different schemes. Fig. 4.9 shows a comparison of the average energy consumed per byte. Note that the different schemes have a similar energy consumption per byte, yet the DTPR has a much longer lifetime. This is due to the load balancing effect that is exercised in our scheme and due to the consideration made for residual energy. Using different values for the MILP solver timeout, different sensor densities, and different number of data collectors have shown similar trends.

4.5.2 Comparing the DTPR with the DCPR

In this subsection we show the results of the experiments we conducted to compare the DTPR and the DCPR schemes in terms of network lifetime and delay. When Γ is small enough, it is expected that the DTPR will have longer lifetime and longer delay as

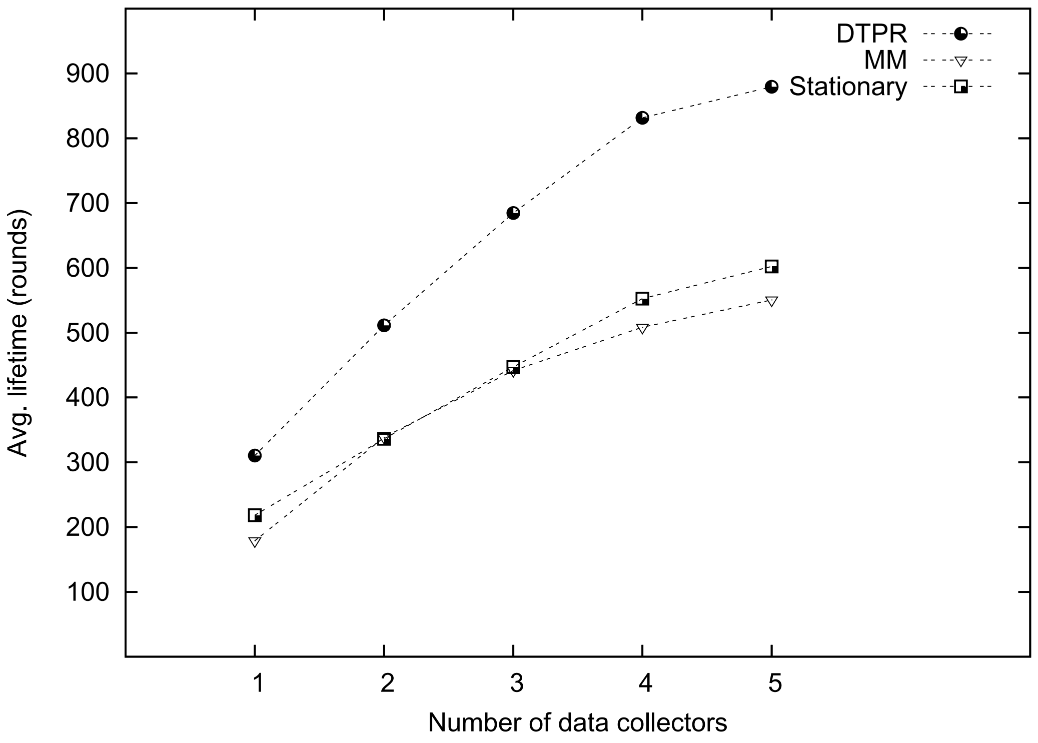


Figure 4.8: Lifetime comparison results.

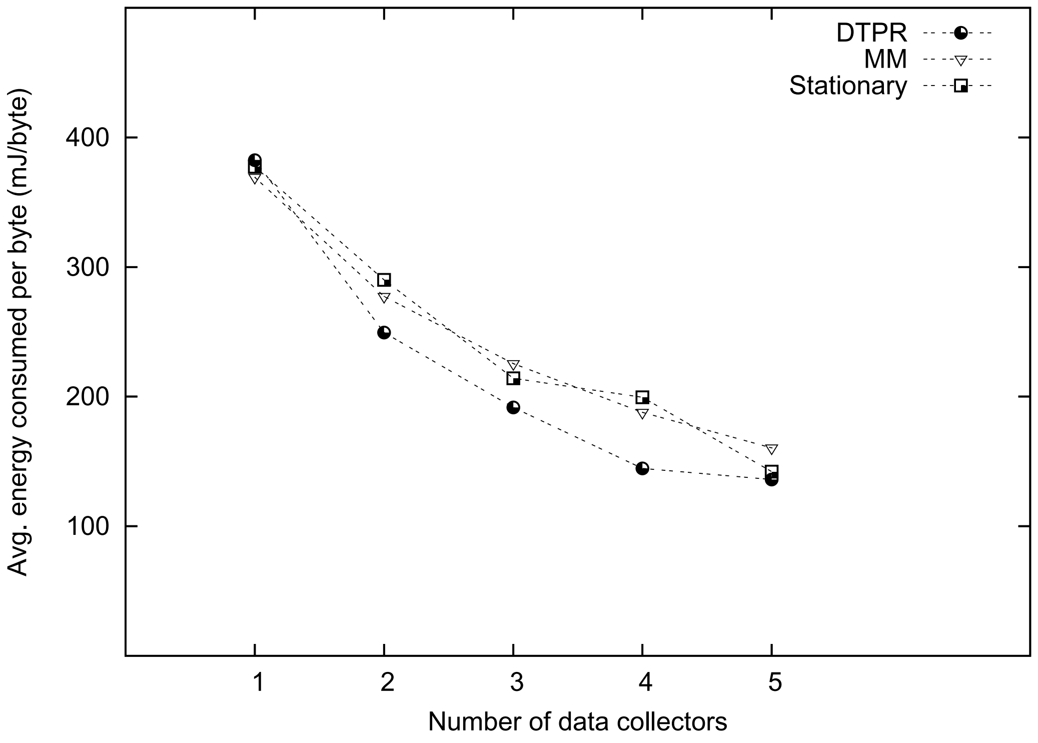


Figure 4.9: A comparison of the average energy consumption per byte.

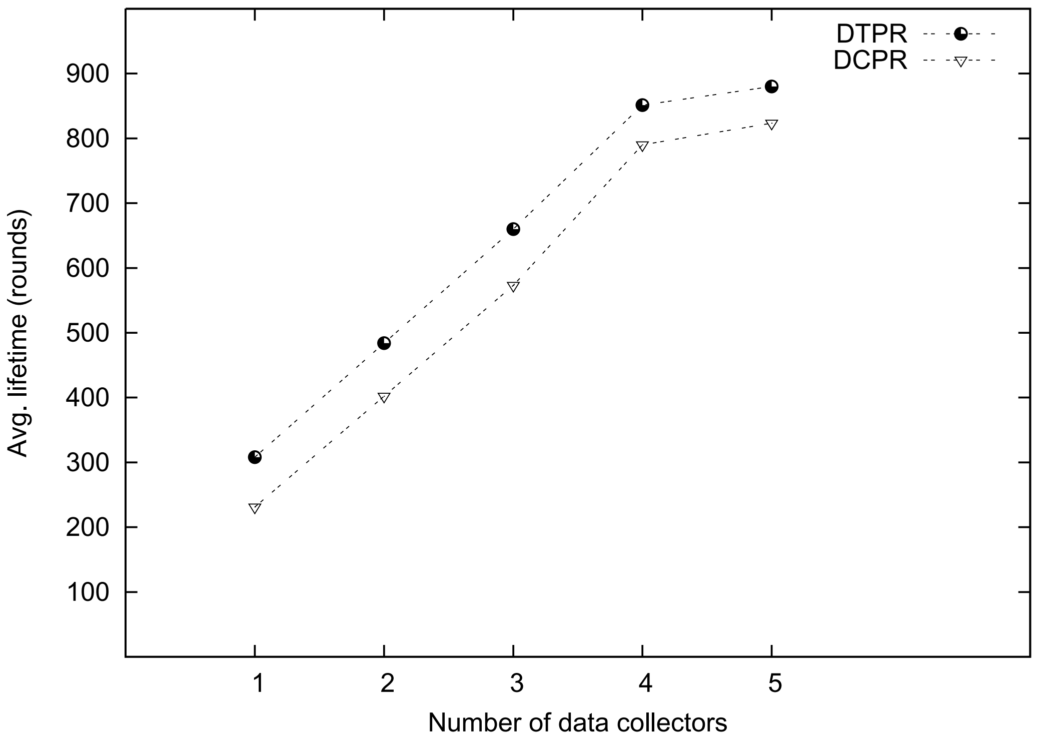


Figure 4.10: Lifetime comparison between the DTPR and the DCPR.

compared with the DCPR, and that is what our experiments have verified. The simulation settings are the same as the ones in subsection 4.5.1 with Δ being set to 1000 m. Γ being set to 22.

Fig. 4.10 shows the lifetime comparison between the two schemes. It is obvious that the DTPR provides a longer lifetime due to the relatively more freedom it has in choosing paths of any length. While the maximum delay encountered in the DCPR is bounded by Γ (i.e., 22), the maximum delay in the DTPR is more than Γ as shown in Fig. 4.11. Apparently, the disparity between the maximum delay encountered in the DTPR and that in the DCPR dwindles as more data collectors become available. That is due to the fact that having more data collectors reduces the chances of having sensor nodes for which the nearest data collector is too far away.

