MODELS AND SOLUTION APPROACHES FOR EFFICIENT DESIGN AND OPERATION OF WIRELESS SENSOR NETWORKS

A Dissertation

by

HUI LIN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

December 2010

Major Subject: Industrial Engineering

MODELS AND SOLUTION APPROACHES FOR EFFICIENT DESIGN AND OPERATION OF WIRELESS SENSOR NETWORKS

A Dissertation

by

HUI LIN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Chair of Committee,	Halit Üster
Committee Members,	Abhijit Deshmukh
	Eylem Tekin
	Ibrahim Karaman
Head of Department,	Brett Peters

December 2010

Major Subject: Industrial Engineering

ABSTRACT

Models and Solution Approaches for Efficient Design and Operation
of Wireless Sensor Networks . (December 2010)
Hui Lin, B.S., Tianjin University, Tianjin,China;
M.S., Tianjin University, Tianjin,China
Chair of Advisory Committee: Dr. Halit Üster

Recent advancements in sensory devices are presenting various opportunities for widespread applications of wireless sensor networks (WSNs). The most distinguishing characteristic of a WSN is the fact that its sensors have finite and non-renewable energy resources. Many research efforts aim at developing energy efficient network topology and routing schemes for prolonging the network lifetime. However, we notice that, in the majority of the literature, topology control and routing problems are handled separately, thus overlooking the interrelationships among them.

In this dissertation, we consider an integrated topology control and routing problem in WSNs which are unique type of data gathering networks characterized by limited energy resources at the sensor nodes distributed over the network. We suggest an underlying hierarchical topology and routing structure that aims to achieve the most prolonged network lifetime via efficient use of limited energy resources and addressing operational specificities of WSNs such as communication-computation trade-off, data aggregation, and multi-hop data transfer for better energy efficiency. We develop and examine three different objectives and their associated mathematical models that define alternative policies to be employed in each period of a deployment cycle for the purpose of maximizing the number of periods so that the network lifetime is prolonged. On the methodology side, we develop effective solution approaches that are

based on decomposition techniques, heuristics and parallel heuristic algorithms. Furthermore, we devise visualization tools to support our optimization efforts and demonstrate that visualization can be very helpful in solving larger and realistic problems with dynamic nature. This dissertation research provides novel analytical models and solution methodologies for important practical problems in WSNs. The solution algorithms developed herein will also contribute to the generalized mixed-discrete optimization problem, especially for the problems with similar characteristics.

To *my family*

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, Dr. Halit Üster during my study at Texas A&M University. Dr.Üster has been a wonderful advisor and mentor, providing me with invaluable support and advice both professionally and personally. His knowledge and enthusiasm for research always inspires me. I have learned a great deal from working with him during these five years. I cannot thank him enough in every aspect of my Ph.D. studies.

I would also like to thank my advisory committee members, Drs. Abhijit Deshmukh, Eylem Tekin and Ibrahim Karaman for providing insightful suggestions and comments throughout the development of this dissertation. I thank Dr. Sıla Çetinkaya for her guidance and encouragement during the years at Texas A&M University.

I thank National Science Foundation for providing financial support during my graduate studies and I also thank Drs. Amit Majumdar, DJ Choi and Yifeng Cui and Mr. Amit Chourasia for their guidance in parallel computing and data visualization. I would like to thank Judy Meeks, Claudia Samford for administrative help and Dennis Allen and Mark Hopcus for technical assistance.

I am thankful to my dear officemates: Homarjun Agrahari, Gopalakrishnan Easwaran, Burcu Keskin, Liqing Zhang, Panitan Kewcharoenwong, Joaquin Torres, Ezgi Eren, Xinghua Wang, Su Zhao, and Abhilasha Katariya. Thank you all for making our lab a fun and wonderful place to work. I also own many thanks to Haifeng Xia, Anne Varisco, Yanbin Xu, Ani Ma, Ying Xie, Qindan Huang, Ting Pang, Junping Xu, Leina Zhu, Peng Lin, Jiang Zhang, Huiliang Zhang and Kai Yin for their friendship and support. I also wish to thank Dr. Andrew Chan and Mrs. Sophia Chan for their mentorship in my personal life.

I am indebted to my parents and parents-in-law for always supporting me and

being there for me. Without their love and enormous support, this dissertation would not have been possible. I thank my brother, my sister-in-law, and my cute niece, for brightening my life. I am blessed to have you all in my life.

I would like to thank my wonderful husband, Jian Xu and my lovely daughter Lydia Xu for always loving me, supporting me and keeping me happy and sane. I know there is nothing I cannot accomplish with them by my side.

I thank my Lord, Jesus Christ for His unfailing love and support though the good and bad times. Lord, you are always my inspiration and motivation towards the completion of the dissertation research.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Recent advances in wireless networking, embedded microprocessors, integration of micro-electro mechanical system (MEMS) and Nanotechnology have enabled the rapid development of low-cost, low-power and multi-functional sensors (Chong and Kumar, 2003; Sohraby et al., 2007). Very small in size, sensors are capable of sensing, data processing and communicating with each other or with user nodes (sinks). Sensors can be used to sense a wide range of natural or artificial phenomena including temperature, pressure, humidity, light, motion, weight, noise, etc. A group of sensors communicating in a wireless medium for the purpose of gathering data and transmitting it to a user (sinks) form a *wireless sensor network* (WSN). Sensor Network technology is a key technology in the future. Indeed, some claims that the advent of tiny, cheap and smart sensors will radically change our world in the way microprocessors did in the 1980s and the Internet did in the 1990s (Saffo, 1997).

WSNs represent a paradigm shift in computing (Saffo, 1997; Estrin et al., 2001; Krishnamachari, 2005). In a traditional computing infrastructure, we interact directly, one-on-one, with computers. In the near future, hundreds or thousands of sensors can be embedded deeply around us, placed inside our cars, homes, offices, hospitals, shopping centers and factories. When we are in control of massive-scale sensors, it may be impossible to interact directly with each one. Instead, these sensors can anticipate our needs and interact with the physical world on our behalf. WSNs provide an interface between the virtual information world and the physical world. According to America's National Research Council report entitled *embedded, Every-*

This dissertation follows the style and format of *Operations Research*.

where, the phenomenon of sensor technology could "well dwarf previous milestones in the information revolution" (Estrin et al., 2001).

WSNs represent a paradigm shift in information extracting (Heinzelman, 2000; Akyildiz et al., 2002b). In a traditional wired sensor network, there are a very limited number of sensor nodes extracting information from the environment. Those sensors are large and expensive and also require large amount of energy for operation. The positions of the sensors and topology infrastructure must be carefully engineered. WSNs, on the other hand, consist of a large amount of sensors that are densely deployed. The position of sensor nodes need not be engineered or pre-determined. This allows random deployment in inhospitable environments and difficult-to-reach terrains. Hence, there would be significant economical and environment gains if we employ WSNs instead of wired sensor network. In fact, there are a wide array of applications in WSNs for data gathering purposes. For example, WSNs have profound effect on military and civil applications such as forests to combat fires, urban or rural battlefields, borderlines; in wild habitats and oceans to monitor and observe natural phenomena; in disaster prevention and relief; in urban environments to monitor and control traffic; and in industrial settings to track inventories and the state of other resources (Wang et al., 2005). Recently, data gathering WSNs find increasing widespread applications in ecological and environmental monitoring (Collins et al., 2006; Hart and Martinez, 2006; Rundel et al., 2009).

WSNs promise to revolutionize the way we conduct business and live our lives. Indeed, Technology Review, MIT's magazine of innovation, lists WSNs as one of the "10 emerging technologies that will change the world " (MIT's Technology Review, 2003). Business 2.0 Magazine identifies environmental sensor networks as one of the " 8 Technologies for a Green Future " (Datta and Woody, 2007). WSNs can have tremendous influence on energy use monitoring and utility management. A

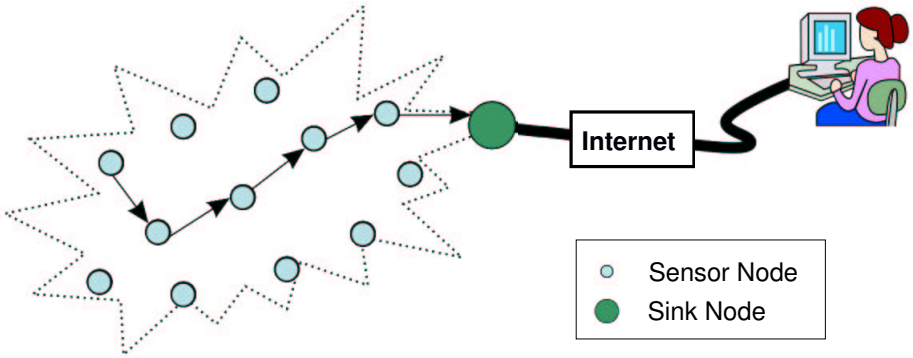
West Technology Research Solution (WTRS 2010) report titled *The WTRS Wireless Sensor Network Technology Trends Q1 2010* notes that nearly 70% of the average household utility bill could be affected by WSN temperature and lighting control applications. In years to come, wireless sensor technology developments will impact every facet of our life, bringing many new opportunities.

I.1. Characteristics of Wireless Sensor Networks

A wireless sensor is a small device with on-board sensing, data processing and storage, transceiver (to transmit and receive data) and power units. It can also include a location finding system since some applications require accurate knowledge of location and a mobilizer that moves the sensor to carry out its tasks. Figure I.1 depicts the typical communication architecture of sensor network. A sensor network is designed to monitor phenomena or detect events, collect and process data, and transmit to a user (or the sink nodes). Sensor nodes are usually deployed in various environments, including remote and hostile regions. The most distinguishing characteristic of a WSN is the fact that its sensors have finite and non-renewable energy resources. Thus, innovative techniques to promote energy efficiency so as to prolong the network lifetime are highly preferred. The communication between the sensors usually exhibits short transmission range. According to Pottie and Kaiser (2000), a multihop data transfer scheme is preferable for reducing energy costs. Sensor nodes should possess the self-organizing abilities as the operation of WSNs is unattended and the communication among sensors is in ad-hoc fashion.

There are important challenging issues that distinguish WSNs from other wireless ad hoc networks (WAHNs) (Akyildiz et al., 2002b; Tilak et al., 2002; Al-Karaki and Kamal, 2004; Akkaya and Younis, 2005):

Figure 1: Typical Wireless Sensor Network Architecture. Source: Al-Karaki and Kamal (2004).



1. The number of sensor nodes in WSNs can be several orders of magnitude higher than the nodes in WAHNs. Sensor nodes are typically densely deployed in WSNs.

2. Sensor nodes are prone to failure and the topology of WSNs changes frequently.

3. In WAHNs, the network nodes have limited but usually rechargeable energy whereas in WSNs, which are mostly unattended, a battery powered sensor node becomes non-operational once its energy is depleted. The limited energy resources take a more critical role while planning for efficiency in WSNs. Hence, one of the main objectives in WSNs is to promote energy efficiency so as to prolong the network lifetime.

4. The communication pattern in WAHNs, is generally a unicast (one-to-one) or its generalizations including multicast (one-to-many) and broadcast (one-to-all), in which the flow originates in one node and disseminates to one or more nodes in the network. However, in WSNs, a communication pattern is such that there is at least one sink node to which the data flow is directed, i.e., the network flow

resembles a convergecast. Therefore, the routing problem in WSNs actually adopts a data gathering purpose rather than providing a pure communication among the network nodes.

5. The data collected in WSNs is usually based on common phenomena and sensors, that are in close proximity to each other, may produce the same or similar data, which, in turn, renders the data redundancy issue.

The last two facts highlight another distinctive characteristic of WSNs and that is the possibility of data aggregation. In particular, WSNs are deployed to obtain information from a region of interest, and, there is usually significant opportunity for in-network processing.

From a planning and operations perspective, two problems that are fundamental to effective and efficient design and operation of sensor networks include *Topology Control*, as a tactical problem, and *Routing*, as an operational problem. Topology control refers to the determination of an underlying network topology that specifies the existing linkages (arcs) available for data flow and routing refers to the determination of paths for transfer of data over the network with this known topology. The relationship between these problems is emphasized by WSN-specific attributes – *energy efficiency* and *computation-communication trade-off*. Energy-efficiency is important because each sensor is equipped with an on-board nonrenewable power unit. The communication-computation trade-off refers to the fact that communication consumes more energy than performing computations on-board in a sensor (Wang and Hassanein, 2005).

I.2. Motivation and Scope of the Dissertation

Because of potentially harsh and dynamic environments, along with energy constraints, sensor network design will likely encounter many challenges. This dissertation focuses on developing power-aware mathematical models and solution approaches for the integrated topology control and data routing problems to prolong the network lifetime. In the light of WSN characteristics presented above, we summarize the motivations for this dissertation research as follows:

1. As sensor nodes have severe constraints in energy supply and are typically operated unattended, network lifetime is perhaps the most important performance metric in WSN design. Innovative techniques to promote energy efficiency so as to prolong the network lifetime are highly required in WSNs.

2. Due to the frequently changing topology and limited energy provision for each sensor node, the interrelationships between topology control and routing problems are more pronounced. In particular, if the issues of designing efficient routing schemes are not taken into consideration in the topology control problem, the underlying network topology might not be suited for supporting a good routing scheme. However, in the majority of the literature most of these problems are handled separately, thus overlooking the interrelationships among them.

3. *Energy-efficiency* is a very important design/operation attribute because of the special operating characteristics of sensors. Each sensor is equipped with an on-board nonrenewable power unit. The failure of a sensor due to power depletion affects overall network performance. This is because, in several contexts, including the one considered in this study, a sensor not only functions to cap-

ture information in its vicinity, but also functions as a relay node to transfer the data generated by other sensors to the sink nodes.

4. *Communication-Computation trade-off* evident in sensor networks is critical as it relates to the energy efficiency and fault tolerance attributes. Communication consumes more energy than performing computations in a sensor node that uses on-board processing capability (Wang and Hassanein, 2005). Although the direct communication of a sensor node with a sink node is preferable since it provides a higher fault tolerance for the overall network, this contributes to the shortening the beneficial network life with excessive energy use. Therefore, routing schemes where the data size is decreased via data aggregation (using energy for computation) along the path to a sink node are usually preferred.

Observing that all these attributes and metrics as discussed above are tightly coupled, in this dissertation, we are motivated to investigate the following research questions:

- How to promote the energy efficiency so as to extend network lifetime?
- How to design a network that can be constructed and updated efficiently while ensuring attractive routing schemes?

In this dissertation research, the main goal is aimed at addressing the above questions via integrating topology control and routing problems under simultaneous consideration of communication-computation trade-off, data aggregation, and multi-hop data transfer for better energy efficiency.

In the WSN applications, the main purpose is to monitor and collect data and then transmit this data to the sink nodes. In general, data sensing and reporting is dependent on the application. Tilak et al. (2002) categorize the data delivery model

in WSN as time-driven, event-driven, query-driven and hybrid. In the time-driven model, the sensor nodes sense their data continuously at a prespecified rate and send it to the sink periodically. For event-driven and query-driven models, sensor nodes react when a certain event occurs or a query is generated by the sink. They are well suited to time-critical applications. A combination of the above methods is also possible. In our study, we concentrate on time-driven sensor networks whose applications are found in the continuous data gathering contexts.

In general, the lifetime of a sensor network can be defined as the time frame between two successive sensor deployments, i.e., a deployment cycle. A deployment cycle consists of successive *periods* of fixed time length for which topology and/or routing decisions are made. Thus, prolonging the network lifetime corresponds to obtaining the maximum number of successive periods that the data generated at the sensors can reach the user. In our case, the end of a deployment cycle is reached when it is not possible to obtain a feasible solution to the problem of transmitting data generated at the sensors to the user. Based on this definition, the goal of prolonging network lifetime can be achieved via reducing the energy consumption while ensuring the energy usage across the network uniformly. This is in contrast to simply minimizing the energy dissipation, which may leave the network with a wide disparity in the energy levels of nodes.

I.3. Contributions of this Dissertation

In this dissertation, we contribute to the current literature by investigating the integrated mathematical models for topology control and routing solutions based on optimization techniques for the design of WSNs. We adopt a hierarchical data flow structure in which data generated at the sensors are first routed to the sensors des-

ignated as clusterheads (CHs). Each sensor is assigned to at least one CH which reduces the total data size that it receives from sensors via aggregation. Each CH routes data to a sink either through other CHs, which act only as relays without aggregation, or directly. Such a structure is beneficial in terms of energy efficiency in three ways: 1) Since the sensors in close proximity of each other are likely to be in the same cluster and may generate very similar data, data aggregation at CHs helps to reduce redundancy and energy consumption in communication; 2) Hierarchical structure distributes the energy usage to multiple sensors on multi-hop paths, thus eliminating the quick expiration of the sensors away from the sinks; 3) Since energy dissipation in communication is proportional to the square of the distance, compared to direct communication, the total energy dissipation due to communication is less on a multi-hop route (Santi, 2005a).

We develop and examine three different objectives and their associated mathematical models that define alternative policies to be employed in each period of a deployment cycle for the purpose of maximizing the number of periods so that the network lifetime is prolonged. The objectives include minimization of 1) *total or average energy usage in the system* 2) *maximum energy used at a sensor node*, and 3) *a weighted sum of the range of end-of-period remaining energy distribution at the sensor nodes and the average energy used in the system*. Furthermore, we consider two important extension models to the setting of the third objective, by incorporating the fixed CH set-up cost (as in the first case), and the single-sourcing requirements for CH assignments and the transmission ranges of sensor nodes (as in the second case).

In devising our models, we consider the use of multiple sinks. This is helpful for energy efficiency since multiple sinks create an opportunity for better proximity to sensors, thus saving energy in communication. It is possible to route the data so that the energy drainage in the network is more evenly distributed to the sensors by

changing the locations of the sinks and the CHs in each period.

Since the models dictate large discrete optimization formulations, exact solutions are highly impractical using exact optimization methods such as branch-and-cut. On the methodology side, we develop effective solution approaches that are based on decomposition techniques, heuristics and parallel heuristic algorithms. Furthermore, we devise visualization tools to support our optimization efforts and demonstrate that visualization can be very helpful in solving larger and realistic problems with dynamic nature. This dissertation research is expected to provide novel analytical models and solution methodologies for important practical problems in WSNs. We aim to apply our models on time-driven sensor networks applications pertaining to monitoring ecological habitats (animals, plants, micro-organisms). In particular, some specific examples of potential applications are presented as follows:

- *Habitat monitoring on Great Duck Island* : Since habitat monitoring is very sensitive to human presence, unattended wireless sensor network provides a noninvasive approach to obtain the real-time environmental data. Researchers (Cerpa et al., 2001; Mainwaring et al., 2002) from the University of California at Berkeley and the College of the Atlantic, deploy sensor networks on Great Duck Island to monitor the nesting burrows of Leach's Storm Petrels. Each sensor collects the data and transmit it to the sink node. The sink node then connects to the users via a satellite communication link.

- *Ecological monitoring on the Big Island of Hawaii* : The PODS project (Biagioni and Bridges, 2002) at the University of Hawaii, deploys sensor network in Volcanoes National Park on the Big Island of Hawaii. The environmental data, such as temperature, light, wind, humidity, and rainfall, collected by WSNs is used to monitor the ecological environment and events around the rare and

endangered species of plants.

I.4. Organization of the Dissertation

The remainder of this dissertation is organized as follows. In Chapter II, we provide an overview of the topology control and routing studies in wireless sensor network design problem. In Chapter III, we present three mathematical models for integrating topology control and routing decisions, and develop a heuristic solution algorithm for the models. Computational evidence demonstrates that, our proposed model **(M3)**, which minimizes average energy usage and the range of remaining energy distribution at the sensors, captures important characteristics of topology control and routing integration in WSN design. In Chapter IV, we consider an extension model of **(M3)** by incorporating the fixed cost associated with locating the CHs. On the methodology side, we develop a Benders decomposition solution approach that incorporates a simple heuristic algorithm, the strengthened Benders cuts and an ε-optimal approach. In Chapter V, we consider another extension model of **(M3)** by incorporating the single-sourcing requirements for CH assignments and explicitly specifying the transmission ranges of sensor nodes. We develop the associated mathematical model and a parallel heuristic algorithm. In Chapter VI, we develop a data visualization toolkit to support our optimization efforts. Finally, conclusions and future research directions are summarized in Chapter VII.

CHAPTER II

LITERATURE REVIEW

In Wireless Sensor Networks (WSNs), the sensors have limited and unchargeable energy provision and need to operate unattended for a long period of time. Thus, innovative techniques that can manage energy resources wisely so as to maximize the network lifetime, are highly preferred. In the literature of WSN design to date, extensive effort has been invested in developing energy efficient protocols and routing paradigms to maintain the requested network topology for prolonging the network lifetime. In this section, we provide an overview of the topology control and routing literature, and also point out the relevant studies on WSNs.

II.1. Sensor Deployment

Sensor Deployment refers to the implantation of sensors in the region of interest, and it can be performed in a deterministic or a random fashion. Note that the network topology and routing decisions have to be made with a given deployment. The deployment of WSNs varies with the application considered. A deterministic sensor placement may be feasible in friendly and accessible environments. Biagioni and Sasaki (2002) suggest regular deployment strategies for reliability and specifically analyze circular and star deployment topologies and deployments in square, triangular and hexagonal grids. They conclude that sensor communication radius can significantly impact the network coverage. On the other hand, a random sensor deployment is generally considered in remote or inhospitable areas. Clouqueur et al. (2003) define a path exposure metric as a deployment measure and discuss deployment strategies to detect moving targets with random deployment. However, a random placement can not guarantee the full coverage and may not provide a uniform sensor distri-

bution. There are some studies that aim to use mobile sensor nodes to enhance network coverage in the self-deployment methods. Howard et al. (2002) present an incremental and greedy self-deployment algorithm for mobile sensor networks. Each node is placed based on the information gathered by the previous deployed nodes. Zou and Chakrabarty (2003) present a two-step deployment strategy where a random deployment is first performed and its coverage is enhanced by a redeployment using a virtual force algorithm and energy constraints. Heo and Varshney (2005) propose a distributed energy-efficient deployment algorithm in terms of coverage and uniformity for intelligent mobile sensor networks. They employ a combination of clustering structure and a peer-to-peer deployment scheme. Cheng et al. (2008) propose a general network lifetime model and evaluate different deployment strategies so to maximize the network lifetime. These strategies include transmission power control, mobile data sink, multiple sinks and different initial energy levels.

Note that, in this research, we assume that the sensors are randomly deployed in the sensor field and each sensor knows its position information. The study of sensor deployment problem is beyond the scope of this dissertation.

II.2. Topology Control Problem

Topology control refers to the determination of an underlying network structure so that a medium is created for the routing of data to take place. Topology control is one of the main problems in the area of WSNs design in which the energy efficiency is an important aspect of network operations. Topology control is not an activity that directly causes energy usage as in the case of routing data. Therefore, an energy related objective in topology control is essentially a proxy measure adopted for instilling energy efficiency at this stage of planning with the hope of achieving real

benefits in the routing stage.

II.2.1. Transmission Range based Topology Control

In the transmission range based studies, topology control is mainly achieved by the adjustment of the sensors' transmission ranges, which is related to the power level settings at the sensors. The transmit range settings at the nodes determine the spatial extent the nodes can send data and, thus, the connectivity of the network. There are several studies that consider the *topology control* problem from this perspective and aim to setup a strongly connected network by considering an energy usage related objective. Ramanathan and Hain (2000) consider a static network and address the problem of adjusting the transmit powers of nodes in a multi-hop setting to minimize the maximum energy used at a node while maintaining bi-connectivity and present minimum spanning tree (MST) based algorithms. Tseng et al. (2004) present a modified MST approach, which builds on an algorithm given in Ramanathan and Hain (2000), to form k-edge and k-vertex connected topologies for $k = 1, 2$. Wattenhofer et al. (2001) introduce a distributed protocol called Cone Based Topology Control, which aims a minimal energy requirement at the nodes while maintaining global connectivity. Lloyd et al. (2005) address the objectives of minimizing the maximum energy use at a node and minimizing the total energy use, and suggest a polynomial algorithm and an approximation algorithm, respectively. Kubisch et al. (2003) suggest two localized algorithms that consider connectivity using a threshold number of neighbors of nodes without specific objectives on energy usage.

In these above studies, the authors assume that all the sensor nodes are homogeneous and have the same transmission range. Liu and Li (2003), on the other hand, consider networks with heterogeneous sensors and present a distributed shortest path algorithm to calculate the per-node minimum transmission power so that

the reachability between any two nodes is guaranteed and nodal transmission power is minimized. Santi (2005b) provides a review of transmission range based topology control studies and further references can also be found in Santi (2005a). Note that, as opposed to a hierarchical structure and a convergecast pattern, most of the studies take the perspective of a flat topology and a unicast or broadcast among the sensors. A topology control approach for a convergecast is given by Rodoplu and Meng (1999). The authors describe a distributed algorithm which generates a minimum total energy, a tree rooted at the master-site to which all the sensor can reach.

Furthermore, we observe that the minimization of total energy used and the maximum energy used by a sensor are two common metrics for energy efficient topology in the literature. The first metric may not maximize the network lifetime, because it does not take the nodes' remaining energy into account. If some critical nodes happen to be on more favorable paths requiring less transmission energy, they might suffer from early failure due to the heavy load in forwarding data packets. The second metric aims to balance the energy consumption by minimizing the highest energy usage at a sensor node. However, it may create long multi-hop routes and consume more energy than the minimum-energy route. In both of these cases, we face the issue of quick energy drainage which occurs at certain nodes in the first case and in the whole network in the second case.

II.2.2. Joint Consideration of Coverage Preservation and Connectivity

Recently, there are also some researchers addressing the coverage problem for the desired topology, while maintaining the connectivity. That is, they only choose a partial set of sensors in the active mode in order to reduce the energy consumption and prolong network lifetime. Zhang and Hou (2005) present a distributed topology control algorithm for maintaining sensing coverage and connectivity by keeping a

minimum number of sensor nodes in the active mode in WSNs. They also show that "the communication range is at least twice the sensing range " is a sufficient condition to ensure a full coverage of a convex area. Tian and Georganas (2005) enhance their work by proving that "the communication range is twice of the sensing range " is the sufficient condition no matter the area is a convex area or not. On the other hand, there are some studies that jointly consider the coverage and connectivity problem based on a theoretical formation of the problem. Nakamura et al. (2005) develop a dynamic mixed integer Linear Programming (LP) model for the multi-period coverage and connectivity problem under a flat topology, and solve it using the commercial package CPLEX. Alfieri et al. (2007) present a mixed integer LP model to exploit data redundancy by defining subset of sensors active in different time periods, to allow sensors to save energy when inactive. Column generation approach and a heuristic algorithm are suggested. Cardei and Du (2005) propose a centralized approach for achieving full coverage by organizing the sensors into a maximal number of disjoint set covers. These disjoint sets are activated successively, so that only one set is response for monitoring the targets at a specific time. Soro and Heinzelman (2009), on the other hand, consider the coverage preservation problem based on cluster-based sensor networks. Several coverage-aware cost metrics are explored for selecting the set of Cluster-heads (CHs) and active sensor nodes that provide full network coverage, as well as the set of routers that forward data to the sink node.

II.3. Routing Problem

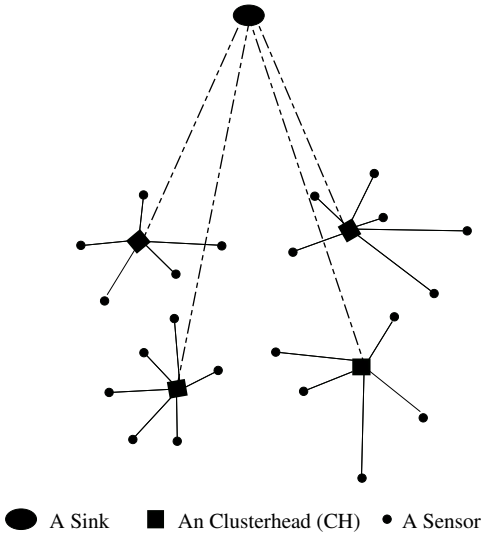
Routing refers to the determination of paths for transfer of data over the network with the given network topology. As discussed in chapter I.2, the main purpose in WSNs is to monitor and collect data and then transmit this data to the data sink(s).

An energy-efficient routing paradigm to find proper paths from the sensors to the sink(s) with the purpose of maximizing the network lifetime is highly required in the WSNs design problem.

II.3.1. Routing Protocol

The majority of the *routing studies* focus on developing communications protocols while others adopt spanning tree, shortest path or multi-commodity flow based approaches. WSNs usually have a multihop physical topology and this topology can result in more efficient routing. Clustering protocols have been investigated extensively for designing energy-efficient and scalable sensor networks. The basic idea is to organize the network into a set of clusters ; within each cluster, sensors transmit their information to their associated cluster-heads(CHs). The CHs in turn, aggregates the received data packets and forwards it to the sink node. Instead of using a fixed network topology, the configuration of network topology is dynamic and it varies over different periods.

Figure 2: LEACH Network Model



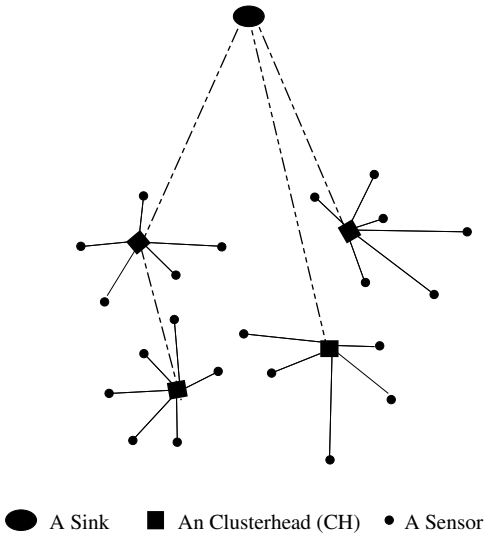
Heinzelman et al. (2000) develop a data aggregating cluster based routing protocol LEACH (Low Energy Adaptive Clustering Hierarchy) to a single sink at a fixed known location. Figure 2 depicts a typical data flow pattern in the network model used by LEACH protocol. Each CH acts as both an aggregator and a relay to the sink. LEACH aims to maximize the number of periods until either one or all the sensors die (lifetime) via a localized approach using randomized CH selections and a minimum energy based assignment of sensors to CHs in each period. A routing scheme based on 1-hop connections between a sensor and its CH, or to the sink, as well as between a CH and the sink, is assumed. In the centralized version, LEACH-C Heinzelman et al. (2002), the sink node determines the CH from a selected group of sensors with higher energy levels and aims at minimizing the amount of total energy spent to transmit data to the CHs.

LEACH is one of the first hierarchical routing approaches in WSNs and later, much work focuses on extending their work with the same general setting. Bandyopadhyay and Coyle (2003) modify LEACH via consideration of a multi-level clustering and a fixed transmission range at the sensors. However, only one period problem is considered and a better energy dissipation rate is obtained; no comparisons on lifetime are given. Khan et al. (2003) provide a low energy localized clustering based routing protocol for the setting in Bandyopadhyay and Coyle (2003), but with 1-hop connections as in LEACH for improved cluster compactness. A comparison with LEACH is provided, based on per period energy usage and cluster quality, with no results on a lifetime measure.