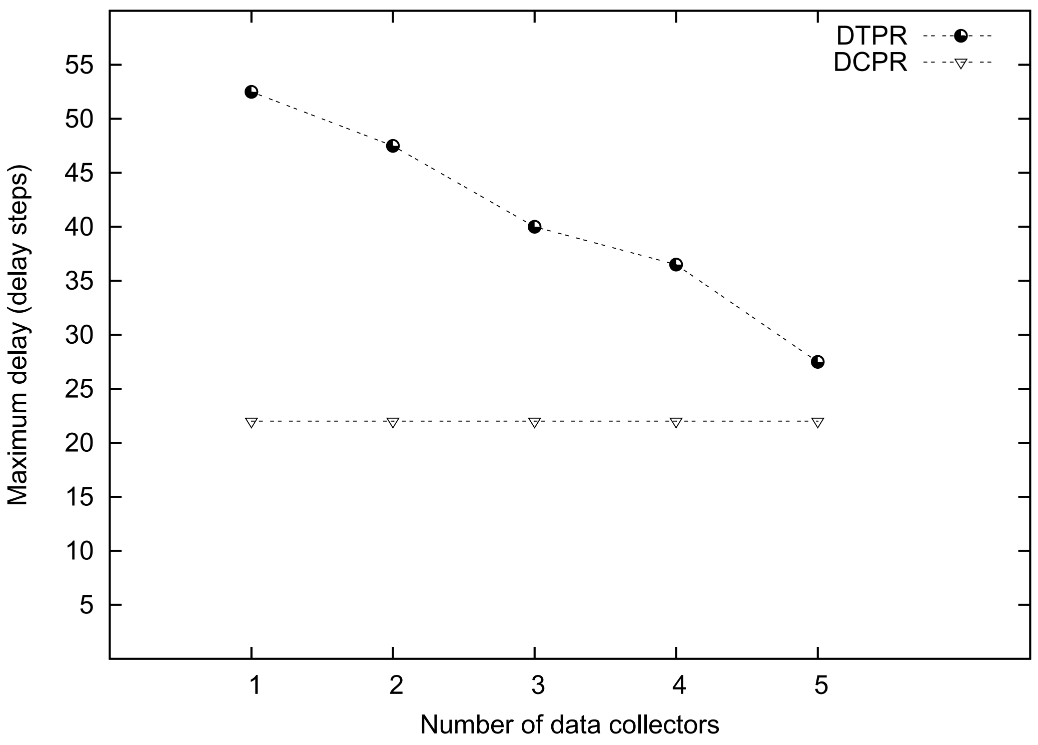


Figure 4.11: Delay comparison between the DTPR and the DCPR.

4.6 Summary

Underwater WSNs have a wide range of potential applications in environmental monitoring and oceanographic data gathering. However, due to the limited energy supply available to sensor nodes, boosting the network longevity is crucial to make the most of underwater WSNs. Besides the network lifetime, bounding the delay is very important in real-time applications, such as a Tsunami warning system.

For the sake of extending the lifetime of an underwater WSN with and without delay constraints, this chapter presents two schemes for finding near-optimal routing and placement of mobile data collectors in underwater WSNs. The novelty of the proposed schemes stems from:

1. Finding a complete, finite search space for data collector locations.

2. Solving the problem of prolonging the lifetime with a delay-based QoS guarantees.

Experimental results show that both schemes have the potential to prolong the lifetime of an underwater WSN significantly as compared with another stationary scheme and a mobile scheme that does not consider the residual energy of nodes. While the DCPR has a slightly shorter lifetime as compared to the DTPR scheme, it has the ability to meet any delay-based QoS guarantees.

Chapter 5

Distributed routing to a mobile data collector

While the schemes presented in the previous two chapters are able to find near-optimal routing and placement solutions, they are centralized and use linear programming which requires significant time to solve. In some situations, it may be more advantageous to come up with reasonable solutions, rather than superior ones, but in a distributed manner and with a lower computational complexity; this is the goal of the work in this chapter.

In this chapter, we propose a distributed scheme for data gathering using a mobile data collector in WSNs. In our scheme, a mobile data collector moves along a predefined trajectory over the sensing field and data are forwarded to nodes whose transmission disks overlap with the trajectory of the data collector; these nodes are called *trajectory nodes*. The trajectory of the data collector may be enforced by properties of the sensing terrain (e.g., a dense forest). It also makes sense to use a fixed trajectory when detailed information needed to find an optimal trajectory are highly dynamic or hard to obtain. We consider two categories of data: delay-sensitive data and delay-tolerant data. While delay-sensitive data are sent to the data collector directly, delay-tolerant data may be sent to a nearby trajectory node, where they wait for the data collector to come and pick them up.

The rest of this chapter is organized as follows. Section 5.1 outlines the proposed scheme and highlights its contribution. Section 5.2 describes the model of the sensor network under study and gives a formal problem definition. In Section 5.3, we present a lifetime and delay theoretical analysis to understand the impact of data collector mobility. Our routing scheme is described in Section 5.4. Section 5.5 shows the experimental results. Finally, Section 5.6 concludes this chapter with a brief summary.

5.1 Scheme outline and contributions

The work presented in the previous two chapters, the work in [5], and the work in [54] use linear programming to find a path for the data collector together with the routes from sensor nodes to the data collector. While able to find optimal (or near-optimal) solutions, these schemes are centralized and suffer the high computational complexity of linear programming. In the work presented in [70] and [71], sensor nodes hold their data until the data collector becomes close enough to receive the data directly in a single hop. This could bring significant energy savings at the cost of a much longer delay. The authors in [72] exploit multi-hop communication to deliver data to the data collector at its current location. However, the scheme requires sensor nodes to be synchronized with the data collector in order to determine the latter's current location. The scheme presented in [73] finds a trajectory for the data collector that balances the relaying load and conserves energy. In [74], a data collector is assumed to repeatedly move along a closed path in the sensing field, and a scheme for controlling the speed of the data collector is given. The main idea is to have the data collector move at a lower speed in regions with a high density of sensor nodes and at a higher speed in regions where there are no or few sensor nodes. However, the scheme in [73] and the one in [74] are not suitable for delay-sensitive data: data are delivered to nodes near the trajectory of the data collector where they wait for the data collector to come and pick them up.

We present a fully distributed and localized routing scheme that utilizes data collector mobility to distribute the load over different parts of the network. The data collector repeatedly moves along a globally known closed trajectory (as in [74]). A sensor node whose transmission range overlaps with the trajectory of the data collector is called a trajectory node. It is assumed that the data collector makes discrete movements and stops at designated locations to receive data from trajectory nodes[1]. Trajectory nodes receive data from other nodes and deliver them to the data collector when it becomes within their transmission range. Data are classified into two categories: delay-sensitive data and delay-tolerant data. Delay-sensitive data are sent to a trajectory node that currently can reach the data collector in a single hop. We call such a node an active trajectory node. Delay-tolerant data are sent to any trajectory node where they wait for the data collector to come and pick them up.

We give theoretical analysis to quantify the impact of data collector mobility on the network lifetime as compared to a WSN with a stationary data collector. Moreover, we use simulations to evaluate the proposed scheme in practice. Simulation results show that the proposed scheme has the potential to significantly extend the lifetime of the network.

The scheme we present here does not require any kind of synchronization between sensor nodes and the data collector. The novel contributions of the work presented in this chapter can be summarized as follows:

1. The routing scheme we present here is fully distributed in the sense that routes are created by the nodes themselves and not by a centralized entity, and it is localized in the sense that every sensor node makes its routing decisions based on local information.

2. Our scheme finds good solutions, rather than optimal ones, with a very low computational complexity (our scheme does not use any computation that takes more than a constant time).

3. Our scheme combines the advantage of a stationary data collector (i.e., short delay)

[1]One way to do so is to make the data collector stop once at each maximal overlapping segment of its trajectory. For more information about maximal overlapping segments, see Section 3.3.

and that of a mobile one (i.e., a long network lifetime) without any synchronization between the data collector and the sensor nodes.

To the best of our knowledge, this is the first scheme that supports on-line delivery of data to a mobile data collector in a distributed, localized, asynchronous fashion.

5.2 System model and problem description

We consider a sensor network of N sensor nodes uniformly deployed in a sensing field. The network is assumed to be connected (i.e., there is at least one multi-hop path between any pair of nodes). Sensor nodes collect data from the surrounding environment and report them to the data collector. The transmission range of all sensor nodes is fixed to r distance units (i.e., each sensor node has a transmission disk of radius r distance units). The topology of the network is modeled as a connectivity graph $G = (V, E)$, where $V = \{n_0, n_1, ..., n_{N-1}\}$ is the set of N sensor nodes, and $(i, j) \in E$, if sensor nodes n_i and n_j are within the transmission range of each other.

As stated before, a mobile data collector moves along a predefined trajectory that overlaps with the transmission disks of trajectory nodes. At a given time, an active trajectory node is one which is not more than r distance units away from the data collector.