Although LEACH performs better than direct-from-sensor-to-sink routing, the minimum transmission energy (MTE) based routing (Singh et al., 1998), and a static clustering based (instead of CH rotations) routing approach, certain issues about its underlying assumptions remain. LEACH assumes that all nodes can reach the sink

node with enough power, which is not always true for WSNs, due to the limited energy provision for each sensor. LEACH adopts the randomized rotation of cluster heads (CHs) to ensure a balanced energy consumption. It is possible that the elected CHs will be concentrated in one part of the network, some nodes will not have any CHs in their vicinity.

Figure 3: HEED Network Model



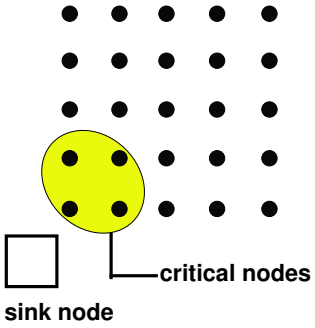
To account for the shortcoming of LEACH protocol, Younis and Fahmy (2004) propose a distributed clustering routing protocol HEED (Hybrid Energy-Efficient Distributed clustering). In HEED, the primary goal is to identify a set of CHs and organize sensor networks into clusters so as to utilize the limited energy resources wisely. The CHs are probabilistically selected based on their remaining energy and the sensor nodes join clusters such that the communication cost is minimized. If a sensor node falls within the range of multiple candidate CHs, it will select the CH with the least communication cost. Figure 3 depicts a typical data flow pattern in the network model used by HEED protocol. A group of sensors define a cluster, and they transmit the collected data to the associated cluster-head (CH) which in turn transmits the

data to the data sink either directly or through other CHs. In addition to acting as a relay node, a CH processes the collected data to eliminate duplications. In comparison to LEACH, HEED assumes a multi-hop connection between CHs or to the sink node and at each period, they select CHs with high remaining energy. Simulation results demonstrate the effectiveness of HEED protocol in terms of network lifetime. HEED provides an effective method for prolonging the network lifetime. However, it does not necessarily ensure a balanced energy consumption because of the following reasons: (1) In HEED, the CHs closer to the sink node may have quick drainage due to their heavy load in forwarding data packets; (2) At each period, HEED attempts to choose the highest energy sensors as CHs. However, it may be biased from the long-term network lifetime perspective.

Liu et al. (2009) provide a distributed energy efficient protocol EAP for the general setting in Younis and Fahmy (2004). In EAP, each cluster-head is probabilistically selected based on its high ratio of the remaining energy to the average remaining energy of all the neighbor nodes within its cluster range. This is simply in contrast to HEED that only chooses CHs based on a node's own remaining energy. To further extend the network lifetime, EAP introduces the idea of "intra-cluster coverage", which allows a partial set of sensors in the active mode within clusters while maintaining coverage expectation of the cluster. A comparison with LEACH and HEED is provided, based on the network lifetime.

As WSNs generally send the collected information to the sinks in a "many to one" (convergecast) fashion, Haenggi (2003) points out that some critical nodes closer to the sink have the heavier load for forwarding data packet to the sink. Figure 4 depicts a critical area in the sensor network whose nodes are present on most forwarding paths in the network. Specifically, in a multi-hop cluster-based sensor network, the CHs closer to the sink node may have quick drainage due to their heavy load in forwarding

Figure 4: Sensor Network with a Data Sink. Source: Haenggi (2003).



data packets. To ensure a balanced energy consumption, Chen et al. (2009) propose an Unequal Cluster-based Routing (UCR) protocol. In UCR, CHs closer to the sink have smaller cluster sizes than those farther from the sink so that "popular" CHs are protected from quick energy depletion. Simulation results show that UCR significantly improves the network lifetime in comparison to HEED.

II.3.2. Optimization Approach

As opposed to the routing protocols which are based on localized or heuristic algorithms, some research addresses the routing algorithm based on a theoretical formation of the problem. The typical approaches are to employ shortest path and multicommodity flow models with modified link costs that incorporate energy requirements and levels at the relay nodes. Chang and Tassiulas (2000, 2004) consider the problem of maximizing the network lifetime with flat topologies. They present a Linear Programming (LP) model where the objective is lifetime (single timeframe) maximization (equivalently maximizing data flow). In fact, they are the first to treat this problem as a LP problem. Bhardwaj and Chandrakasan (2002) develop upper bounds on the lifetime of networks based on optimum role assignments to sensors (e.g., whether they should act as routers or aggregators). Ordónez and Krishna-

machari (2004) present models both for maximizing the total information gathered subject to energy constraints, and for minimizing the energy usage subject to information constraints. Hua and Yum (2008) aim to maximize the network lifetime by jointly optimizing data aggregation and routing. The main drawback in these studies is that they assume the static network topology, which may not be optimal for balancing the energy consumption over the periods. Kalpakis et al. (2003) study the maximum lifetime data aggreation (MLDA) problem using the setting in Heinzelman et al. (2000) and formulate the lifetime as a maximization linear program. The solution gives the overall arc flows and it is later decomposed to determine spanning (routing) trees, one for each period, rooted at the sink.

Also, it is well acknowledged that mobile sink or multiple sinks can be used to increase network manageability and to reduce the energy dissipation at each node. It is possible to route the data so that the energy drainage in the network is more evenly distributed to the sensors by varying the sink location or employing multiple sinks. Moreover, multiple sinks create an opportunity for better proximity to sensors, thus saving energy in communication. Gandham et al. (2003) suggest an integer program to determine sink locations under a flat-routing without CHs and aggregation. Minimization of total energy used and the maximum energy used by a sensor, are considered and the value of employing multiple, mobile sinks is illustrated. Xue et al. (2005) extend the framework of a multicommodity flow problem and suggest the use of multiple data sinks to increase the network lifetime. However, they do not consider data aggregation to eliminate redundancy. Papadimitriou and Georgiadis (2006) present a LP model to exploit the capability of the sink to be located in different places during network operation in order to maximize network lifetime. This goal is achieved by solving two joint problems: a scheduling problem that determines the sojourn times of the sink at different locations, and a routing problem that aims

to develop an energy-efficient data transfer scheme from the sensor nodes to the sink.

Alfieri et al. (2007) propose to exploit data redundancy by defining subset of sensors active in different time periods, to allow sensors to save energy when inactive. They also present the mathematical programming model which includes two subproblems: routing and scheduling problems. Al-Karaki et al. (2009) propose an energy efficient routing scheme GRASS (Grid-based Routing and Aggregator Selection Scheme). GRASS aims to maximize the network lifetime by jointly addressing the cluster-based routing problem with application specific data aggregation. Mathematical formulation and heuristic solution approaches are suggested. Numerical studies demonstrate the effectiveness of GRASS in terms of network lifetime performance.

Recently, cross-layer design and optimization in WSNs have received significant attention. In the cross-layer approach, different layers of the protocol stack can integrate and share information among each other to enhance network performance and maximize the lifetime. Xing et al. (2005) propose the Minimum Power Configuration (MPC) approach which integrates topology control, energy-efficient routing, and sleep management as a joint optimization problem. Burri et al. (2007) propose a data-gathering protocol *Dozer*, that jointly considers medium access control (MAC) layer, topology control and routing to save energy. *Dozer* employs a tree-based network structure to route the data, coordinates the nodes sleep schedules and achieves low radio duty cycles. Madan et al. (2007) consider the optimization of transmission schemes to maximize the network lifetime among the link, MAC and routing layer. A simple network topology is given to compute energy consumption and network lifetime. We note that these studies adopt a predetermined physical topology structure such as tree-based and linear topology, for data routing. Research on the relation between topology control and routing is very limited. Specifically, if the issues of routing are not taken into consideration in the topology control problem, then the

underlying topology might not be suited for supporting an efficient routing paradigm.

Reviews on routing are given in Akkaya and Younis (2005); Al-Karaki and Kamal (2004); Abbasi and Younis (2007) and excellent reviews of WSNs include Akyildiz et al. (2002a,b); Karl and Willig (2005); Yick et al. (2008).

II.4. Positioning in the Current Literature

In the light of the discussion above, we summarize the position of this dissertation research in the current literature as follows:

1. In the literature of WSN design, extensive effort has been invested in reducing energy consumption and balancing the energy usage of the network so as to prolong sensor network lifetime. However, in the majority of the literature, topology control and routing problems are handled separately, thus overlooking the interrelationships among them. For this purpose, we have studied an integrated topology control and routing problem in WSNs to promote energy efficiency so as to extend network lifetime.

2. Clustering method has been shown effective for prolonging the network lifetime in the literature. This is beneficial in terms of energy efficiency in three ways: (1) Hierarchical structure distributes the energy usage to multiple sensors on multi-hop paths, thus, eliminating the quick expiration of the sensors away from the sink node; (2) Data aggregation at CHs is used to reduce redundancy and energy consumption in communication; (3) Periodic re-clustering can balance the energy consumption. However, we note that most of the studies in the clustered sensor network adopt the heuristic and/or localized method to select and vary CHs over the periods. Such methods may be biased from the long-term network lifetime perspective. Therefore, we are motivated to develop power-

aware mathematical models for a hierarchical cluster-based data flow structure.

3. Minimization of total energy consumption and the maximum energy used by a sensor are two common metrics for energy efficient topology and routing scheme. However, as discussed in section II.2.1, they do not always lead to a uniform energy consumption pattern. In both of these cases, we may face the issue of quick energy drainage which occurs at certain nodes in the first case and in the whole network in the second case. In this dissertation, we devise three mathematical models for integrated topology and routing decisions for data-gathering WSNs. The first two models include minimization of total energy usage in the system and the maximum energy used at a sensor node, respectively. Though they are commonly considered in devising communication protocols, this has not been done from an integrated mathematical modelling perspective as in our case. We consider these two models as benchmark models for our third proposed model, which minimizes the total energy and the range of remaining energy distribution in the network.

4. In devising our models, we consider the use of multiple sinks. Xue et al. (2005) consider multiple sinks, however, with known locations as opposed to our case where the locations are also determined. Gandham et al. (2003) illustrate the benefit of employing multiple sinks via suggesting an integer program to determine sink locations. However they adopt a flat-routing structure without CHs and aggregation.

5. In previous studies with data aggregation (e.g. Heinzelman et al. (2000, 2002); Kalpakis et al. (2003); Younis and Fahmy (2004)), we observe that aggregation of data into a single signal at each CH, i.e., regardless of the amount of data received, is common which is applicable in such cases as monitoring maximum

temperature in the sensor field. We consider cases where an overall view of a measure, such as spatial and temporal temperature/humidiy/pressure gradients in a large sensor field deployed for environmental monitoring, is of interest. To this end, we employ a general data aggregation approach at the CHs that represents the elimination of data redundancy.

In summary, we contribute to the current literature by developing the integrated mathematical models and their solution algorithms for cluster-based topology control and routing, along with consideration of multiple sinks, based on optimization techniques for the design of WSNs. Furthermore, we employ a generalized aggregation approach suitable for data-gathering related applications such as in environmental monitoring. To the best of our knowledge, this dissertation research is the first one in the design of WSNs to investigate the integrated topology control and routing based on optimization techniques with a unified generalization to the various settings in the literature summarized above.

CHAPTER III

INTEGRATED TOPOLOGY CONTROL AND ROUTING DECISIONS: A CLUSTER-BASED SENSOR NETWORK DESIGN PROBLEM

In this chapter, we address wireless sensor network design problem as an integrated topology control and routing model. We develop and examine three different objectives and their associated mathematical models where each defines a different *decision policy*. We aim to identify the one that provides the most prolonged network lifetime in a multi-period setting. The objectives include minimization of 1) *total or average energy usage in the system* 2) *maximum energy used at a sensor node*, and 3) *a weighted sum of the range of end-of-period remaining energy distribution at the sensor nodes and the average energy used in the system.*

Since our mathematical models dictate discrete optimization formulations, even small size instances are highly impractical to be solved using exact optimization methods. Thus, we develop efficient heuristic methods, including feasible solution construction heuristics and improvement heuristic approaches, that also incorporate exact optimization methods for solving subproblems. On the other hand, we also utilize our heuristic solutions in order to define effective cuts that improve the solution time of exact approaches. Among the objectives that we examine, we identify one that utilizes both the average energy use in the system and the range of remaining energy distribution at the sensor nodes as the most effective approach as a policy.

III.1. Problem Setting

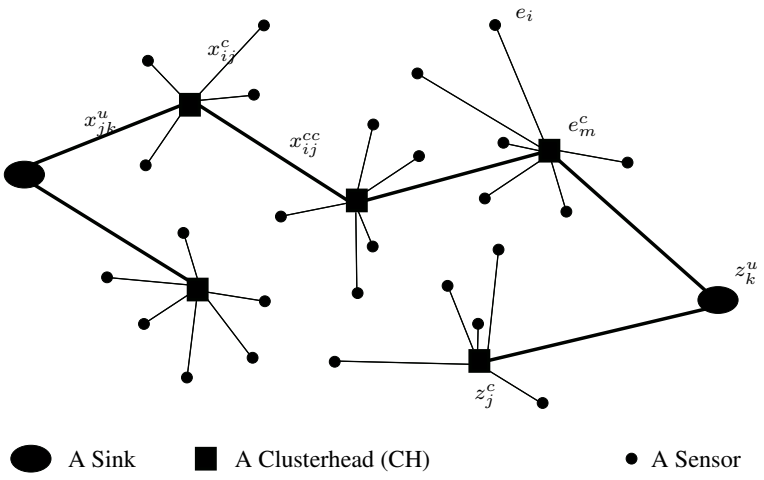
In this dissertation, we concentrate on time-driven sensor networks applications pertaining to monitoring ecological habitats (animals, plants, micro-organisms) where data collection is performed periodically. Once the sensors are randomly deployed

in the region of interest, they approximate their positions via, for example, triangulation utilizing some of the sensors with GPS units (Bulusu et al., 2000), and in some cases, without relying on GPS capabilities (Savvides et al., 2001). This sensor location information along with pre-configured sensor ids are routed to the sink nodes using, for example, a minimum cost forwarding protocol (Ye et al., 2001), they are used throughout the network lifetime by the user who, being also the main controller, typically has access to the information at the sink nodes via, for example, a satellite or fiber connections.

In a given period, we consider a hierarchical setting where data flow from sensor nodes to the sink nodes occurs via intermediate transfer (cluster-head) nodes. Figure 5 depicts a typical data flow pattern in the network and also the location and flow variables associated with the model. A group of sensors define a cluster, and they transmit the collected data to the associated cluster-head (CH) which in turn transmits the data to the sink nodes either directly or through other CHs. In addition to acting as a relay node, a CH processes collected data to eliminate duplications. Since, for a sensor, performing computations is much less energy consuming than communications, a hierarchial underlying topology results in higher energy efficiency. We also employ multiple sink nodes as they create an opportunity for better proximity to sensor nodes, thus, effectively reducing the energy consumption. In addition to the data flows, the sink and CH locations are selected from the associated candidate set and are determined by a centralized mathematical formulation in each period. This setting gives a dynamic topology where, at the end of each period, the energy information will be updated and the configuration will change in each period accordingly. Such a structure is beneficial in terms of energy efficiency in two ways. First, data-aggregation is applied within a cluster to reduce the amount of data to be transmitted. Second, the rotation of CHs and sink nodes helps to ensure a balanced

energy consumption.