We are interested in a delay-tolerant routing scheme to deliver delay-tolerant data to any trajectory node and a delay-sensitive routing scheme to deliver delay-sensitive data to an active trajectory node. The objective is to prolong the lifetime of the network, namely, the time until a particular proportion of the sensor nodes run out of energy.

5.3 Lifetime and delay analysis

Assuming a dense sensor network on a circular sensing field, we give simplified lifetime and delay comparisons between the following two scenarios:

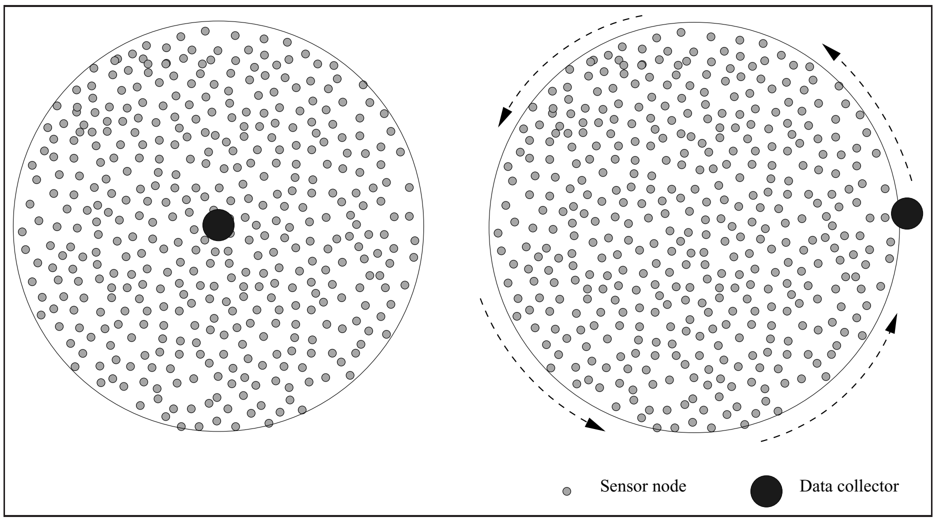


Figure 5.1: A network with a stationary data collector (left) and a network with a mobile data collector (right).

1. A network with a stationary data collector located at the center of the sensing field. All data packets are destined to the data collector.

2. A network with a mobile data collector that moves along the boundary (i.e., the perimeter) of the sensing field[2]. Data packets are forwarded to trajectory nodes, where they wait for the data collector to come and pick them up.

Figure 5.1 shows a graphical illustration of the two scenarios. To simplify our analysis, we make the following assumptions:

1. All sensor nodes have the same data generation rate; every sensor node generates D packets per time unit.

2. Sensor nodes are uniformly distributed over the sensing field.

3. The sensing field is a circle of radius R distance units.

4. Sensor nodes consume E_{Tr} energy units to send one packet.

5. Every sensor node starts with an energy supply of E_{Init} energy units.

[2]Our scheme is applicable to trajectories of any shape. However, the assumption of a circular trajectory is made to simplify the analysis.

5.3.1 Lifetime comparison

Here we compare the two scenarios in terms of network lifetime. Network lifetime is defined as the time until the data collector is unreachable. Our comparison assumes that a perfect load balancing is exercised in both scenarios (i.e., the load is evenly distributed over trajectory nodes). We also ignore the energy consumed to receive data; transmission is known to be the dominant energy consuming operation [4].

A stationary data collector

With multi-hop communication, the lifetime of the network is determined by the lifetime of trajectory nodes (i.e., nodes whose distance to the data collector is not more than r distance units). Since sensor nodes are uniformly distributed over the sensing field, the expected number of trajectory nodes is

$$\frac{N\pi r^2}{\pi R^2} = \frac{Nr^2}{R^2} \quad .$$

Trajectory nodes are in charge of delivering all data generated all over the network to the data collector. Therefore, they transmit DN packets to the data collector. With a perfect load balancing, each trajectory node will transmit

$$\frac{DNR^2}{Nr^2} = \frac{DR^2}{r^2} \text{ packets.}$$

Thus, the lifetime of a trajectory node is

$$\left\lfloor \frac{E_{Init} r^2}{DR^2 E_{Tr}} \right\rfloor \text{ time units.}$$

A mobile data collector

Since the data collector moves along the perimeter of the sensing field, the expected number of trajectory nodes is

$$\frac{\pi(R^2-(R-r)^2)N}{\pi R^2}=\frac{(2Rr-r^2)N}{R^2} \quad .$$

With a perfect load balancing and constant data generation rate, each trajectory node will be in charge of transmitting

$$\frac{DNR^2}{(2Rr-r^2)N}=\frac{DR^2}{(2Rr-r^2)} \text{ packets.}$$

Thus, the lifetime of a trajectory node is

$$\left\lfloor \frac{E_{Init}(2Rr-r^2)}{DR^2E_{Tr}} \right\rfloor \text{ time units.}$$

From this comparison, the lifetime of a network with a mobile data collector is longer than that of a network with a stationary data collector by a factor of

$$\frac{2R-r}{r} \quad .$$

For example, if $r = 100$ m and $R = 1000$ m, the lifetime of a network with a mobile data collector is 19 times longer than the lifetime of the same network with a stationary data collector.

5.3.2 Delay comparison

The delay a packet encounters is proportional to the number of hops between the packet's source and destination. The number of hops between two points can be approximated to be a linear function of the Euclidean distance between them. Therefore, the delay a packet encounters can be expressed by

$$delay = \alpha L,$$

where α is a constant and L is the Euclidean distance between the packet's source and destination.

With a stationary data collector located at the center of the sensing field, the maximum delay a packet may encounter occurs when the source is at the perimeter of the sensing field; this results in a delay of

$$\alpha R \text{ time units.}$$

On the other hand, the maximum delay in a network with a mobile data collector is significantly worse. With a mobile data collector, the worst case occurs when a packet arrives to a trajectory node which has just been left by the data collector; such a packet needs to wait for the data collector to complete a full round along the perimeter of the sensing field which depends on the speed of the data collector and on other factors. It is obvious that such a delay is much longer than that of a network with a stationary data collector.

To summarize, having a mobile data collector has a great potential to prolong the lifetime of the network, yet it suffers longer delay as compared with a stationary data collector. Accordingly, we are motivated to come up with a scheme that combines the lifetime of a mobile data collector network and a delay comparable to that of a stationary data collector network.

5.4 Routing scheme

The scheme we propose here takes in consideration two types of data: delay-tolerant data and delay-sensitive data. Accordingly, sensor nodes use two routing schemes to deliver their data to the data collector: delay-tolerant routing and delay-sensitive routing. In the delay-tolerant routing, data packets are sent to any trajectory node where they wait for the data collector to come and pick them up. On the other hand, when a node has delay-sensitive data, it needs first to locate the data collector in order to send the data to an active trajectory node.

5.4.1 Delay-tolerant routing

In order for sensor nodes to deliver their delay-tolerant data to the data collector, they need to have a path to at least one trajectory node. To do so, trajectory nodes broadcast their identity at the deployment stage and each sensor node keeps a record of the next hop towards some trajectory node. Each sensor node n_i has a Trajectory Node Record (TNR_i) which has the following fields:

id: the id of the trajectory node to which delay-tolerant data will be sent.

$next\_hop$: a neighbor of n_i which is used as a next hop towards the trajectory node.

$number\_of\_hops$: the number of hops to the trajectory node.

Algorithm 9 describes the process of setting the trajectory node records of all sensor nodes, assuming that each sensor node uses the nearest trajectory node in terms of number of hops. Note that this process will construct a tree for each trajectory node; the tree of a trajectory node n_i is rooted at n_i and involves all sensor nodes whose nearest trajectory node is n_i. After this initialization process, TNRs are used to forward delay-tolerant data to trajectory nodes.

To simplify the presentation of our scheme, we assume one TNR for each sensor node. However, each sensor node can maintain multiple TNRs to recover failures of trajectory nodes and changes in the topology of the network.

Algorithm 9: Setting trajectory node records.

```
foreach sensor node $n_i$ do
    if $n_i$ is a trajectory node then
        $TNR_i$.id=$i$;
        $TNR_i.next\_hop$=$i$;
        $TNR_i.number\_of\_hops$=0;
        broadcast $TNR_i$ to all neighbors of $n_i$ ;
    else
        $TNR_i.number\_of\_hops$=N+1;
    end
end
when a sensor node $n_i$ receives a broadcasted $TNR_j$ :
    if $TNR_j.number\_of\_hops + 1 < TNR_i.number\_of\_hops$ then
        $TNR_i.number\_of\_hops = TNR_j.number\_of\_hops + 1$;
        $TNR_i.id = TNR_j.id$;
        $TNR_i.next\_hop = j$;
        $n_i$ broadcasts $TNR_i$ to all of its neighbors;
    end
end
```

5.4.2 Delay-sensitive routing

Our delay-sensitive routing has two phases: locating the data collector and forwarding data to an active trajectory node. To facilitate locating the data collector, announcements about the current location of the data collector are disseminated periodically to a subset of sensor nodes in the network. When a sensor node has delay-sensitive data, it sends a query to a subset of sensor nodes in the network. To guarantee that the query reaches a node which has received the most recent announcement, we use a simple and reliable direction based approach: announcements cross the network vertically (i.e., north to south) and queries cross the network horizontally (i.e., east to west) as shown in Fig. 5.2. A similar approach was used in the context of information discovery in large networks [75]. We assume that sensor nodes know their geographical location in a 2D space.