Figure 5: An Example Solution and Notation for Topological Variables



The main task of a sensor node is to detect events, perform data processing, and then transmit the data. Power consumption can hence be divided into three parts: sensing, communication, and data processing. In WSNs, sensing energy represents only a small percentage of the total energy consumption and the majority of the consumed power is in computing and communication. Thus, we consider the energy dissipation only for communication and data processing in this dissertation study. We employ a widely adopted first order radio model (e.g. see Heinzelman et al. (2000)) in which energy dissipation is w (*joules/bit*) to run the circuitry and v (*joules/bit/m*^{2}) for the transmit amplifier. Then, transmitting x_{ab} (*bits*) of data from node a to node b dissipates $(w + v\, D^2_{ab})\, x_{ab}$ and to receive the same amount of data dissipates $w\, x_{ab}$ where D_{ab} is the distance (in meters) between a and b. In addition, to account for the power dissipation due to data aggregation/processing efforts at a CH, we employ a dissipation rate of c (*joules/bit*). We employ the values of the energy dissipation related parameters as in Heinzelman et al. (2000); specifically, $v = 100$ *pJ/bit/m*^{2}, $w = 50$ *nJ/bit*, and we use a c value of 5 *nJ/bit* (note that *pJ=pico-joules* and

nJ=nano-joules).

III.2. The Integrated Topology Control and Routing Models

In the models that follow, we assume that the sensors are randomly deployed in a two-dimensional field of finite area and each sensor node knows its position information either through a low-power Global Position System (GPS) receiver (Savvides et al., 2001) or some other way (such as location service and localization algorithms (Bulusu et al., 2000)). We assume that the sinks are not energy-constrained and are accessible by a user. The sink are located around the periphery of the sensor field. We also assume that a sensor collects data at an average rate and forwards this information to its CH for data aggregation and transmission. We assume that each wireless sensor node has an omni-directional antenna, so that a single transmission of a node can be received by all nodes within its vicinity which is a disk centered at the node.

We use an average aggregation ratio at a CH, mainly to reflect the elimination of data overlap, and note that this ratio is highly dependent on the specific application and network parameters such as deployment density. We also assume a fixed number of CHs being active in a period which can be controlled by the user. In our computational studies, similar to previous studies (Heinzelman et al., 2002; Soro and Heinzelman, 2009; Younis and Fahmy, 2004), we consider varying number of CHs, approximately 8-20% of the number of sensors.

We limit the usable energy that a sensor or a CH can use. We represent the usable amount as a fraction p of the total available energy at a sensor and refer to it also as *topology control parameter*. This provides the ability to manage topology control implicitly. This is an important characteristic because allowing a sensor's whole energy to be usable can easily make the sensor vulnerable to quick energy

depletion since it can be selected as a CH repeatedly.

We define the following notation:

Model Parameters

$\mathcal{I}$	set of sensors, $i \in \mathcal{I}$,
$\mathcal{J}$	set of candidate CHs, $j \in \mathcal{J}$,
$\mathcal{K}$	set of candidate sinks, $k \in \mathcal{K}$,
R_i	data generation rate (*bits/unit-time*) at a sensor i,
D_{pq}	distance (m) between any two nodes p and q,
H	number of required CHs,
U	number of required sinks,
s	an average data aggregation ratio,
E_i	available energy (*Joules*) at a sensor i,
p	fraction of E_i reserved for usage at a sensor,
T	the length of a period.

Decision Variables

x_{ij}^{c}	fraction of data flow per unit time from a sensor i to a CH j,
x_{ij}^{cc}	data flow per unit time from a CH i to a CH j,
x_{jk}^{u}	data flow per unit time from a CH j to a sink node k,
z_{j}^{c}	1 if a node j is setup as a CH, 0 o.w.,
z_{k}^{u}	1 if a node k is setup as a sink, 0 o.w.,
e_i	energy consumed by a sensor node i,
e_{m}^{c}	energy consumed by a CH m,
E_{max}^{C}	maximum energy consumed at a sensor,
E_{max}^{R}	maximum remaining energy at a sensor,
E_{min}^{R}	minimum remaining energy at a sensor.

Due to common characteristics such as underlying network topology and flow structure and energy consumption calculations, we have the following set of common constraints for our models.

Common Constraints

$$\sum_{k\in\mathcal{K}}(w+v\,D_{mk}^{2})\,T\,x_{mk}^{u}+\sum_{j\in\mathcal{J}\setminus\{m\}}(w+v\,D_{mj}^{2})\,T\,x_{mj}^{cc} + \sum_{j\in\mathcal{J}\setminus\{m\}} w\,T\,x_{jm}^{cc}+\sum_{i\in\mathcal{I}}(w+c\,s)\,R_i\,T\,x_{im}^{c}=e_{m}^{c} \qquad \forall\, m\in\mathcal{J} \tag{3.1}$$

$$\sum_{j\in\mathcal{J}}(w+v\,D_{ij}^{2})\;R_i\,T x_{ij}^{c}=e_i \qquad \forall\, i\in\mathcal{I} \tag{3.2}$$

$$\sum_{k\in\mathcal{K}}x_{mk}^{u}+\sum_{j\in\mathcal{J}\setminus\{m\}}x_{mj}^{cc}-\left(\sum_{j\in\mathcal{J}\setminus\{m\}}x_{jm}^{cc}+(1-s)\sum_{i\in\mathcal{I}}R_i\,x_{im}^{c}\right)=0\;\forall\, m\in\mathcal{J} \tag{3.3}$$

$$\sum_{j\in\mathcal{J}}x_{ij}^{c}=1 \qquad \forall\, i\in\mathcal{I} \tag{3.4}$$

$$x_{ij}^{c} \leq z_{j}^{c} \qquad \forall\, i \in \mathcal{I}, \forall\, j \in \mathcal{J} \tag{3.5}$$

$$x_{mj}^{cc} \leq \sum_{i \in \mathcal{I}} R_i\, z_{j}^{c} \qquad \forall\, m,\, j \in \mathcal{J} \tag{3.6}$$

$$x_{jk}^{u} \leq \sum_{i \in \mathcal{I}} R_i\, z_{k}^{u} \qquad \forall\, j \in \mathcal{J}, \forall\, k \in \mathcal{K} \tag{3.7}$$

$$x_{jk}^{c} \leq \sum_{i \in \mathcal{I}} R_i\, z_{j}^{c} \qquad \forall\, j \in \mathcal{J}, \forall\, k \in \mathcal{K} \tag{3.8}$$

$$e_i \leq p\, E_i \qquad \forall\, i \in \mathcal{I} \tag{3.9}$$

$$e_{j}^{c} \leq p\, E_j \qquad \forall\, j \in \mathcal{J} \tag{3.10}$$

$$\sum_{j \in \mathcal{J}} z_{j}^{c} = H \tag{3.11}$$

$$\sum_{k \in \mathcal{K}} z_{k}^{u} = U \tag{3.12}$$

$$z_{j}^{c},\, z_{k}^{u} \in \{0, 1\} \qquad \forall\, i \in \mathcal{I},\ j \in \mathcal{J} \tag{3.13}$$

$$x_{ij}^{c},\, x_{ij}^{cc},\, x_{jk}^{u},\, e_i,\, e_{j}^{c} \geq 0 \qquad \forall\, i \in \mathcal{I},\ j \in \mathcal{J},\ k \in \mathcal{K} \tag{3.14}$$

Constraints (3.1) and (3.2) assign the values of the total energy consumed by a CH and a sensor, respectively. Note that the energy dissipation due to data aggregation is embedded in the last term of the left-hand side in constraint (3.1) in such a way that more aggregation (a higher s value) results in higher energy dissipation. Constraints (3.3) state the data flow balance at each CH , which also ensure that the data collected at a sensor node is aggregated with data from other sensors only once after it is transferred to the associated CH. The aggregation ratio s represents the ratio of the data eliminated due to redundancy. Constraint (3.4) guarantees that the data generated at each sensor node reaches some CH. Constraints (3.5)–(3.8) assign the values of binary variables related to CH and sink location selections. Constraint

sets (3.9) and (3.10) ensure that the total energy consumed at a node cannot exceed the total available energy at the corresponding sensors. Constraints (3.11) and (3.12) establish the required number of CHs and sinks, respectively, and (3.13)-(3.14) include the integrality and non-negativity of the decision variables.

Alternative Models

(M1) Minimize the total energy used:

$$\text{Min} \quad \sum_{m \in \mathcal{J}} e_m^c + \sum_{i \in \mathcal{I}} e_i \tag{3.15}$$

$$\text{subject to} \quad (3.1) - (3.14)$$

(M2) Minimize the maximum energy consumed by a sensor node:

$$\text{Min} \quad E_{max}^C \tag{3.16}$$

$$\text{subject to} \quad (3.1) - (3.14)$$

$$e_i \leq E_{max}^C \qquad \forall\, i \in \mathcal{I} \tag{3.17}$$

$$e_j^c \leq E_{max}^C \qquad \forall\, j \in \mathcal{J} \tag{3.18}$$

$$E_{max}^C \geq 0 \tag{3.19}$$

Constraint sets (3.17) and (3.18) impose a variable upper bound on the energy consumed by any sensor node.

(M3) Minimize the weighted (where t is the weight) sum of average energy consumption and range of remaining energy levels:

$$\text{Min} \quad t \left(\sum_{m \in \mathcal{J}} e_m^c + \sum_{i \in \mathcal{I}} e_i \right) / |\mathcal{I}| + E_{max}^R - E_{min}^R \tag{3.20}$$

subject to $(3.1) - (3.14)$

$$E^R_{min} \leq E_j - e^c_j \qquad \forall\, j \in \mathcal{J} \qquad (3.21)$$

$$E^R_{min} \leq E_i - e_i \qquad \forall\, i \in \mathcal{I} \qquad (3.22)$$

$$z^c_j\, E_j - e^c_j \;\leq\; E^R_{max} \qquad \forall\, j \in \mathcal{J} \qquad (3.23)$$

$$(1 - z^c_i)\, E_i - e_i \;\leq\; E^R_{max} \qquad \forall\, i \in \mathcal{I} \qquad (3.24)$$

$$E^R_{max},\, E^R_{min} \geq 0 \qquad (3.25)$$

Constraint set pairs (3.21)-(3.22) and (3.23)-(3.24) express the the minimum and the maximum remaining energy levels for each sensor node, respectively.

Recall that once a model is adopted, it is solved successively for each period until the end of the deployment cycle marked by the infeasibility of the model. The objectives in **(M1)** and **(M2)** are commonly considered both in topology control and routing studies, although not necessarily in a mathematical modelling context as in this study. However, we observe that, when a lifetime measure is considered, they do not exactly capture the energy depletion pattern that we would like to see in the network.

More specifically, **(M1)** minimizes the overall energy dissipation, which may lead to an energy drainage at certain nodes due to their successive usage in several periods if they happen to be on more favorable paths requiring less transmission energy. On the other hand, since **(M2)** aims to minimize the highest energy usage at a sensor node, it can do so by creating long multi-hop routes from sensors to the sinks so that each sensor dissipates only a small amount of energy. In both of these cases, we face the issue of quick energy drainage which occurs at certain nodes in **(M1)** case and in the whole network in **(M2)** case. Limiting the usable energy reserve at sensor node

can alleviate this problem by only making a fraction of the energy available at the sensor nodes in each period.

We develop the third model in order to address the drawbacks of **(M1)** and **(M2)** via the objective function formation that promotes, from the perspective of lifetime maximization, a more favorable energy depletion pattern in a network. Specifically, the objective in **(M3)** directly addresses the energy usage and also the variation in the remaining energy distribution at the sensor nodes by minimizing its range. That is, **(M3)** aims to distribute the energy usage across the network uniformly.

III.3. Experiment Data

In our following numerical studies, we generate $|\mathcal{I}|$ sensor coordinates randomly using a uniform distribution in a square of size $N\,meters(m)$. The candidate sites for sinks, $\mathcal{K}$, are also generated randomly on the periphery of the sensor field. We set the period length as $T = 4000$ *time-units* and the aggregation ratio as $s = 0.3$. In addition, we set the weight for the average energy dissipation component of the objectives **(M3)** as $t = 5$ which we determined as a reasonably good value after some empirical testing. The computational studies are performed on a machine with Pentium D 3.2 GHz CPU and 2.0GB RAM and the algorithms are implemented in C++ utilizing STL (Standard Template Library) and Concert Technology when CPLEX was used.

III.4. Preliminary Analysis of the Models

In order to gain insights into the characteristics of the models and as well as compare them based on various criteria both in single-period and multi-period settings, we solve a set of small size instances to optimality using the exact branch-and-cut implementation in CPLEX 9.0 with default parameters. In particular, the preliminary

analysis is aimed to examine the remaining energy distribution characteristics (mean and variance) at the end of the single-period solutions associated with these models; the impact of considering a high-energy subset of $\mathcal{I}$ as the set of candidate CHs, which we denote by $\mathcal{I}^R$, and the effects of parameters including the number of user nodes and usable energy percentage on network lifetime.

III.4.1. Single-Period Characteristics

Due to our specific objective of prolonging the network lifetime by obtaining the maximum number of fixed-length periods under limited energy resources, it is very important to understand the energy status characteristics in the system as we move from one period to another. For this reason, we examine the remaining energy distribution after a typical period for which we solve our models to determine CH as well as the sink locations and data routing. Furthermore, it is also important for the purposes of computational efficiency in solving the optimization models that we understand the impact of considering $\mathcal{I}^R$ on the the optimum solution of the models. Employing $\mathcal{I}^R$ instead of $\mathcal{I}$ is also useful from the energy efficiency perspective for a prolonged network lifetime since it promotes, in each period, the protection of low energy sensors from depleting their energy quickly. To determine the set of high energy sensors in a period, $\mathcal{I}^R$, we use a threshold value TH_Ψ calculated as $\Psi\%$ of the average initial energy level at the sensors, i.e., $TH_\Psi = (\Psi/100) * (\sum_{i\in\mathcal{I}} E_i/|\mathcal{I}|)$. We consider 18 problem sets where each includes 10 instances generated randomly with fixed N and $|\mathcal{K}|$ values of 50 and 8, respectively. We vary values of $|\mathcal{I}|$, U, and H as summarized in Table 1. Note that, in all of the instances, we assume that the initial energy levels at the sensors are uniformly distributed in the range [0.1, 0.5] J and fully available.

In examining the remaining energy distribution moments, we employ the set $\mathcal{I}$

Table 1: Parameter values for the data sets

Set	$\|\mathcal{I}\|$	U	H	Set	$\|\mathcal{I}\|$	U	H	Set	$\|\mathcal{I}\|$	U	H
1	15	1	2	7	20	1	2	13	25	1	2
2			3	8			3	14			3
3			4	9			4	15			4
4		2	2	10		2	2	16		2	2
5			3	11			3	17			3
6			4	12			4	18			4

as $\mathcal{J}$ and solve the random instances using our models. The averages of means and the variances of the corresponding distributions for each data set are given in Figures 6(a) and 6(b), respectively. In Figure 6(a), we observe that there is not a substantial difference among the models in terms of mean remaining energy values. For each model, a slight difference occurs when the $|\mathcal{I}|$ value changes in such a way that the remaining energy means decrease as the $|\mathcal{I}|$ increases (every six data sets), however, this is expected since the larger number of sensors generates more data and lead to a higher energy consumption in the system. Also, we observe that, for a fixed $|\mathcal{I}|$ value corresponding to six data sets, there is an upward trend in remaining energies due to the increase in U and H values. On the other hand, in Figure 6(b), it is clear that the model **(M3)** provides a significantly less variance in remaining energy distribution. Given the variation in the initial energy distribution at the sensors, this observation is particularly interesting in showing the impact of minimizing the range along with the average of remaining energy distribution.

To examine the impact of employing an $\mathcal{I}^R$ that involves only the sensors whose energy level is higher than $TH_{\Psi=60}$, we solve each model first assuming $\mathcal{J} = \mathcal{I}$ and, then, $\mathcal{J} = \mathcal{I}^R$ with objective values of $Z_{\mathcal{I}}$ and $Z_{\mathcal{I}^R}$, respectively. Hence, we compute the percentage gap between objective function values as $G^R(\%) = 100*(Z_{\mathcal{I}^R} - Z_{\mathcal{I}})/Z_{\mathcal{I}}$.