Announcements and queries are disseminated using GPSR [39]. GPSR uses a planar graph $G'(V', E')$ which is a subgraph of the connectivity graph $G(V, E)$ and has the same set of vertices as G (i.e., $V' = V$ and $E' \subset E$). G' partitions the space into several closed faces and one open external face. *Border nodes* are those nodes that form the external

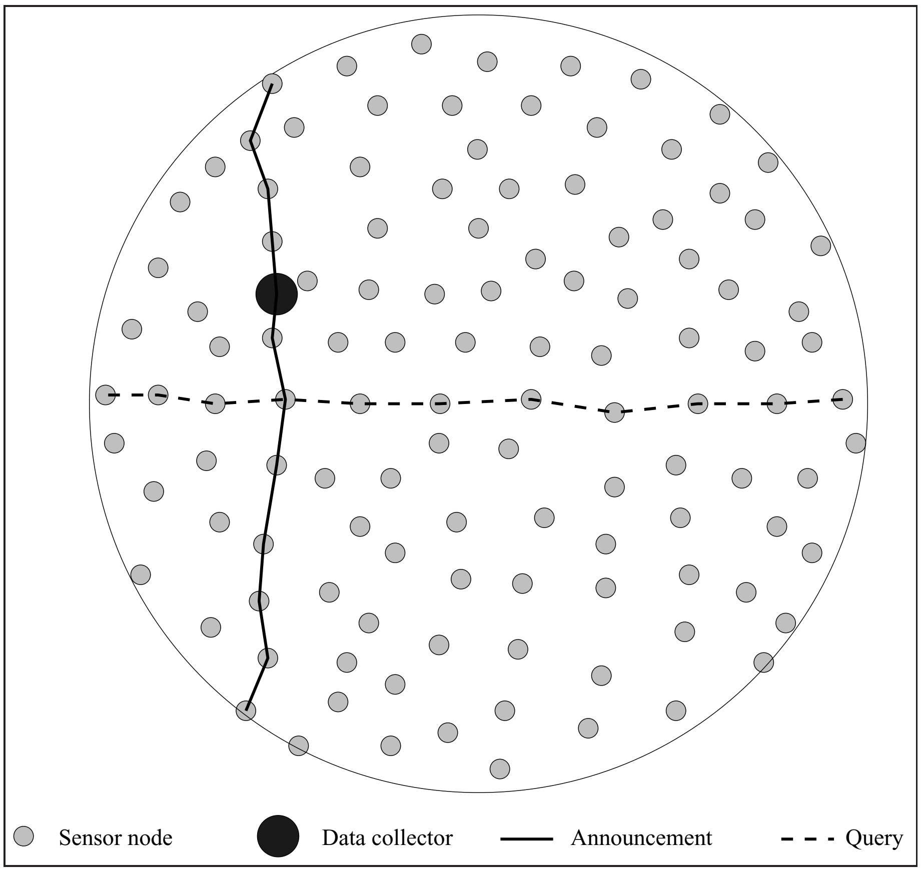


Figure 5.2: dissemination of queries and announcements.

face. We exploit a nice property of GPSR: when a packet is sent to an arbitrary location in the external face, it is delivered to a border node which is the closest to that location. Thereby, to send a query from a node at location (x, y) to the west, a query is sent to the location $(x - \alpha, y)$ where α is the diameter of the network[3] (i.e., $(x - \alpha, y)$ is outside the sensing field and to the west of it). We call such a point an *extreme point*. Since no sensor node is located at that location, GPSR will deliver the query to a node at the west border of the network. To send a message to the north, the east, or the south, we use the extreme points $(x, y + \alpha)$, $(x + \alpha, y)$, $(x, y - \alpha)$, respectively.

An announcement message has three fields: *loc*, *time_stamp*, and *destination*. *loc* is the current location of the data collector in the form of (x, y), *time_stamp* is the time at which the announcement is made according to the data collector clock, and *destination* is either $(x, y + \alpha)$ or $(x, y - \alpha)$. Every sensor node maintains a Data Collector Record

[3]The diameter of the network is the maximum Euclidean distance between two nodes in the network.

(DCR) to store the most up-to-date information it knows about the location of the data collector. DCR has two fields: *loc* and *time_stamp*. Since all transmissions of the announcement messages are made through a wireless medium, all neighboring nodes of the sender will observe (hear) the announcement and update their DCRs accordingly. When an announcement message arrives to its final destination (i.e., a border node), that node broadcasts the announcement for the last time to tell its neighbors about the new information.

A query has two fields: *source* and *destination*, where *source* is the location of the sender, say (x, y), and *destination* is either $(x - \alpha, y)$ or $(x + \alpha, y)$. When a query Q arrives to its destination, a reply R is sent back to $Q.source$. A reply has three fields: *destination*, *loc*, and *time_stamp*, where *destination* is the location of a node that initiated the query, *loc* is the location of the data collector according to the most recent announcement seen by the reply message, and *time_stamp* is the time stamp associated with that location. As it moves along its path to $Q.source$, R carries the most recent location of the data collector it has encountered. When R arrives to a node on its way to $Q.source$, it exchanges information about the location of the data collector with the DCR of that node before being forwarded to the next hop.

To summarize, whenever the data collector changes its location, it will disseminate announcements about the new location vertically, and when a sensor node has delay-sensitive data, it sends a query horizontally, waits for the replies to come back carrying the location of the data collector, and then sends the data to the data collector. This protocol is described in Algorithm 10. To show the correctness of this protocol, we need to show that when a query Q is initiated after an announcement A has been disseminated, Q's reply R is guaranteed to reach a node that has received A. Let $U(R)$ denote the subset of nodes that have received a reply R, let $W(A)$ denote the subset of nodes that have received an announcement A, and let $\overline{W(A)}$ denote the subset of nodes that have at least one neighbor in $W(A)$ (i.e., $\overline{W(A)} = \{n_i : \exists n_j, n_j \in W(A) \wedge (n_i, n_j) \in E\}$). Note that all nodes in $W(A) \cup \overline{W(A)}$ have heard A because of the broadcast nature of the wireless

Algorithm 10: Disseminating announcements, queries, and replies (continued next page).

```
when a sensor node $n_i$ receives or observes an announcement $A$ :
    $DCR_i.time\_stamp = A.time\_stamp$;
    $DCR_i.loc = A.loc$;
    if $A$ was forwarded to $n_i$ (i.e., $n_i \in W(A)$) then
        if $n_i$ is the final destination for $A$ then
            $n_i$ broadcasts $A$ so that $A$ is observed by all of $n_i$'s neighbors;
        else
            To reach $A.destination$, $n_i$ forwards $A$ to one of its neighbors according
            to GPSR;
        end
    end
end
when a sensor node $n_i$ receives a query $Q$ :
    if $n_i$ is the final destination for $Q$ then
        $n_i$ initiates a reply $R$;
        $R.time\_stamp = DCR_i.time\_stamp$;
        $R.loc = DCR_i.loc$;
        $R.destination = Q.source$;
        To reach $R.destination$, $n_i$ forwards $R$ to one of its neighbors according to
        GPSR;
    else
        To reach $Q.destination$, $n_i$ forwards $Q$ to one of its neighbors according to
        GPSR;
    end
end
when a sensor node $n_i$ receives a reply $R$ :
    if $DCR_i.time\_stamp > R.time\_stamp$ then
        $R.time\_stamp = DCR_i.time\_stamp$;
        $R.loc = DCR_i.loc$;
    else
        $DCR_i.time\_stamp = R.time\_stamp$;
        $DCR_i.loc = R.loc$;
    end
    if $n_i$ is not the final destination for $R$ then
        To reach $R.destination$, $n_i$ forwards $R$ to one of its neighbors according to
        GPSR;
    end
end
```

Algorithm 10 continued

```
when a sensor node $n_i$, that has sent a query, receives two replies :
    $n_i$ sends its delay-sensitive data to $DCR_i.loc$ using GPSR;
end
when the data collector moves to a new location loc :
    The data collector initiates an announcement $A$;
    $A.time\_stamp$ =the current time according to the data collector's clock;
    $A.loc = loc$;
    $A.destination$ = a north extreme point;
    The data collector picks an active trajectory node $n_i$;
    The data collector sends $A$ to $n_i$;
    $A.destination$ = a south extreme point;
    The data collector sends $A$ to $n_i$;
end
```

communication. We state the following theorem to prove the correctness of the protocol described above.

Theorem 4 *If announcements, queries, and replies are disseminated as described in Algorithm 10, for any pair of an announcement A and a reply R,*
$U(R) \cap (W(A) \cup \overline{W(A)}) \neq \phi$.

Proof Let $|ab|$ denote the Euclidean distance between two points[4]. Let τ denote the path connecting the north extreme point, all sensor nodes in $W(A)$, and the south extreme point for an announcement A. And let ψ denote the path connecting the east extreme point, all sensor nodes in $U(R)$, and the west extreme point for a reply R. Fig. 5.3 shows an example of such paths. τ is connecting a north extreme point and a south extreme point through the sensing field, and ψ is connecting an east extreme point and a west extreme point through the sensing field. Therefore, τ and ψ must intersect, and their intersection may occur at a sensor node, at a line segment connecting two sensor nodes, or at a line segment connecting a sensor node and an extreme point. Fig. 5.4 shows all possible cases for the intersection of τ and ψ. We prove this theorem in each of these cases as follows:

[4]n_i, which we usually use to refer to a sensor node, may also be used to refer to the point where that sensor node is located.

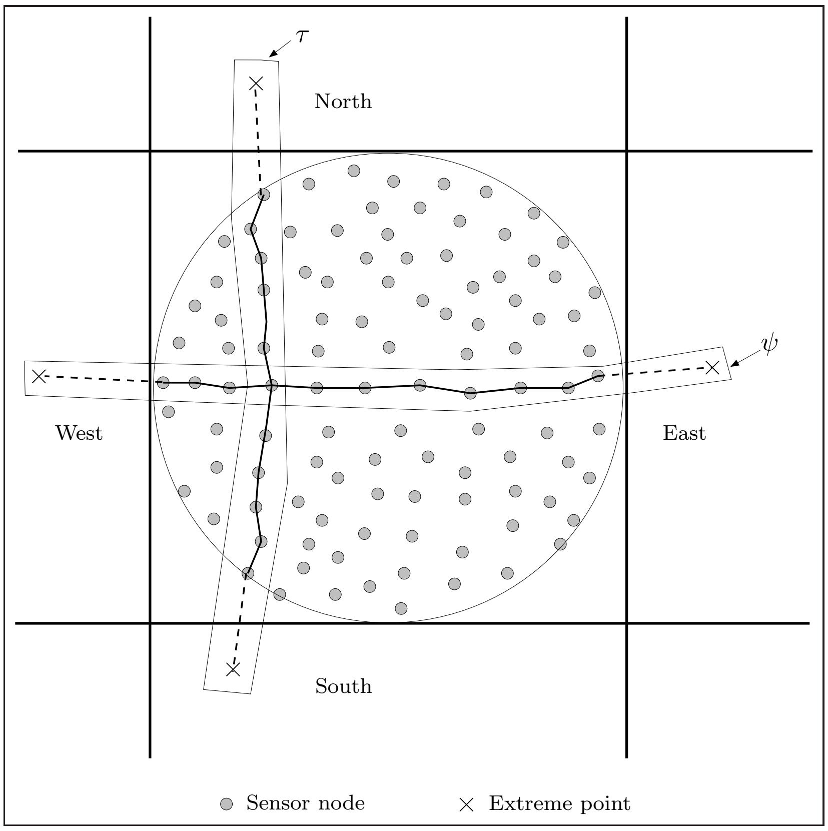


Figure 5.3: τ and ψ.

(a) The intersection occurs at a sensor node n_i. Therefore, $n_i \in U(R)$ and $n_i \in W(A)$, and this is sufficient to make $U(R) \cap (W(A) \cup \overline{W(A)}) \neq \phi$

(b) The intersection occurs at a point p which is the intersection of a line segment of τ, connecting two sensor nodes n_i and n_k, and a line segment of ψ, connecting a sensor node n_j and an extreme point e_j. Let us compare $|pn_i|$ and $|pn_j|$. If $|pn_i| < |pn_j|$, n_i is closer to e_j than n_j, which contradicts with the fact that GPSR delivers the query to the sensor node which is the closest to e_j. If $|pn_i| \geq |pn_j|$, $|n_jn_k| \leq |n_in_k|$, which means that n_j and n_k are neighbors. Therefore, $n_j \in U(R)$ and $n_j \in \overline{W(A)}$, and this is sufficient to make $U(R) \cap (W(A) \cup \overline{W(A)}) \neq \phi$.

(c) The intersection occurs at a point p which is the intersection of a line segment of τ, connecting a sensor node n_j and an extreme point e_j, and a line segment of ψ, connecting two sensor nodes n_i and n_k . This case is similar to case (b). Let us compare $|pn_i|$ and $|pn_j|$. If $|pn_i| < |pn_j|$, n_i is closer to e_j than n_j, which contradicts

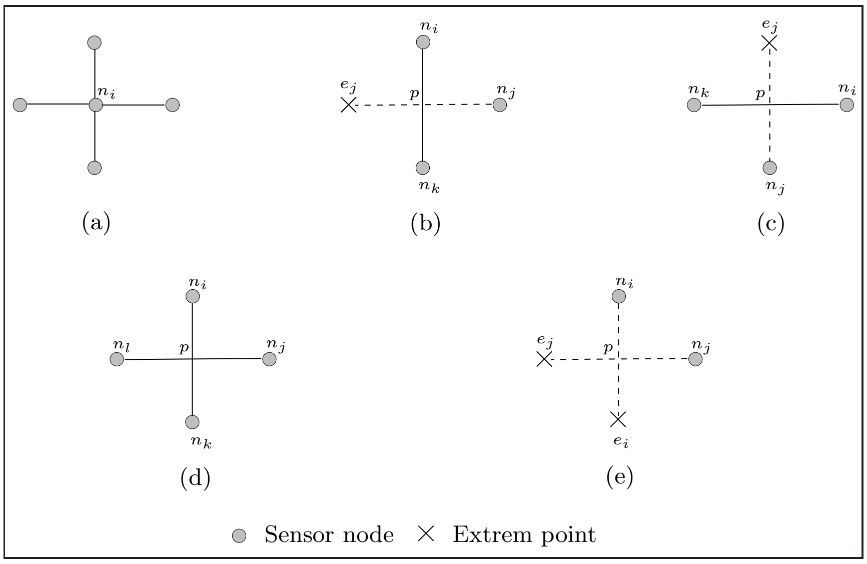


Figure 5.4: τ and ψ intersection cases.

with the fact that GPSR delivers the announcement to the sensor node which is the closest to e_j. If $|pn_i| \geq |pn_j|$, $|n_jn_k| \leq |n_in_k|$, which means that n_j and n_k are neighbors. Therefore, $n_k \in U(R)$ and $n_k \in \overline{W(A)}$, and this is sufficient to make $U(R) \cap (W(A) \cup \overline{W(A)}) \neq \phi$.

(d) The intersection occurs at a point p which is the intersection of a line segment of τ, connecting two sensor nodes n_i and n_k , and a line segment of ψ, connecting two sensor nodes n_j and n_l . Let us compare $|pn_i|$ and $|pn_j|$. If $|pn_i| < |pn_j|$, $|n_in_l| < |n_jn_l|$, which means that n_i and n_l are neighbors. Therefore, $n_l \in U(R)$ and $n_l \in \overline{W(A)}$, and this is sufficient to make $U(R) \cap (W(A) \cup \overline{W(A)}) \neq \phi$. If $|pn_i| \geq |pn_j|$, $|n_jn_k| \leq |n_in_k|$, which means that n_j and n_k are neighbors. Therefore, $n_j \in U(R)$ and $n_j \in \overline{W(A)}$, and this is sufficient to make $U(R) \cap (W(A) \cup \overline{W(A)}) \neq \phi$.

(e) The intersection occurs at a point p which is the intersection of a line segment of τ, connecting a sensor node n_i and an extreme point e_i, and a line segment of ψ, connecting a sensor node n_j and an extreme point e_j. Let us compare $|pn_i|$ and $|pn_j|$. If $|pn_i| < |pn_j|$, n_i is closer to e_j than n_j, which contradicts with the fact that GPSR delivers the query to the sensor node which is the closest to e_j. If $|pn_i| \geq |pn_j|$, n_j is closer to e_i than n_i, which contradicts with the fact that GPSR delivers the announcement to the sensor node which is the closest to e_i. Therefore, this case is impossible to occur.□

5.4.3 Implementation issues

Since an announcement or a query message is destined to a location outside the sensing field (i.e., an extreme point), GPSR delivers it to the nearest boundary node. However, that requires the message to traverse the external face before realizing that the destination is unreachable. This results in a large number of unnecessary transmissions. To resolve this issue, we make a slight modification to GPSR: when a message, originating at a node n_i, is destined to an extreme point, it is delivered to the nearest boundary node n_j, and the location of n_j is returned to n_i. Subsequent announcement or query messages from n_i are destined to n_j directly, rather than to an extreme point, to avoid traversing the external face.

Another practical concern here is what if a packet, carrying delay-sensitive data, arrives to an active trajectory node that has just been left by the data collector and is not active any more. This can be dealt with using a mechanism proposed in [76]. The main idea is to forward late data from the previous active trajectory node to one of the current active ones using some intermediate nodes. This can be facilitated by having the data collector leaving imprints and gradients at active trajectory nodes. This enables trajectory nodes, which are not active but are in the vicinity of the data collector, to track the data collector. Another alternative is to inform the source of the data to locate the data collector and resend its data again.