For each model and data set, we compute the average G^R values over ten instances, as reported them in Figure 6(c). We observe that the **(M3)** tends to heavily utilize high energy sensors as CHs in their optimal solutions. This is illustrated by the fact that the optimal objective function values with $\mathcal{J} = \mathcal{I}$ and $\mathcal{J} = \mathcal{I}^R$ are very close, as shown in Figure 6(c).

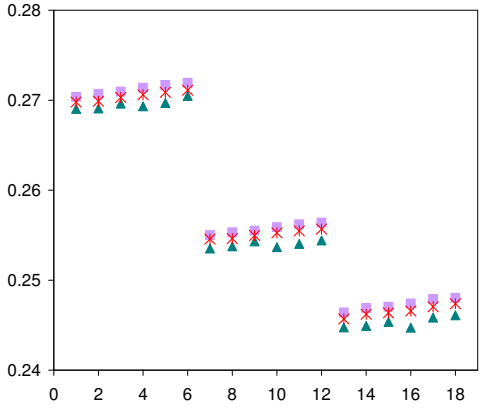
III.4.2. Multiple-Period Characteristics

In our multi-period setting, we empirically examine three specific network attributes – the number of user nodes U, the usable energy percentage p and the period length T – that we specifically incorporate for the purpose of prolonging network lifetime. For this, we employ a data set of 10 sample instances and optimally solve our models for each instance with varying p, U and T values using exact branch-and-cut implementation in CPLEX 9.0 with default parameters. We use the data set 5 in Table 1.

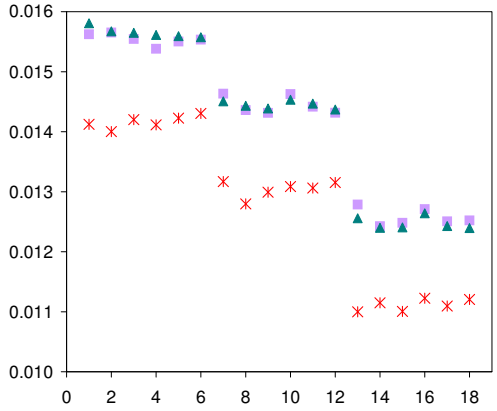
The Impact of Usable Energy Fraction p on Network Lifetime

First, we specifically examine the impact of topology control parameter p, which is the fraction of total energy available at a sensor made usable in a period, on network lifetime under the three models **(M1)**, **(M2)**, and **(M3)**. For a given model, we run each instance in the data set for a full deployment cycle (lifetime) ten times, where each one of ten runs corresponds to *initial* p values (p_I) ranging from 0.1 to 1.0. We start an instance with an initial p_I value as the current p value and, whenever in a period the solution to the model is infeasible, we increase the current p value by an increment (*inc*) value for which we assume a value of 0.1. The most recent value of p in a period is carried as the current p value for the next period. The condition in which the p reaches a value of 1.0 and the model's solution is infeasible marks the end of

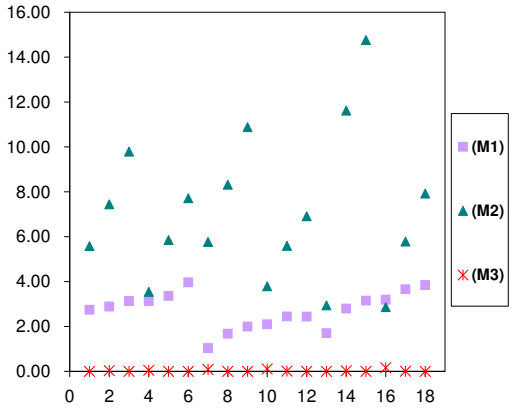
Figure 6: Comparison of Models using Single-Period Solutions



(a) Ave. of Remaining Energy Means



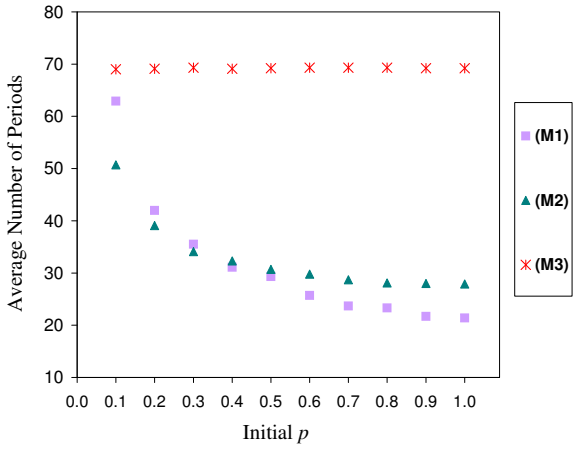
(b) Ave. of Remaining Energy Variances



(c) Ave. of G^R Values (%)

the deployment cycle for the instance. In general, a topology control scheme specified as such can be represented as (p_I, inc) and it should be calibrated, i.e., the values of p_I and inc should be determined for a specific application via experimentation. In Figure 7, we summarize the results for all three models where the lifetimes are averaged over 10 instances for each initial p value.

Figure 7: Average Network Lifetimes under Varying Initial p Values



We observe that the performance of **(M1)** is highly sensitive to the initial value of p. If we do not restrict the maximum energy usage for each node, i.e. start the deployment cycle with a large p value, the **(M1)** performs very poorly. It seems that determining a good p value, possibly via a conservative use of energy from the beginning of a deployment cycle and a calibration of its increments, provides a good lifetime measure. Although not as significant, **(M2)** is also sensitive to the initial value of p. On the other hand, it is interesting to observe that **(M3)** is influenced very little, if at all, by the choice of p_I value, i.e., it is highly robust to changes in the amount of energy reserved for usage at the sensors. We can explain these differences in the performance of the models with respect to the p value as follows. If the energy usage is not constrained in **(M1)**, then the same underlying network,

given by the optimum model solution, is repeatedly obtained. This leads to a quick energy depletion at certain sensors, and, thus, shortening the network lifetime. **(M2)**, which minimizes the maximum energy depletion at a sensor node, does not have any particular means to avoid a similar repetitive use of certain nodes when coupled with the use of a conservative p; therefore, even having a conservative strategy with a low p is not very helpful in terms of prolonging the network lifetime. **(M3)** is a multi-objective approach and, in general, it incorporates remaining energy measures into the optimization and attempt to address energy dissipation both in terms of minimizing its average in the network and its variation among sensor nodes. Thus, it promotes a favorable transition in terms of energy status in the network from period-to-period. The choice of p is insignificant in **(M3)** because its role is already embedded in the models very effectively, leaving no question about how the value of p should be calibrated from period-to-period in order to prolong network lifetime.

The Impact of Number of Sinks U on Network Lifetime

Second, to study the effect on network lifetime of employing multiple sink nodes, we consider the varying values of number of sinks U in the $[1, 4]$ range. Based on our above observations on the (p_I, inc), we consider two cases including (1.0, 0.0) and (0.1, 0.1). We further note that although in the first period of the first deployment cycle the initial energy levels at the sensors are equal, when a redeployment in a WSN is performed, the initial energy levels at the sensors are expected to be varying. Thus, to also examine any bias on the impact of p and U due to having different initial energy levels at the sensors at the beginning of a deployment cycle, we consider cases where the initial energy levels at the sensors are equal to 0.5 J, and they are randomly drawn from a uniform distribution in the range [0.1, 0.5] J. The average network lifetimes over the 10 instances is summarized in Table 2.

Table 2: Average network lifetimes under varying E_i, U, and p values

		Same Initial Energy Levels				Different Initial Energy Levels			
Model	$(p_I, inc)\,/\,U$	1	2	3	4	1	2	3	4
(M1)	$(1.0, 0.0)$	19	21	23	23	8	10	10	10
	$(0.1, 0.1)$	55	63	64	64	27	31	31	31
(M2)	$(1.0, 0.0)$	27	28	28	28	8	9	9	9
	$(0.1, 0.1)$	50	51	51	51	24	25	26	26
(M3)	$(1.0, 0.0)$	61	69	71	71	26	30	30	31
	$(0.1, 0.1)$	61	69	71	71	29	32	33	33

We observe that employing multiple sink nodes helps to achieve longer network lifetimes. This is generally true for each model, independent of the initial energy levels. However, the added benefit from multiple sinks diminishes after a certain value of U. Thus, there appears a critical value of U that provides the most prolonged network lifetime. In a real application, multiple sinks also provide added benefits due to the better fault tolerance abilities they offer. Furthermore, **(M3)** outperform the other models in terms of lifetime, and, clearly, it is very little, if any, impacted by the choice of p regardless of the nature of initial energy levels.

III.5. Heuristic Approaches

Our models of interest dictate mixed integer programming problems and, for relatively large-scale instances, the use of branch-and-cut solution algorithms (as in CPLEX) is not helpful due to high memory and runtime requirements. Therefore, we study the development of efficient heuristic solution approaches which take advantage of the availability of explicit model formulations and underlying model and solution characteristics.

While developing our solution approaches and conducting our computational studies using both the heuristic methods and the exact branch-and-cut approach, we also incorporate objective function value-based cuts into the models (in exact approach) or subproblems (in heuristic approach) in order to significantly improve their solution times. In particular, supposing that an upper bound $\hat{Z}$ for the model of interest is available, then for **(M1)**, **(M2)**, and **(M3)**, we utilize the cut inequalities

$$\sum_{m \in \mathcal{J}} e_m^c + \sum_{i \in \mathcal{I}} e_i \leq \hat{Z}, \tag{3.26}$$

$$E_{max}^C \leq \hat{Z}, \tag{3.27}$$

and

$$t \left(\sum_{m \in \mathcal{J}} e_m^c + \sum_{i \in \mathcal{I}} e_i \right) / |\mathcal{I}| + E_{max}^R - E_{min}^R \leq \hat{Z}, \tag{3.28}$$

respectively. Furthermore, for **(M3)**, using the following two logical inequalities

$$E_{max}^R - E_{min}^R \geq 0 \qquad \text{and}$$

$$\left(\sum_{i \in \mathcal{I}} E_i - (\sum_{m \in \mathcal{J}} e_m^c + \sum_{i \in \mathcal{I}} e_i) \right) / |\mathcal{I}| \leq E_{max}^R$$

in conjunction with (3.28), we obtain an additional cut inequality given by

$$t \left(\sum_{i \in \mathcal{I}} E_i / |\mathcal{I}| - E_{max}^R \right) \leq \hat{Z} \tag{3.29}$$

Thus, we are able to benchmark the quality of our heuristic solutions employing a wider set of problem instances. We develop an efficient overall solution procedure that addresses all of the three models, **(M1)**, **(M2)** and **(M3)**, which differ only in the initial solution construction component. Before we present this complete procedure, we discuss its components in detail.

III.5.1. Solution Representations and Subproblems

Although the models we consider differ structurally, they embody the same sets of binary variables. Thus, we utilize the binary variables to obtain two solution representations. In the first one, we represent a solution by a set of fixed CH locations given by $\mathcal{C} = \{j \in \mathcal{J} : z_j^c = 1\}$. To evaluate the goodness (objective value) of such a solution, we solve a subproblem, denoted by (**SubC**), by simply fixing the corresponding binary variables z_j^c at 1 in the model of interest. Thus, we obtain the sink locations and data routing along with the objective value $Z(\mathcal{C})$. In the second case, we represent a solution by the fixed sink locations, i.e., $\mathcal{D} = \{k \in \mathcal{K} : z_k^u = 1\}$. Similarly to the previous representation, in order to evaluate the goodness of a given solution, we solve a subproblem (**SubU**), which is derived from the model of interest by using $\mathcal{D}$, and obtain the CH locations and data routing along with the objective value $Z(\mathcal{D})$. Note that thirdly we can consider the combination of the above two representations, however, our numerical studies indicate that adoption of this representation of solution algorithms is highly ineffective, thus, we do not pursue its use. Because the solution space is smaller due to a smaller number of candidate sink locations compared to candidate CH locations, a solution search approach based on $\mathcal{D}$ and (**SubU**), in general, is expected to provide better quality solutions. However, solution times are expected to be longer since the solution of a subproblem (**SubU**) is more time-consuming than the solution time of a (**SubC**) due to the larger number of variables.

III.5.1.1. Construction Heuristics

We devise two different construction heuristic methods, one for **(M2)** and the other for **(M1)** and **(M3)**. In doing so, we try to utilize the model characteristics to obtain good initial solutions which contribute to better performance during the improvement

stage of the overall procedure.

Construction Heuristic for (M2)

Recall that in **(M2)**, we aim to minimize the maximum energy consumed by a sensor. Then, in a solution, there is at least one sensor whose energy consumption (highest in all the sensors) uniquely determines the objective value. Since the energy consumption is largely due to data transmission, it makes sense to employ a set of CHs that are spread all over the sensor field. This way, no individual sensor or a small set of sensors are subjected to exceptionally high energy consumption. To ensure this spread of CHs over the sensor field, in the `ConstructM2()` heuristic, given in Algorithm 1, we proceed as follows. First, we note that `ConstructM2()` is a multi-start approach, i.e., its core algorithm (lines 3-15) is run *Maxiter* times; and the best of all the *Maxiter* solutions obtained is the final solution.

Each start first randomly picks a single node as a CH from the candidate set of CHs (lines 4-5). Then the rest of the CHs ($H-1$ of them) is picked based on distance in such a way that a good spread of CHs in the sensor field is promoted. Distance $d(j,\mathcal{C})$ between a node j and the set of already selected CHs ($\mathcal{C}$) is measured as $\min_{i\in\mathcal{C}} d(j,i)$. We pick the next CH as being the farthest away from the already selected CHs (lines 7-9). Once a solution $\mathcal{S}^c$ is obtained after all of the CHs are picked this way, its goodness is evaluated by solving the problem **SubC** described above. If $\mathcal{S}^c$ is better than the best solution so far ($\mathcal{S}^b$), then it becomes the new $\mathcal{S}^b$ (lines 11-14). In our computational studies, we use a *Maxiter* value of $|\mathcal{J}|/2$ if $|\mathcal{J}| < 100$ and $|\mathcal{J}|/5$ otherwise.

Algorithm 1 Procedure `ConstructM2()`

initialize $Z(\mathcal{S}^b) = \infty$, *Maxiter*;
while *Maxiter* > 0 **do**
  $\mathcal{C} = \emptyset$, $h = 1$;
  Randomly pick a node i from $\mathcal{J}$;
  $\mathcal{C} = \mathcal{C} \bigcup \{i\}$, $\bar{\mathcal{C}} = \mathcal{J} \setminus \{i\}$;
  while $h < H$ **do**
    $j^* = \arg\max\{d(j, \mathcal{C}) : j \in \bar{\mathcal{C}}\}$;
    $\mathcal{C} = \mathcal{C} \bigcup \{j^*\}$, $\bar{\mathcal{C}} = \bar{\mathcal{C}} \setminus \{j^*\}$;
    $h = h + 1$;
  end while
  $\mathcal{S}^c = \mathcal{C}$, solve (**SubC**) to obtain $Z(\mathcal{S}^c)$;
  if $Z(\mathcal{S}^c) < Z(\mathcal{S}^b)$ **then**
    $\mathcal{S}^b = \mathcal{S}^c$, $Z(\mathcal{S}^b) = Z(\mathcal{S}^c)$;
  end if
  Maxiter = *Maxiter*-1;
end while
Return $\mathcal{S}^b$ and $Z(\mathcal{S}^b)$

Construction Heuristic for (M1) and (M3)

Both **(M1)** and **(M3)** mainly aim to minimize energy consumption with an additional requirement for **(M3)**, whereby the range of the remaining energy distribution is also minimized. Since the data flow in the network is toward the sinks, it is preferable to ensure that some CHs are selected close to the periphery while the remaining CHs are chosen from the center of the sensor field so that energy efficient natural paths can be established towards sinks. The `ConstructM1M3()` heuristic, given in Algorithm 2, aims to achieve this by dividing the whole sensor field in two nested parts, a box centered in the sensor field and the band around it. In particular, the candidate CHs in the first part, denoted by $\mathcal{F}_1$, are the ones that are in a square of size β centered in the sensor field. Then, the set of candidate CHs in the second part (close to periphery of the field), called $\mathcal{F}_2$, is simply given by $\mathcal{J} \setminus F_1$.