5.5 Experimental results

We compare the proposed scheme with a static scheme that has a stationary data collector. We also show the difference between the network lifetime of our scheme and the optimal network lifetime obtained using linear programming in a centralized fashion (as shown in Chapter 3). In the static scheme, a data collector is placed in the center of the sensing field, and sensor nodes use shortest path routing to deliver their data to the data collector.

We use the general energy consumption model described in Section 3.4. The packet size is 512 bits. Every sensor node has an initial energy of 50 J and generates 150 packets/round. 50% of the data is delay-sensitive and 50% is delay-tolerant.

Our simulations involve networks of size 200, 400, 600, 800, and 1000 sensor nodes randomly deployed in a 300 x 300 m^2, 400 x 400 m^2, 500 x 500 m^2, 600 x 600 m^2, 700 x 700 m^2 fields, respectively. For each network size, we test 20 instances and take the average. The standard deviations of all averaged results in our simulations in this chapter were within 12.2% of the average value.

To generate a trajectory for the data collector, we use a simple method. We divide the sensing field in to four equal-size squares; the trajectory of the data collector is a quadrilateral that has a vertex inside each square.

Fig. 5.5 shows a comparison of network lifetime of the three schemes. The lifetime of a network with a mobile data collector is at least 3 times longer than that of a network with a stationary data collector. The ratio of the network lifetime using a mobile data collector to that using a stationary one is proportional to the size of the network as shown in Fig. 5.6. This is because the number of trajectory nodes using a stationary data collector is the same regardless of the size of the network. This makes the load assigned to each trajectory node proportional to the size of the network. On the other hand, the number of trajectory nodes with a mobile data collector is proportional to the size of the network[5]; this means that with a larger network, the increase in the amount of data generated over the network is confronted with a similar increase in the number of trajectory nodes.

Fig. 5.7 shows a comparison of the average energy consumed per bit between different schemes. Besides load balancing, data collector mobility brings a decrease in the total

[5]Here we assume that the size of the data collector trajectory is proportional to the size of the sensing field.

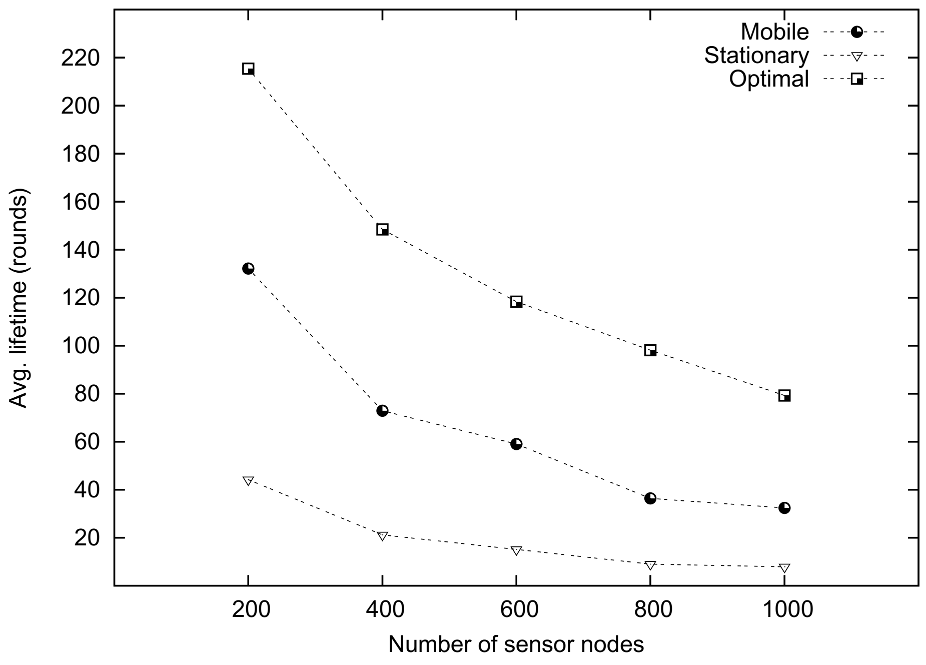


Figure 5.5: Lifetime comparison.

consumed energy. This is a direct result of having more trajectory nodes distributed over the sensing field; this results in sending delay-tolerant data over shorter routes.

The network lifetime of our distributed scheme is 50% to 75% of that of the centralized, optimal scheme. That is because of the careful placement of the data collector experienced in the optimal scheme. Furthermore, the optimal scheme is centralized, i.e., all information about the network is available to a centralized entity where the optimal routing and placement of the data collector is computed, and it is assumed that the optimal routes are delivered to sensor nodes at no cost.

5.6 Summary

For the sake of load balancing and lifetime prolonging in WSNs, we propose a scheme for data gathering using a mobile data collector. The novelty of our scheme stems from its distributed nature and from exploiting data collector mobility with a reasonable data delivery delay. Furthermore, we do not assume any time synchronization between the data collector and the sensor nodes. The low complexity of our scheme, as compared

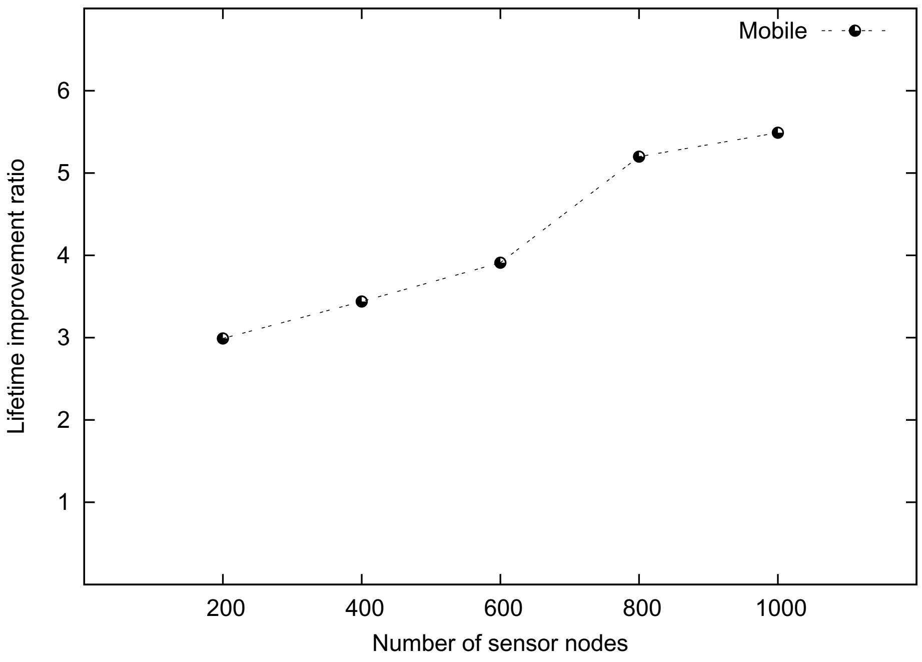


Figure 5.6: Lifetime improvement ratio of data collector mobility.

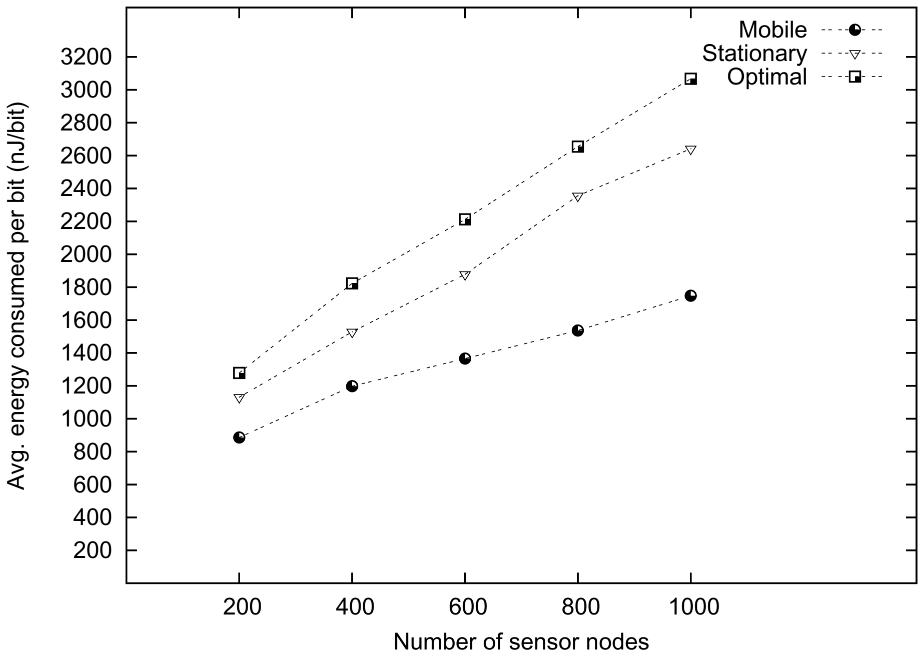


Figure 5.7: Energy consumption comparison.

to other schemes that use comprehensive optimization tools (e.g., linear programming), makes it easy to implement in ad hoc WSNs. Simulation results show that our scheme has the potential to prolong the lifetime of a WSN significantly as compared with a classical WSN with a stationary data collector.