In this process, we also attempt to avoid coincidentally well-positioned sensors (from an energy dissipation minimization point-of-view) being selected as CHs repeatedly in successive periods and to protect low-energy sensors from being selected as CHs. For this purpose, we consider only a subset of sensors with higher-energy as the set of candidate CHs $\mathcal{J}$. We denote this subset as $\mathcal{I}^R$ since it is a subset of sensor set $\mathcal{I}$. Specifically, to determine the $\mathcal{I}^R$ set, we use a threshold value TH_Ψ calculated as $\Psi\%$ of the average initial energy level at the sensors, i.e., $TH_\Psi = (\Psi/100) * (\sum_{i \in \mathcal{I}} E_i/|\mathcal{I}|)$ and $\mathcal{I}^R = \{i \in \mathcal{I} : E_i \geq TH_\Psi\}$.

The `ConstructM1M3()` heuristic is also a multi-start approach as `ConstructM2()` and we use the same maximum number of iterations, *Maxiter*, as in `ConstructM2()`. In each start, we proceed as follows (lines 5-23 in Algorithm 2). We first pick about a fraction α of total required CHs in $\mathcal{F}_2$ (periphery band), specifically we pick a total of $\lfloor \alpha * H \rfloor$ CHs from $\mathcal{F}_2$. For this, we start by randomly picking a CH and then choose the rest of the CHs one at a time at the median distance from the currently selected ones (lines 5-12). Then, the rest of the CHs (i.e., $H - \lfloor \alpha * H \rfloor$) are picked from set $\mathcal{F}_1$ similarly (lines 13-18). We determine the goodness of the obtained solution $\mathcal{S}^c$ by solving the associated **SubC**. We update the best solution $\mathcal{S}^b$ by $\mathcal{S}^c$ if necessary (lines 19-23).

In our implementations, before we choose the CHs, we check the number of candidates in $\mathcal{F}_1$ and $\mathcal{F}_2$ starting with an α value of 0.80 and a β (size of the center box) value determined as $\lfloor N/2 \rfloor - 2$ where N is the size of a square sensor field. If there are not enough candidates in sets $\mathcal{F}_1$ or $\mathcal{F}_2$, we increase or decrease β by one unit. Note that, by changing the value of α, the ratio of CHs located around the periphery to those in the inner part of the sensor field can be changed. Also, we use a Ψ value of 60 in our numerical studies.

Algorithm 2 Procedure `ConstructM1M3()`

initialize $F_1 = F_2 = \emptyset$, $Z(\mathcal{S}^b) = \infty$, *Maxiter*;
Form $\mathcal{F}_1 \subseteq \mathcal{J}$ using the center square of size β;
$\mathcal{F}_2 = \mathcal{J} \setminus \mathcal{F}_1$;
while *Maxiter*> 0 **do**
  $\mathcal{C} = \emptyset$, $h = 1$;
  Randomly pick a node i from $\mathcal{F}_2$;
  $\mathcal{C} = \mathcal{C} \bigcup \{i\}$, $\bar{\mathcal{C}} = \mathcal{F}_2 \setminus \{i\}$;
  while $h < \lfloor H * \alpha \rfloor$ **do**
    Pick j^* which has the median $d(j, \mathcal{C})$, $\forall\, j \in \bar{\mathcal{C}}$;
    $\mathcal{C} = \mathcal{C} \bigcup \{j^*\}$, $\bar{\mathcal{C}} = \bar{\mathcal{C}} \setminus \{j^*\}$;
    $h = h + 1$;
  end while
  $\bar{\mathcal{C}} = \mathcal{F}_1$;
  while $h < H$ **do**
    Pick j^* which has the median $d(j, \mathcal{C})$, $\forall\, j \in \bar{\mathcal{C}}$;
    $\mathcal{C} = \mathcal{C} \bigcup \{j^*\}$, $\bar{\mathcal{C}} = \bar{\mathcal{C}} \setminus \{j^*\}$;
    $h = h + 1$;
  end while
  $\mathcal{S}^c = \mathcal{C}$, solve (**SubC**) to obtain $Z(\mathcal{S}^c)$;
  if $Z(\mathcal{S}^c) < Z(\mathcal{S}^b)$ **then**
    $\mathcal{S}^b = \mathcal{S}^c$, $Z(\mathcal{S}^b) = Z(\mathcal{S}^c)$;
  end if
  Maxiter = *Maxiter*-1;
end while
Return $\mathcal{S}^b$ and $Z(\mathcal{S}^b)$

III.5.2. Solution Improvement Procedures

In this section, we devise two search procedures that we later utilize in our complete procedure for improvement purposes after applying the construction heuristic described above. In the first method, we employ the CH set $\mathcal{C}$ as the solution representation and accordingly utilize subproblem (**SubC**) to evaluate the goodness of the solution. In the second one, we represent a solution by a set of selected sink locations $\mathcal{D}$ and employ subproblem (**SubU**) and the solution value obtained in the

first search procedure. The second procedure is similar to the construction heuristic `ConstructM2()` which operates using only $\mathcal{C}$ and generates an initial solution without using any a priori solution information.

The `SearchSubC()` Procedure

This procedure starts with a feasible solution, denoted by $\mathcal{S}^b_{init}$, which includes the CHs as determined by a construction heuristic. Inspired by the variable depth search approach presented by Lin and Kernighan (1973), it searches the solution space by applying an extended form of exchange neighborhood on $\mathcal{C}$. In each iteration of the algorithm, CH set is modified by making a number of non-CHs serve as CHs while unassigning the same number of CHs as non-CHs in the current solution.

In particular, we employ an h-exchange neighborhood function where, in general terms, starting with a 1-exchange (a single pair-exchange of a CH and a non-CH sensor), h is increased up to h_{max} as long as the solution goodness is monotonously improved. However, as we explain later in Section III.5.3, we assume that only a *subset* of the existing CHs can be exchanged (assigned as non-CH) in generating neighborhood solutions. For a current solution $\mathcal{S}^c$, we use a derived set, $\mathcal{S}^c_{\text{free}}$, to represent this subset of exchangeable CHs. Furthermore, we note that the variable depth search approach is a technique that searches a large solution space as it generates a large number of neighboring solutions of a current solution in each iteration. We enhance its efficiency further by utilizing both the proximity and the energy level information at the sensor nodes leading to a consideration of restricted and dynamic neighborhoods for CHs. In particular, we use distance and energy information as follows: for a CH to be exchanged, we consider only the non-CH sensors in $\mathcal{J}$ which are within its ρ radius and have at least a TH_Ψ energy level. Then, the neighborhood for each node is clearly restricted to a subset of $\mathcal{J}$. Also, it is very likely that

the possible non-CHs that can be exchanged with a CH vary in each period due to changing energy levels. This neighborhood generation approach helps to limit the neighborhood size with only a small impact on the network lifetime since the low energy sensors are prevented from being repeatedly employed as CHs.

Algorithm 3 Procedure `SearchSubC()`

initialize $h = 1$, h_{max}, $G^* = 0$, $g_0 = \infty$,
  $\mathcal{S}^b_{init}$ (from a construction heuristic),
  Maxiter, G_{max}, ρ, ρ_{min}, $\mathcal{J}$;
while $h \leq h_{max}$ and $G^* = g_{h-1}$ **do**
  for $j = 1$ to h **do**
    $\mathcal{S}^c = \mathcal{S}^b_{init}$
    for $i = 1$ to *Maxiter* **do**
      Randomly pick j nodes from $\mathcal{S}^c_{\text{free}}$
      Generate a neighborhood solutions set Ω_i;
      For each $\mathcal{C} \in \Omega_i$, solve (**SubC**) to obtain $Z(\mathcal{C})$;
      $\mathcal{S}_i = \arg\min\{Z(\mathcal{C}) : \mathcal{C} \in \Omega_i\}$
      if $Z(\mathcal{S}_i) < Z(\mathcal{S}^c)$ **then**
        $\mathcal{S}^c = S_i$;
      end if
    end for
    $g_j = Z(\mathcal{S}^b) - Z(\mathcal{S}^c)$;
    if $g_j > 0$ **then**
      $\mathcal{S}^b = \mathcal{S}^c$, $Z(\mathcal{S}^b) = Z(\mathcal{S}^c)$;
    end if
  end for
  $g_h = \sum_{j=1}^{h} g_j$;
  if $g_h > G^*$ **then**
    $G^* = g_h$;
  end if
  if $G^* > G_{max}$ and $\rho \geq \rho_{min}$ **then**
    $\rho = \rho - 1$;
  end if
  $h = h + 1$;
end while
Return $\mathcal{S}^b$ and $Z(\mathcal{S}^b)$

The `SearchSubC()` heuristic proceeds as detailed in Algorithm 3. The outside while loop, lines 2–28, controls the search procedure by conditioning on the maximum allowable exchange neighborhood size and the continuity of improvements over the best solution $\mathcal{S}^b$. Since, as h increases, the generation of a neighborhood and the evaluation of the solutions require longer runtimes, we limit the exchange neighborhood size by h_{max}. For a given h value, we generate the neighborhoods of j-exchange, $j = 1, \ldots, h$, in order (lines 3–18). For each j, we perform at most *Maxiter* j-exchanges (lines 5–13). For each such exchange i, we randomly pick j nodes from the $\mathcal{S}^c_{\text{free}}$ set of the current solution and generate a neighborhood solution set Ω_i. The generation of Ω_i for $j = 1$ is straightforward. For $j \geq 2$, we proceed by generating individual neighborhoods of j CHs (as in $j = 1$ case) and then, we consider all feasible combinations for a j-exchange. To evaluate the goodness of a solution generated, we employ the subproblem **SubC** and assign the most improving solution after *Maxiter* iterations as the current solution $\mathcal{S}^c$. If an improvement g_j over the best solution $\mathcal{S}^b$ (line 14) is obtained, then we update the best solution (lines 15–17). After this for loop is completed, considering all possible j values (lines 3–18), in line 19, we calculate the cumulative improvement g_h for h-exchange neighborhood search, and if it improves the highest so far improvement quantity G^*, G^* is also updated (lines 20–22). The operations on lines 23–25 aim mainly at improving the solution time in the neighborhood search. Specifically, if the highest objective improvement amount is greater than a maximum preset value G_{max} *and* the neighborhood radius ρ is greater than a preset minimum value, we decrease ρ by one unit so that, in the next maximum neighborhood size h, the generation and evaluation of neighboring solutions in each iteration i (lines 6–9) are less time consuming.

We note that, while evaluating a neighboring solution via solving **SubC** on line 8, we also employ the cut inequalities (3.26) for **(M1)**, (3.27) for **(M2)**, and (3.28)

and (3.29) for **(M3)**. In all cases, we replace $\hat{Z}$ with $Z(\mathcal{S}^b)$ and observe significant improvement in solution times. In our numerical studies, given in Section III.6, we set the values of h_{max}, initial ρ, ρ_{min}, and G_{max} as 3, 5, 2 and $0.1 * Z(\mathcal{S}^b_{init})$. In addition, we use a *Maxiter* value of H if $|\mathcal{J}| < 100$, $H/5$ if $|\mathcal{J}| > 200$, and $H/2$ otherwise.

The `SearchSubU()` Procedure

In this method, we use a solution representation based on sink selection. Finding the optimum solution of **SubU** (subproblem for fixed sinks $\mathcal{D}$) is generally a time-consuming process. Although the candidate set of sink nodes $\mathcal{K}$ is typically not very large when compared to the set of candidate CHs $\mathcal{J}$, a neighborhood search procedure based on exchanges, similar to the one described above, is still very time-consuming. On the other hand, we observe that the solution space over a set $\mathcal{K}$ can efficiently be searched by embedding a randomized construction component into an algorithm and by employing the cut inequalities while solving a **SubU** whenever needed.

More specifically, the procedure `SearchSubU()`, given in Algorithm 4, includes a fixed number of iterations, *Maxiter*, which we recommend to be set much less than the $|\mathcal{K}|$, e.g. $|\mathcal{K}|/2$ as in Section III.6. The procedure involves mainly two components. In the *first part* (lines 3–9), we generate a solution by picking the first sink location randomly and the others by ensuring a good separation between them. The approach in this first part is motivated by the observation that sinks that are significantly separated from each other will promote a divergent data flow to well-apart locations; thus, utilizing a spread set of sinks leads to more uniform energy depletion in the network. In the *second part* (lines 10–14), we assign the solution obtained from the first part as the current solution, and solve the associated subproblem **SubU** by also incorporating the cut inequality corresponding to the particular model being solved, specifically, (3.26) for **(M1)**, (3.27) for **(M2)**, and (3.28) and (3.29) for **(M3)**, with

a $\hat{Z}$ value equivalent to $Z(\mathcal{S}^b)$. While solving the subproblem, again to alleviate the problem of excessive runtimes, we employ a stopping criterion given by a `EpGap` optimality gap or `TiLim` time limit, whichever is reached first (EpGap and TiLim are CPLEX parameters). The best solution so far, $\mathcal{S}^b$, is updated if improved by the current solution $\mathcal{S}^c$, and a new iteration is started. Once the iterations are completed, we attempt to improve the final solution again by solving the subproblem **SubU** with the best cut inequality; however, this time we consider `TiLim` only as the stopping criterion, i.e., `EpGap` is the default CPLEX value of 10^{-6}. Although this procedure is simple, it is very effective in terms of solution quality and serves the purpose of exploring the solution space efficiently with inexpensive computational times. This is especially true when it is embedded in the overall procedure that we describe next.

Algorithm 4 Procedure `SearchSubU()`

initialize $\mathcal{S}^b$ (the best available solution),
  $Maxiter$, $\mathcal{D} = \emptyset$, $u = 1$;
while $Maxiter > 0$ **do**
  Randomly pick a node k from $\mathcal{K}$;
  $\mathcal{D} = \mathcal{D} \bigcup \{k\}$, $\bar{\mathcal{D}} = \mathcal{K} \setminus \{k\}$;
  while $u < U$ **do**
    $k^* = \arg\max\{d(k, \mathcal{D}) : k \in \bar{\mathcal{D}}\}$;
    $\mathcal{D} = \mathcal{D} \bigcup \{k^*\}$, $\bar{\mathcal{D}} = \bar{\mathcal{D}} \setminus \{k^*\}$;
    $u = u + 1$;
  end while
  $\mathcal{S}^c = \mathcal{D}$, solve (**SubU**) with θ, τ, and
  cut with $\hat{Z} = Z(\mathcal{S}^b)$ to obtain $Z(\mathcal{S}^c)$;
  if $Z(\mathcal{S}^c) < Z(\mathcal{S}^b)$ **then**
    $\mathcal{S}^b = \mathcal{S}^c$, $Z(\mathcal{S}^b) = Z(\mathcal{S}^c)$;
  end if
  $Maxiter = Maxiter - 1$;
end while
Solve (**SubU**) with τ and cut with $\hat{Z} = Z(\mathcal{S}^b)$;
Return $\mathcal{S}^b$ and $Z(\mathcal{S}^b)$.