Chapter 6

Conclusion

The last decade has witnessed a growing interest in WSNs due to their unique potential in a wide range of environmental, scientific, and civilian applications. Sensor nodes are known to be untethered in terms of power and communication. While that is the key for enabling this bunch of applications, it brings several challenges in the hardware and software design of these sensor nodes. One of these challenges is the limited energy supply and, hence, the limited lifetime of these battery-operated devices. The multi-hop relaying, which is a direct result from the energy constraints, causes the energy expenditure to vary significantly over different parts of the network; nodes closer to the data collector do much more work and run out of energy much faster than other nodes. A promising solution to this problem is to employ multiple data collectors and to change their locations periodically to distribute the load evenly among different parts of the network. The work presented in this thesis strives to extend the network lifetime through exploiting mobile data collectors.

Schemes presented in this thesis are summarized in Section 6.1. Section 6.2 highlights some future research directions.

6.1 Summary

The schemes presented in Chapter 3, Chapter 4, and Chapter 5 aim at prolonging the lifetime of a WSN by finding a near-optimal placement of the mobile data collectors and/or by setting the routing paths to deliver data from the stationary sensor nodes to the mobile data collectors. These schemes differ in the assumptions about the sensing field, in the assumptions about the transmission range of sensor nodes, in being delay-tolerant or delay-sensitive, and in being distributed or centralized.

In Chapter 3, two schemes for routing and placement of mobile data collectors in 2D terrestrial WSNs are proposed. The first one places data collectors on predefined tracks spanning the sensing field, and the second one places them at any location in the sensing field. Both schemes use linear programming in a centralized fashion. All sensor nodes are assumed to have the same fixed transmission range.

In Chapter 4, two schemes for routing and placement of mobile data collectors in 3D underwater WSNs are proposed. In both schemes, mobile data collectors are placed on the surface of the water, yet sensor nodes float at different depths underwater. Sensor nodes are assumed to have multiple discrete transmission ranges. The first scheme is delay-tolerant in the sense that it strives to prolong the network lifetime without any constraints on the delay. The second scheme is delay-sensitive; it aims at prolonging the network lifetime while maintaining an upper bound on the data delivery delay. Both schemes use linear programming in a centralized fashion.

One of the problems we solved in this thesis is discretizing the search space for data collectors: for the placement problems defined in Chapter 3 and Chapter 4, we devise an algorithm to find a relatively small set of points that embodies an optimal placement (we call such a set a complete set). Once a complete set is found, finding the optimal placement is formulated as a MILP.

In Chapter 5, we present a distributed scheme for routing data from sensor nodes to a mobile data collector on a globally known trajectory. This scheme employs two routing protocols: delay-tolerant routing and delay-sensitive routing. The former is used to deliver delay-tolerant data to a nearby trajectory node where they wait for the data collector to come and pick them up. The latter is used to locate the data collector and to deliver delay-sensitive data to the data collector in its current location.

6.2 Future work

Several future research problems can be derived from our work thus far. In this section, we point out some of these problems.

1. Current proposals for data collectors placement suffer either a high computational complexity or a low performance in practice. A low complexity heuristic that is capable of finding good placements (rather than optimal ones) is missing.

2. We used the maximal overlapping regions of a set of disks to find complete sets in 2D spaces. A more challenging problem is finding the maximal overlapping regions of a set of spheres in a 3D space. That is useful when data collectors are allowed to go underwater rather than being limited to the surface of the water.

3. In Chapter 5, it is assumed that the trajectory of the data collector is given. An interesting problem is finding a good trajectory given the data generation rates of sensor nodes and the routing protocol they are using.

4. The scheme in Chapter 5 can be modified as follows. Sensor nodes send requests to the data collector to come and pick the delay-tolerant data rather than using a fixed trajectory for the data collector. In this case, a good route for the data collector to visit nodes which have sent data pick-up requests is needed.

Bibliography

[1] http://www.xbow.com. February 25, 2009.

[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “A survey on sensor networks,” *IEEE Personal Communications Magazine*, vol. 40, no. 8, pp. 102–114, 2002.

[3] G. Pottie and W. Kaiser, “Wireless integrated network sensors,” *Communications of the ACM*, vol. 43, no. 5, pp. 51–58, 2000.

[4] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “Energy-efficient communication protocol for wireless microsensor networks,” in *Proc. The 33rd Annual Hawaii International Conference on System Sciences*, January 2000.

[5] S. Gandham, M. Dawande, R. Prakash, and S. Venkatesan, “Energy efficient schemes for wireless sensor networks with multiple mobile base stations,” in *Proc. The IEEE Global Telecommunications Conference (GLOBECOM)*, December 2003.

[6] S. Olariu and I. Stojmenovic, “Design guidelines for maximizing lifetime and avoiding energy holes in sensor networks with uniform distribution and uniform reporting,” in *Proc. The 25th IEEE International Conference on Computer Communications (INFOCOM)*, April 2006.

[7] K. Akkaya, M. Younis, and W. Youssef, “Positioning of base stations in wireless sensor networks,” *IEEE Communications Magazine*, vol. 45, no. 4, pp. 96–102, April 2007.

[8] K. Akkaya, M. Younis, and M. Bangad, "Sink repositioning for enhanced performance in wireless sensor networks," *Elsevier Computer Networks*, vol. 49, no. 4, pp. 512–534, 2005.

[9] IEEE 802.15.4 WPAN-LR Task Group. http://www.ieee802.org/15/pub/TG4.html. February 25, 2009.

[10] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. S. J. Pister, "System architecture directions for networked sensors," in *Proc. Architectural Support for Programming Languages and Operating Systems*, November 2000.

[11] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proc. the 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, September 2002.

[12] P. Zhang, C. M. Sadler, S. A. Lyon, and M. Martonosi, "Hardware design experiences in zebranet," in *Proc. The 2nd International Conference on Embedded Networked Sensor Systemss (SenSys)*, November 2004.

[13] S. Helal, W. Mann, H. El-Zabadani, J. King, Y. Kaddoura, and E. Jansen, "The gator tech smart house: a programmable pervasive space," *Computer*, vol. 38, no. 3, pp. 50–60, March 2005.

[14] A. Ailamaki, C. Faloutos, P. S. Fischbeck, M. J. Small, and J. VanBriesen, "An environmental sensor network to determine drinking water quality and security," *SIGMOD Record*, vol. 32, no. 4, pp. 47–52, 2003.

[15] R. Cardell-Oliver, M. Kranz, K. Smettem, and K. Mayer, "A reactive soil moisture sensor network: Design and field evaluation," *International Journal of Distributed Sensor Networks*, vol. 1, pp. 149–162, 2005.

[16] V. Mehta and M. ElZarki, "A bluetooth based sensor network for civil infrastructure health monitoring," *Wireless Networks*, vol. 10, no. 4, pp. 401–412, 2004.

[17] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC protocols for wireless sensor networks: a survey," *IEEE Communications Magazine*, vol. 44, no. 4, pp. 115–121, 2006.

[18] V. Rajendran and K. Obraczka, "Energy-efficient, collision-free medium access control for wireless sensor networks," in *Proc. The 1st International Conference on Embedded Networked Sensor Systems,(SensSys)*, November 2003.

[19] C. Enz, A. El-Hoiydi, J.-D. Decotignie, and V. Peiris, "Wisenet: an ultralow-power wireless sensor network solution," *IEEE Computer*, vol. 37, no. 8, pp. 62–70, 2004.

[20] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 12, no. 3, pp. 493–506, 2004.

[21] P. Lin, C. Qiao, and X. Wang, "Medium access control with a dynamic duty cycle for sensor networks," in *Proc. Wireless Communications and Networking Conference (WCNC)*, March 2004.

[22] J. Al-Karaki and A. Kamal, "Routing techniques in wireless sensor networks: a survey," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 6–28, 2004.

[23] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Ad Hoc Networks*, vol. 3, pp. 325–349, 2005.

[24] N. Thepvilojanapong, Y. Tobe, and K. Sezaki, "On the construction of efficient data gathering tree in wireless sensor networks," in *Proc. The IEEE International Symposium on Circuits and Systems*, May 2005.

[25] F. Ye, A. Chen, S. Lu, and L. Zhang, "A scalable solution to minimum cost forwarding in large sensor networks," in *Proc. The 10th International Conference on Computer Communications and Networks*, October 2001.

[26] C. Zhou and B. Krishnamachari, “Localized topology generation mechanisms for wireless sensor networks,” in *Proc. The IEEE Global Telecommunications Conference (GLOBECOM)*, December 2003.

[27] Y. Shi, Y. T. Hou, and A. Efrat, “Algorithm design for base station placement problems in sensor networks,” in *Proc. The 3rd International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QShine)*, August 2006.

[28] S. Tai, R. Benkoczi, H. Hassanein, and S. Akl, “A performance study of splittable and unsplittable traffic allocation in wireless sensor networks,” in *Proc. IEEE International Conference on Communications (ICC)*, June 2006.

[29] O. Younis and S. Fahmy, “Heed: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks,” *IEEE Transactions on Mobile Computing*, vol. 03, no. 4, pp. 366–379, 2004.