III.5.3. The Complete Procedure

Our complete procedure brings together the above components including construction and improvement heuristics. We apply the complete procedure to solve **(M1)**, **(M2)**, and **(M3)** by varying only the construction heuristic component, denoted by `ConstructM·()`, in Algorithm 5.

Algorithm 5 Complete Procedure

1: **initialize** Algorithmic parameters for `ConstructM·()`, `SearchSubC()` , and `SearchSubU()`;
2: Generate an initial solution using `ConstructM·()`
3: Generate $Q-1$ CH based initial solutions randomly;
4: Improve each of Q solutions using `SearchSubC()`;
5: Assign the best of Q solutions as the $\mathcal{S}^b$;
6: Determine the $\mathcal{S}^b_{\text{free}}$ associated with $\mathcal{S}^b$;
7: Apply `SearchSubC()` to $\mathcal{S}^b$ and record $Z(\mathcal{S}^b)$;
8: Apply `SearchSubU()` using $\mathcal{S}^b$ as the input solution.
9: Return $\mathcal{S}^b$ and $Z(\mathcal{S}^b)$

In particular, to obtain diversification in searching the solution space, we initially generate Q solutions where each solution is represented by a set of CHs, $\mathcal{C}$ (lines 2-3). One of these Q solutions is found by using the appropriate construction heuristic and the others are generated randomly. Note that both `ConstructM2()` and `ConstructM1M3()` already employ the $\mathcal{C}$ set to represent a solution. We then apply the neighborhood search procedure `SearchSubC()` to each of these solutions independently, assuming $\mathcal{S}^b_{\text{free}} = \mathcal{S}^b$ (line 4). That is, we perform an intensified search in the exchange neighborhood of each of Q solutions. Upon completion, we identify the common, if any, CHs that appear in all of the Q solutions obtained; these CHs are

clearly favored in each solution, thus, we choose to keep them in the final solution. We also identify the best solution among the Q solutions, assign it as the $\mathcal{S}^b$ (line 5), and form the set $\mathcal{S}^b_{\text{free}}$ as the CHs in $\mathcal{S}^b$ that are not common in all of the Q solutions (line 6). As mentioned in Section III.5.2, subset $\mathcal{S}^b_{\text{free}}$ represents the exchangeable CHs in a solution $\mathcal{S}^b$. We treat the CHs that are common to all of the Q solutions as a preferable (non-exchangeable) CHs, and, thus, we do not engage them in the improvement procedure `SearchSubC()`. We then apply the `SearchSubC()` with initial solution $\mathcal{S}^b$ and record the objective value $Z(\mathcal{S}^b)$ (line 7). In the last stage (line 8), we utilize the procedure `SearchSubU()` with $\mathcal{S}^b$ used in the initialization step.

III.6. Computational Study

The objective of our computational studies in this section is twofold. First, we consider a single-period setting and evaluate the performance of our algorithms on the basis of solution quality and time via utilizing exact solutions for benchmarking. Secondly, now utilizing our algorithms, we evaluate the effectiveness of the three policies in prolonging the network lifetime in a multi-period setting. To this end, we compare **(M3)** with both our benchmark models **(M1)** and **(M2)** as well as with HEED Younis and Fahmy (2004).

Note that, unless stated otherwise, all of the input and algorithmic parameter values are set as mentioned in the previous sections, specifically, the input parameters as in Section III.3 and the algorithmic parameters as in Section III.5. Additional instance-based parameter values are given in the results tables that follow.

III.6.1. Performance of Heuristics

Table 3 summarizes the results for all three models where we assume that $|\mathcal{K}|$ is 8 and the U is 2 for settings with $|\mathcal{I}|$ less than 100 and 3 for the ones with $|\mathcal{I}|$ values greater than or equal to 100. We also assume that the N is 50 for all the settings, except the ones with $|\mathcal{I}|$ equal to 150 for which N is 75. Furthermore, we set the data generation rate R_i to 10 for settings with $|\mathcal{I}|$ less than 75, and to 9, 8, and 7 for the ones with $|\mathcal{I}|$ values of 75, 100, and 150, respectively. For each problem setting, where we have 15 different instance sets for **(M1)**, 18 for **(M2)**, and 12 for **(M3)** obtained by varying the values of $|\mathcal{I}|$ and the number of required CHs H, we solve 30 randomly generated instances.

The values reported in columns 4 to 7 of the table are averaged over these ten instances. The fourth (T^{O}_{ave}) and the fifth (T^{O-C}_{ave}) columns concern the exact solution times (in seconds) for these instances. In the former, we report the average solution times for optimality (ns represents the instances are not solved in 20 hours of runtime), whereas in the latter, we report the same measure, however, this time incorporating the complete heuristic solution value into the formulation as a cut inequality as described in (3.26) for **(M1)**, (3.27) for **(M2)**, and (3.28) and (3.29) for **(M3)**. Clearly, for an instance, the optimum objective function values, Z^{O}, for both of these cases are the same. However, the solution times are significantly improved, as observed in column for T^{O-C}_{ave}, if the heuristic solution (Algorithm 5) is obtained first and is utilized in solving a model to optimality. The sixth column (T^{H}_{ave}) includes the average solution times obtained using our complete heuristic given in Algorithm 5. We observe that they are very significantly lower than the solution times for exact solutions. In our tests, we consider only one additional random initial solution which provides enough diversification and notable improvements in solution quality. In terms of solution

quality, we calculate the optimality gap as $\Delta = 100 * (Z^H - Z^O)/Z^O$, where Z^H represents the heuristic objective function value for an instance. The average optimality gaps, Δ_{ave}, are reported in the last column of Table 3, and they illustrate that our heuristic approach, which amalgamates various problem and solution characteristics, is very effective in addressing a rather complex problem.

III.6.2. Network Lifetime Comparison of Models

We use our heuristic procedure to solve the associated problems in each period of a multi-period setting to examine the performance of the models in terms of network lifetime measure. For this purpose, we consider, as before, a Ψ value of 60 in each period to obtain a candidate CH set of $\mathcal{J}$ given by $\mathcal{I}^R$, and two topology control schemes (p_I, inc) as $(1.0, 0.0)$ and $(0.1, 0.1)$. In Table 4, we present our results for a number of problem settings, each given in a row, and ten randomly generated instances for each setting. The first part of the table corresponds to the case where we consider no calibration for usable energy reserve by setting p to 1.0 so that the complete energy level at each sensor node is available for usage. Similarly to the results we obtained in our preliminary analysis, where we solved each period's problem to optimality in small instances, **(M3)** performs significantly better in terms of network lifetime than **(M1)** and **(M2)** which perform particularly poorly.

Thus, we further consider the calibration of p as reported in the second part of Table 4 under $(p_I, inc) = (0.1, 0.1)$ scheme. In this case, lifetime performance with both **(M1)** and **(M2)** improves dramatically, illustrating the effectiveness of incorporating topology control via parameter p. Notably, the improvement in **(M1)** is more significant than the **(M2)**. Observe that while **(M2)** performs better than **(M1)** under $p = 1.0$, this is reversed under the $(0.1, 0.1)$ scheme. On the other hand, it is clear that even **(M1)** still cannot perform as good as **(M3)**. Although we

Table 3: Performance results for solving single period problems

Model	$\|\mathcal{I}\|$	H	T^{O}_{ave}	T^{O-C}_{ave}	T^{H}_{ave}	Δ_{ave}
(M1)	25	2	0.63	0.98	2.05	0.87
		4	2.16	2.32	2.21	0.66
		5	2.53	2.75	2.41	0.58
	50	4	22.42	24.69	13.09	0.79
		8	19.09	24.48	13.05	0.11
		10	17.78	19.00	13.96	0.21
	75	6	112.52	127.15	82.81	0.13
		12	83.90	82.63	67.67	0.30
		15	63.53	79.31	63.34	0.19
	100	8	963.07	902.92	220.51	0.50
		16	567.52	649.54	132.20	0.12
		20	377.12	472.15	122.78	0.12
	150	12	6646.53	5962.81	427.96	1.16
		24	3600.35	3227.57	245.97	0.27
		30	2620.82	3483.89	274.84	0.41
(M2)	15	2	5.10	2.80	1.98	1.09
		3	25.22	2.30	2.08	1.39
		4	58.98	2.54	1.99	1.33
	20	2	19.23	3.28	4.35	1.04
		3	123.47	5.87	4.63	1.13
		4	424.37	6.68	4.57	1.10
	25	2	87.14	15.75	9.09	0.70
		4	1845.17	22.31	9.27	0.82
		5	7698.09	33.88	9.14	0.89
	30	2	220.04	20.72	16.66	0.53
		4	8339.40	77.84	17.64	0.70
		6	ns	179.87	16.57	0.79
	40	4	ns	300.36	52.10	0.52
		6	ns	944.07	49.71	0.57
		8	ns	1241.67	49.07	0.67
	50	4	ns	1186.85	126.37	0.61
		8	ns	848.82	108.19	0.84
		10	ns	4938.61	106.98	1.11
(M3)	15	2	2.79	2.68	3.06	0.00
		3	7.53	7.15	11.77	0.14
		4	17.09	16.12	29.30	0.16
	20	2	8.36	6.72	8.81	0.21
		3	38.26	29.27	23.69	1.09
		4	125.15	145.50	72.65	0.26
	25	2	23.50	23.38	13.22	0.12
		4	755.86	82.20	70.22	0.10
		5	3105.96	1081.02	99.12	1.26
	30	2	56.02	50.87	52.16	1.40
		4	2243.28	212.30	60.01	0.22
		6	ns	8233.12	279.70	0.66

try only one scheme out of many possible ones, it is clear that a good strategy for adjusting the value of p in the course of a deployment cycle is useful for extending network lifetimes. However, there are two issues associated with such an approach. First, with changing problem size and environment depending on the application, calibrating for a good value p for **(M1)** and an inc for a (p_I, inc) scheme during a deployment cycle are very difficult and impractical. Second, even if a topology control scheme, such as $(0.1, 0.1)$, is employed, finding the feasible p in each period implies solving the same problem multiple times with varying p values. On the other hand, we observe that none of these difficulties is associated with our **(M3)** since it is very robust to the changes in p, and, thus, it truly integrates the topology control and the routing problems effectively.

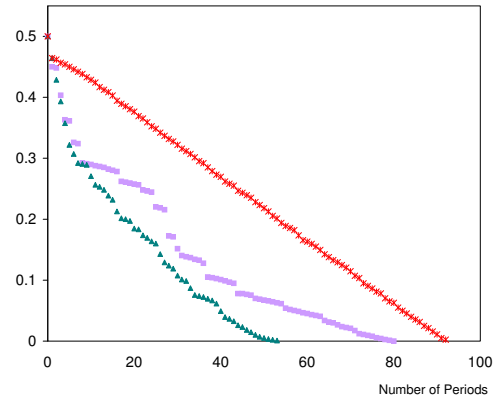
We next examine the remaining energy distributions in a relatively large problem setting both from the models' performance perspective and their impact on the redeployment strategies. For this purpose, we consider the setting given by $|\mathcal{I}| = 200$, $|\mathcal{K}|$=16, U=3, N=75, R_i=6, and H=16 values. In Figures 8(a), 8(b), and 8(c), we plot the minimum, the average, and the maximum remaining energy levels (E^R_{min}, E^R_{ave}, and E^R_{max}, respectively) at the sensor nodes after each period for each model. Furthermore, in Figures 9(a), 9(b), and 9(c), we give the remaining energy levels of the sensor nodes at the end of the deployment cycle.

In Figures 8(a) – 8(c), we observe that the variations in remaining energy levels for **(M1)** and **(M2)** are very large when compared to **(M3)**. More specifically, for **(M1)**, the E^R_{min} and E^R_{max} values are apart from each other, thus implying a large variation in remaining energy levels. This is also illustrated by the end-of-deployment-cycle energy levels, which are quite scattered for the sensors as observed in Figure 9(a). Since **(M1)** concentrates on minimizing the total energy usage, some sensors are more frequently chosen as CHs and deplete their energy more. On the other hand,

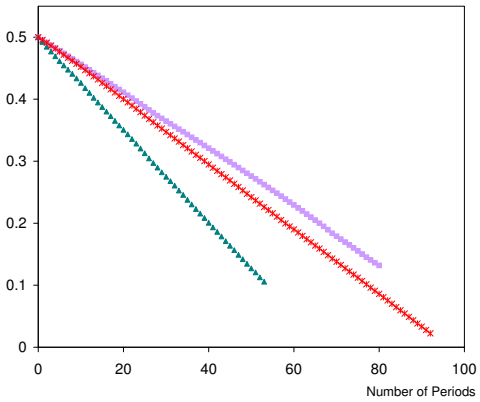
Table 4: Comparison of different objective in the multi-period based on heuristic method

Problem Setting	H	$p = 1.0$ (M1)	$p = 1.0$ (M2)	$p = 1.0$ (M3)	$p = 0.1+$ (M1)	$p = 0.1+$ (M2)
$\|\mathcal{I}\| = 25$, $\|\mathcal{K}\|$=8, U=2, N=50, R_i=10	2	9	13	**61**	60	49
	4	11	22	**66**	59	57
	5	12	26	**69**	63	51
$\|\mathcal{I}\| = 50$, $\|\mathcal{K}\|$=8, U=2, N=50, R_i=10	4	7	12	**62**	50	53
	8	9	20	**67**	58	55
	10	11	25	**69**	51	54
$\|\mathcal{I}\| = 75$, $\|\mathcal{K}\|$=8, U=2, N=50, R_i=9	6	5	13	**71**	59	51
	12	12	21	**76**	62	59
	15	14	26	**78**	54	55
$\|\mathcal{I}\| = 100$, $\|\mathcal{K}\|$=16, U=3, N=50, R_i=8	8	20	15	**87**	77	58
	16	15	25	**93**	81	63
	20	13	31	**93**	67	65
$\|\mathcal{I}\| = 150$, $\|\mathcal{K}\|$=16, U=3, N=75, R_i=7	12	10	14	**78**	63	44
	24	12	25	**85**	70	52
	30	18	30	**87**	56	58
$\|\mathcal{I}\| = 200$, $\|\mathcal{K}\|$=16, U=3, N=75, R_i=6	16	13	16	**92**	80	53
	32	10	28	**100**	78	58
	40	22	34	**102**	78	65
$\|\mathcal{I}\| = 250$, $\|\mathcal{K}\|$=16, U=3, N=100, R_i=5	20	7	16	**92**	62	43
	40	9	28	**100**	80	57
	50	12	36	**103**	74	64
$\|\mathcal{I}\| = 300$, $\|\mathcal{K}\|$=16, U=3, N=100, R_i=4	24	11	19	**115**	95	54
	48	24	36	**126**	91	68
	60	20	44	**130**	87	61

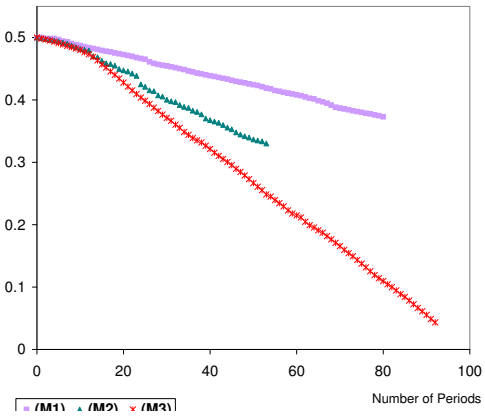
Figure 8: Remaining Energy Characteristics – Progression



(a) Minimum Remaining Energy (E^R_{min})