[30] L. Girod and D. Estrin, “Robust range estimation using acoustic and multimodal sensing,” in *Proc. IEEE/RSJ Conference on Intelligent Robots and Systems*, October 2001.

[31] N. Priyantha, A. Chakraborty, and H. Balakrishnan, “The cricket location-support system,” in *Proc. The 6th Annual International Conference on Mobile Computing and Networking*, August 2000.

[32] A. Savvides, C. Han, and M. Strivastava, “Dynamic fine-grained localization in ad-hoc networks of sensors,” in *Proc. The 7th Annual International Conference on Mobile Computing and Networking*, July 2001.

[33] P. Bose and P. Morin, “Online routing in triangulations,” in *Proc. The 10th International Symposium on Algorithms and Computation*, December 1999.

[34] E. Kranakis, H. Singh, and J. Urrutia, “Compass routing on geometric networks,” in *Proc. The 11th Canadian Conference on Computational Geometry*, August 1999.

[35] F. Zhao and L. Guibas, *Wireless Sensor Networks: An Information Processing Approach.* Morgan Kaufmann, 2004.

[36] J. O'Rourke, *Computational geometry in C (2nd Edition).* Cambridge University Press, 1998.

[37] G. Toussaint, "The relative neighborhood graph of a finite planar set," *Pattern Recognition*, vol. 12, no. 4, pp. 261–268, 1980.

[38] ——, "Pattern recognition and geometric complexity," in *Proc. The 5th International Conference on Pattern Recognition*, December 1980.

[39] B. Karp and H.Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *Proc. The Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, August 2000.

[40] X. Wu, G. Chen, and S. Das, "On the energy hole problem of nonuniform node distribution in wireless sensor networks," in *Proc. The IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS)*, October 2006.

[41] K. Padmanabh and R. Roy, "Bottleneck around base station in wireless sensor network and its solution," in *Proc. The 3rd Annual International Conference on Mobile and Ubiquitous Systems*, July 2006.

[42] A. Bogdanov, E. Maneva, and S. Riesenfeld, "Power-aware base station positioning for sensor networks," in *Proc. The 23rd IEEE International Conference on Computer Communications (INFOCOM)*, March 2004.

[43] B. Clark, C. Colbourn, and D. Johnson, "Unit disk graphs," *Discrete Math*, vol. 86, no. 1-3, pp. 165–177, 1990.

[44] J. Pan, Y. Hou, L. Cai, Y. Shi, and S. Shen, "Locating base-stations for video sensor networks," in *Proc. The 58th IEEE Vehicular Technology Conference (VTC)*, October 2003.

[45] E. Oyman and C. Ersoy, “Multiple sink network design problem in large scale wireless sensor networks,” in *Proc. IEEE International Conference on Communications (ICC)*, June 2004.

[46] W. Youssef and M. Younis, “Intelligent gateways placement for reduced data latency in wireless sensor networks,” in *Proc. IEEE International Conference on Communications (ICC)*, June 2007.

[47] ——, “Intelligent estimation of gateways count for reduced data latency in wireless sensor networks,” in *Proc. The IEEE Global Telecommunications Conference (GLOBECOM)*, November 2007.

[48] K. Akkaya and M. Younis, “Coverage and latency aware actor placement mechanisms in wsans,” *International Journal of Sensor Networks*, vol. 3, no. 3, pp. 152–164, May 2008.

[49] S. Soro and W. Heinzelman, “Prolonging the lifetime of wireless sensor networks via unequal clustering,” in *Proc. The IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, April 2005.

[50] Y. Hou, Y. Shi, H. Sherali, and S. Midkiff, “On energy provisioning and relay node placement for wireless sensor networks,” *IEEE Transactions on Wireless Communications*, vol. 4, no. 5, pp. 2579–2590, 2005.

[51] Q. Wang, K. Xu, H. Hassanein, and G. Takahara, “Minimum cost guaranteed lifetime design for heterogeneous wireless sensor networks,” in *Proc. The IEEE International Performance, Computing, and Communications Conference (IPCCC)*, April 2005.

[52] Q. Wang, K. Xu, G. Takahara, and H. Hassanein, “Locally optimal relay node placement in heterogeneous wireless sensor networks,” in *Proc. The IEEE Global Telecommunications Conference (GLOBECOM)*, November 2005.

[53] A. Azad and A. Chockalingam, "Mobile base stations placement and energy aware routing in wireless sensor networks," in *Proc. The IEEE Wireless Communications and Networking Conference (WCNC)*, April 2006.

[54] M. Gatzianas and L. Georgiadis, "A distributed algorithm for maximum lifetime routing in sensor networks with mobile sink," *IEEE Transactions on Wireless Communications*, vol. 7, no. 3, pp. 984–994, 2008.

[55] Y. Shi and Y. Hou, "Theoretical results on base station movement problem for sensor network," in *Proc. The 27th IEEE International Conference on Computer Communications (INFOCOM)*, April 2008.

[56] M. Younis and Q. Pan, "On handling weakened topologies of wireless sensor networks," in *Proc. IEEE Conference on Local Computer Networks (LCN)*, October 2008.

[57] I. Vasilescu, K. Kotay, D. Rus, M. Dunbabin, and P. Corke, "Data collection, storage, and retrieval with an underwater sensor network," in *Proc. International Conference on Embedded Networked Sensor Systems (SenSys)*, November 2005.

[58] E. Magistretti, J. Kong, U. Lee, M. Gerla, P. Bellavista, and A. Corradi, "A mobile delay-tolerant approach to long-term energy-efficient underwater sensor networking," in *Proc. The IEEE Wireless Communications and Networking Conference (WCNC)*, March 2007.

[59] C. Papadimitriou and K. Steiglitz, *Combinatorial Optimization : Algorithms and Complexity.* Dover Publications, 1998.

[60] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less low cost outdoor localization for very small devices," *IEEE Communications Magazine*, vol. 7, no. 5, pp. 28–34, 2000.

[61] J. Jaromczyk and G. Toussaint, “Relative neighborhood graphs and their relatives,” *Proc. IEEE*, vol. 80, no. 9, pp. 1502–1517, 1992.

[62] http://lpsolve.sourceforge.net/5.5/. February 25, 2009.

[63] I. Akyildiz, D. Pompili, and T. Melodia, “State-of-the-art in protocol research for underwater acoustic sensor networks,” in *Proc. The 1st ACM International Workshop on Underwater Networks*, September 2006.

[64] http://web.uvic.ca/ neptune/. February 25, 2009.

[65] I. Akyildiz, D. Pompili, and T. Melodia, “Underwater acoustic sensor networks: Research challenges,” *Ad Hoc Networks*, vol. 03, no. 3, pp. 257–279, 2005.

[66] V. Rodoplu and M. Park, “An energy-efficient MAC protocol for underwater wireless acoustic networks,” in *Proc. MTS/IEEE OCEANS*, September 2005.

[67] E. Sozer, M. Stojanovic, and J. Proakis, “Underwater acoustic networks,” *IEEE Journal of Oceanic Engineering*, vol. 25, no. 1, pp. 72–83, 2000.

[68] I. Albert F. Harris, M. Stojanovic, and M. Zorzi, “When underwater acoustic nodes should sleep with one eye open: idle-time power management in underwater sensor networks,” in *Proc. The 1st ACM International Workshop on Underwater Networks (WUWNet)*, September 2006.

[69] http://acomms.whoi.edu. February 25, 2009.

[70] A. Chakrabarti, A. Sabharwal, and B. Aazhang, “Using predictable observer mobility for power efficient design of sensor networks,” in *Proc. The 2nd IEEE International Workshop on Information Processing in Sensor Networks (IPSN)*, 2003, pp. 129–145.

[71] R. C. Shah, S. Roy, S. Jain, and W. Brunette, “Data mules: Modeling a three-tier architecture for sparse sensor networks,” in *Proc. The 1st IEEE International Workshop on Sensor Network Protocols and Applications (SNPA)*, 2003, pp. 30–41.

[72] J. Luo and J.-P. Hubaux, "Joint mobility and routing for lifetime elongation in wireless sensor networks," in *Proc. The 24th IEEE International Conference on Computer Communications (INFOCOM)*, 2005.

[73] G. Shi, M. Liao, M. Ma, and Y. Shu, "Exploiting sink movement for energy-efficient load-balancing in wireless sensor networks," in *Proc. The 1st ACM International Workshop on Foundations of Wireless Ad Hoc and Sensor Networking and Computing*, May 2008.

[74] A. Somasundara, A. Kansal, D. Jea, D. Estrin, and M. Srivastava, "Controllably mobile infrastructure for low energy embedded networks," *IEEE transactions on Mobile Computing*, vol. 5, no. 8, pp. 958–973, August 2006.

[75] X. Liu, Q. Huang, and Y. Zhang, "Combs, needles, haystacks: balancing push and pull for discovery in large-scale sensor networks," in *Proc. The 2nd International Conference on Embedded Networked Sensor Systems*, 2004.

[76] K. Akkaya and M. Younis, "Energy-aware routing to a mobile gateway in wireless sensor networks," in *Proc. The IEEE Global Telecommunications Conference Workshops*, November 2004.