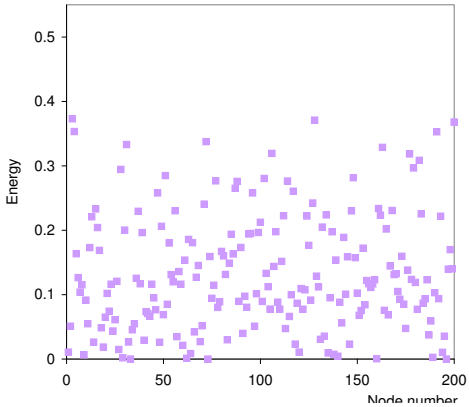


(b) Average Remaining Energy (E^R_{ave})

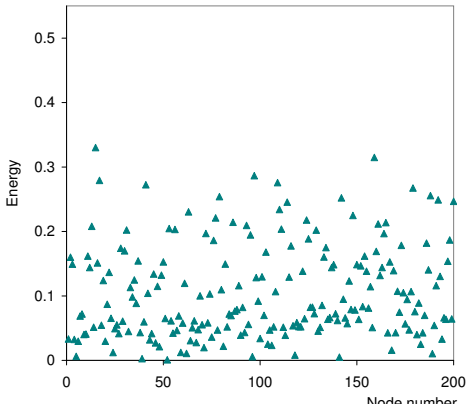


(c) Maximum Remaining Energy (E^R_{max})

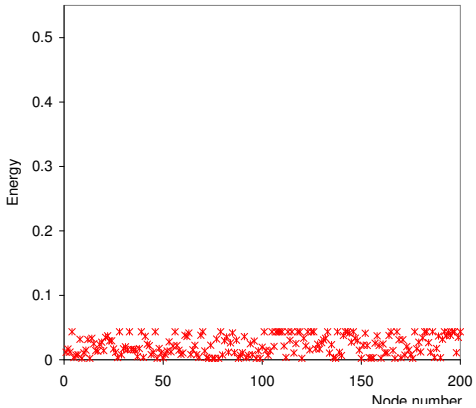
Figure 9: Remaining Energy Distributions – End-of-Deployment-Cycle Snapshot



(a) **(M1)**



(b) **(M2)**



(c) **(M3)**

since **(M2)** does not address any network-wide energy usage measures, the objective of minimizing the maximum energy usage implies one or a few critical sensors whose energy usage is determinant of the overall performance. The usage level determined for the critical sensor(s) also dictates the energy usage allowance (upper bound) for other sensors, and the relative closeness of the end-of-period E^R_{ave} and E^R_{min} values in Figures 8(a) and 8(b) show that the sensors dissipate their energy at similar rates. This is also observed in Figure 9(b) where most of the sensors have remaining energy levels, although somewhat scattered, more clustered close to zero. In both **(M1)** and **(M2)**, the energy depletion rate at the sensors is also implicitly determined by the $(p_I, inc) = (0.1, 0.1)$ scheme. We note that the use of more stringent schemes generally causes infeasibilities, and, thus, the p is incremented early in the deployment cycle; and an exact calibration of p is very difficult. Coupled with the results given in Table 4, it is clear that the energy reserve scheme is the main reason we obtain relatively better lifetime measures with **(M1)** and **(M2)**. In Figures 8(a) and 8(c), we clearly observe that **(M3)** always has the highest E^R_{min} and the lowest E^R_{max}. Furthermore, these values are very close to each other, presenting a narrow range (a small variation) in the remaining energy levels, which implies that most of the sensors deplete their energy more or less at the same rate during the progression of the deployment cycle. This is also clearly reflected in Figure 9(c) in which the remaining energy levels form a very narrow band. Since the minimization of total energy usage is also considered in **(M3)**, the depletion rate is slow, providing a good network lifetime measure without relying on an explicit control of the usable energy reserve scheme. This is not unexpected, since we formulate the objective function of **(M3)** to incorporate this characteristic as well.

We finally note that, from a redeployment strategy perspective, **(M3)** is appealing since it contains primarily low energy sensors at the end of the deployment cycle.

This can facilitate a random deployment in the beginning of each cycle, so as to start with similar initial settings. On the other hand, **(M1)** and **(M2)** require special deployment and/or further attention to topology control in successive deployment cycles due to a relatively high overall variation in the end-of-deployment-cycle energy levels.

III.6.3. Network Lifetime Comparison of M3 and HEED

To further evaluate the performance of our proposed model, we compare model **(M3)** with HEED (Hybrid Energy-Efficient Distributed clustering) Younis and Fahmy (2004) which is a well-known method for its performance in terms of network lifetime. As mentioned in Section II.3.1, HEED's main goal is to identify CHs and assign sensors to clusters for better energy efficiency. The CHs are probabilistically selected based on their remaining energy and the sensors join clusters so as to minimize the communication cost. HEED does not specify a particular scheme for routing CHs to the sink after the clusters and CHs are determined; however, the authors specifically mention routing to achieve minimum power usage across the network as a possible approach among others. For detailed description on HEED protocol, we refer the reader to Younis and Fahmy (2004).

We choose HEED for comparison because it has the following features similar to the setting we consider: (1) It is a cluster-based routing protocol for data gathering purpose; HEED assumes a multi-hop connection between CHs and to the sink and, at each period, it selects CHs with high remaining energy; (2) Data collection is performed periodically which is suitable for continuous monitoring; (3) Data aggregation is performed at each cluster-head for energy efficiency.

On the other hand, there are also some differences in the model assumptions of HEED and **(M3)** : (1) In HEED, only one sink node is considered and each sensor

can only choose one CH to transmit its data; (2) The cluster radius is explicitly specified, i.e., each sensor can only transmit its data to a CH within the specified range; (3) The data is aggregated into a single representation signal at each CH, i.e., regardless of the amount of data received, a CH transmits a fixed amount of data out towards the sink, thus the sink eventually receives this fixed amount of data from the network (e.g., maximum temperature in the sensor field); (4) HEED adopts the radio model where both the free space (typical D^2 power loss) and the multipath fading (D^4 power loss) models are used. To transmit x_{ij} ($bits$) of data from node i to node j dissipates $f(D_{ij})\, x_{ij}$, where $f(D_{ij})$ is defined as

$$f(D_{ij}) = \begin{cases} w + v_1 D_{ij}^2 & \text{if } D_{ij} < d_0 \\ \\ w + v_2 D_{ij}^4 & \text{if } D_{ij} \geq d_0 \end{cases} \tag{3.30}$$

with parameters set as $d_0 = 75m$, $v_1 = 10\ pJ/bit/m^2$, $v_2 = 0.0013\ pJ/bit/m^4$, and $w = 50\ nJ/bit$.

In the **(M3)** model, we consider multiple sinks and a general aggregation scheme in which, instead of a single representative data for the sensor field, a more general view of a measure is of interest, e.g., temperature/humidiy/pressure gradients in a large sensor field employed for environmental monitoring. More importantly, **(M3)** also integrates selection of CHs and sinks with routing decisions. Therefore, for comparison purposes, we modify model **(M3)** to handle the several characteristics of the setting in HEED by introducing cluster radius, single sink, and the new distance representation as follows.

First, we introduce two new parameters into **(M3)**: r as the cluster radius and k as the total amount of data per period generated at each sensor. Furthermore, we redefine the variable x_{ij}^c as a binary variable with a value of 1 if a sensor i is assigned

to a CH j, and 0 otherwise. We also note that since there is a single sink we use the variable x^u_{m0}, where index 0 represents the sink, for flow from CH j to the sink. Then, the modified formulation, called **(eM3)**, is ($i \in \mathcal{I}$, $j, m \in \mathcal{J}$ unless stated otherwise)

$$\text{Min} \qquad t(1/|\mathcal{I}|)(\sum_{m\in\mathcal{J}} e^c_m + \sum_{i\in\mathcal{I}} e_i) + (E^R_{max} - E^R_{min}) \tag{3.31}$$

subject to

$$f(D_{m0})\, k\, x^u_{m0} + \sum_{j\in\mathcal{J}\setminus\{m\}} f(D_{mj})\, k\, x^{cc}_{mj} + \sum_{j\in\mathcal{J}\setminus\{m\}} w\, k\, x^{cc}_{jm} + (w + c\, s)\, k \sum_{i\in\mathcal{I}} x^c_{im} = e^c_m \qquad \forall\, m \tag{3.32}$$

$$\sum_{j\in\mathcal{J}} f(D_{ij})\, k\, x^c_{ij} = e_i \qquad \forall\, i \tag{3.33}$$

$$x^u_{m0} + \sum_{j\in\mathcal{J}\setminus\{m\}} x^{cc}_{mj} - (\sum_{j\in\mathcal{J}\setminus\{m\}} x^{cc}_{jm} + (1-s) \sum_{i\in\mathcal{I}} k\, x^c_{im}) = 0 \qquad \forall\, m \tag{3.34}$$

$$\sum_{j\in\mathcal{J}} x^c_{ij} = 1 \qquad \forall\, i \tag{3.35}$$

$$x^c_{ij} \le \lfloor r/D_{ij} \rfloor z^c_j \qquad \forall i, \forall j \tag{3.36}$$

$$x^{cc}_{mj} \le k\, |\mathcal{I}|\, z^c_j \qquad \forall m, j \tag{3.37}$$

$$x^c_{j0} \le k\, |\mathcal{I}| z^c_j \qquad \forall j \tag{3.38}$$

$$e_i \le p\, E_i \qquad \forall\, i \tag{3.39}$$

$$e^c_j \le p\, E_j \qquad \forall\, j \tag{3.40}$$

$$\sum_{j\in\mathcal{J}} z^c_j = H \tag{3.41}$$

$$z_j^c\, E_j - e_j^c \;\leq\; E_{max}^R \qquad \forall\, j \qquad (3.42)$$

$$(1 - z_i^c)\, E_i - e_i \;\leq\; E_{max}^R \qquad \forall\, i \qquad (3.43)$$

$$E_{min}^R \leq E_i - e_i \qquad \forall\, i \qquad (3.44)$$

$$E_{min}^R \leq E_j - e_j^c \qquad \forall\, j \qquad (3.45)$$

$$x_{ij}^c,\; z_j^c, \in \{0, 1\} \qquad \forall\, i,\; j \qquad (3.46)$$

$$x_{ij}^{cc},\; x_{j0}^u,\; e_i,\; e_j^c,\; E_{max}^R,\; E_{min}^R \geq 0 \qquad \forall\, i,\; j \qquad (3.47)$$

Constraints (3.32) and (3.33) assign the values of the total energy consumed by a CH and a sensor node, respectively. Constraints (3.34) state the data flow balance at each CH node and constraint (3.35) guarantees that each sensor is assigned to one CH. Constraints (3.36)–(3.38) assign the values of binary variables related to CH location selections. Note that constraints (3.36) also ensure that only the sensors within the cluster radius r can transmit the data to the associated CH. Constraint sets (3.39) and (3.40) ensure that the total energy consumed at a node cannot exceed the total available energy at the corresponding sensors. Constraints (3.41) establish the required number of CHs. Constraint sets (3.42)–(3.45) give the maximum and minimum remaining energy at a sensor node. Finally, (3.46) and (3.47) include the integrality and non-negativity of the decision variables.

Our solution approach in Section III.5 is directly applicable to solve the modified model as well. However, since only one sink is employed in **eM3**, steps 7 and 8 in the Algorithm 5 are excluded.

As mentioned above, the main component of the HEED is the selection of CHs and the assignment of sensors to CHs i.e., forming the clusters); no specific routing protocol to compute inter-cluster paths between CHs or to the sink is given in HEED.

Thus, as suggested in the context of HEED, we minimize energy usage by employing a subproblem of our **(M1)** model (which minimizes average energy usage in the network) while determining data routing from sensors to the sink via CHs. In doing so, we readily incorporate the same general aggregation in **(M3)** in the routing scheme since **(M1)** employs the same aggregation approach. In each period, we obtain the formation of clusters, i.e., the values of x_{ij}^c and z_j^c using HEED algorithm, then we solve the model **(M1)** by fixing the binary variables x_{ij}^c and z_j^c to these values to obtain data routing given by variables x_{ij}^{cc} and x_{j0}^u.

For our studies, similar to the setting in HEED, we assume the nodes are randomly distributed in square of size 100m sensor field (with its lower left one corner at the origin) and the sink node is located at coordinates (50,175). The cluster radius r is set 25m and the amount of sensor data generated per period, k, is taken as 2000 bits. The aggregation ratio, s, values of 0.15, 0.20, 0.25, and 0.30; and the number of CHs, H, of 9, 10, 13, and 15 are used for sensor fields with the number of sensors, $\mathcal{I}$, of 150, 200, 250, and 300, respectively. In doing so, we ensure that the number of CHs is about 5% of the number of sensors, similar to HEED, and the aggregation ratio increases (i.e., amount of data eliminated increases) as the sensor density in the field increases. Finally, as before, the initial energy at the sensors are randomly drawn from U[0.1,0.5].

We summarize the average lifetime results over our instances in Table 5. It can easily be observed that the integrated approach, which we devise via integrating topology and routing decisions, significantly performs better than HEED; the network lifetime is about six-fold longer on average with the integrated approach **eM3**.

Table 5: Network lifetime via HEED and (eM3) approaches

	Number of nodes, $\|\mathcal{I}\|$			
Approach	150	200	250	300
HEED	42	50	55	50
eM3	264	283	294	302

III.7. Summary and Conclusions

In this chapter, we introduce three alternative mathematical models for integrated topology and routing decisions for data-gathering WSNs so as to prolong their lifetime subject to limited energy at the sensors. In doing so, we also consider a hierarchical network structure with multi-hop routing, multiple sinks, and a general data aggregation approach and devise the models to determine clusterhead and active sink locations as well as data flow routes from sensors to sinks in each period. We also describe a topology control scheme on how the usable energy fraction (of the available energy) at a sensor changes from period to period in a deployment cycle.

The first two approaches, **(M1)** and **(M2)**, which have the objectives of average energy usage minimization and minimization of the maximum energy usage at a sensor, are considered previously in the literature while prescribing communication protocols as opposed to an integrated mathematical modelling perspective as in this study. Thus, these models can be considered as benchmark models for the performance of the proposed third model, **(M3)**, which has an objective of the minimizing the total energy *and* the range of remaining energy distribution at the sensors.

Since the models dictate large discrete optimization formulations, employing exact optimization approaches is highly impractical, thus, we develop a general heuristic algorithm, applicable for each model, that performs very well in our computa-

tional tests. Our procedure utilizes efficient construction heuristics, two types of solution representations, a combination of multiple neighborhoods, and an objective value based on cut inequalities for better efficiency in evaluating candidate solutions. Heuristic solutions are also employed via cut inequalities to improve the time performance and alleviate the memory difficulties associated with an exact branch-and-cut implementation.

Using our heuristic approach, we conduct numerical tests and analyses of the models in a multi-period setting. We observe that **(M2)** and especially **(M1)** perform very poorly when usable energy fraction p is set to 1.0 during a deployment cycle. When the usable energy scheme is changed to a conservative one given by $(p_I, inc) = (0.1, 0.1)$, i.e. initial p is 0.1 and increased by 0.1 whenever energy levels are insufficient in a period, both **(M1)** (which now performs better than **(M2)**) and **(M2)** exhibit better network lifetime performance; however, they still perform inferior to **(M3)** and appear to be highly sensitive to the topology control scheme.

Our proposed model **(M3)** is highly insensitive to the setting of usable energy fraction p and performs very well in terms of network lifetime. It incorporates both energy usage and variation in the end-of-period remaining energy levels in its objective, and, thus, truly integrates topology control and routing decisions without the need to exogenously set and calibrate a p value. In our numerical tests, we also observe that, even when the first period energy levels in a deployment cycle are varied, **(M3)** performs very well with a full available energy (i.e. $p = 1.0$). This property also contributes to efficiencies in terms of redeployment since the reconfiguration via topology and routing decisions in each period leads to a self-adjustment in the network. As a result, the energy levels at the end of a deployment cycle are confined in a narrow band which implies that uniform deployment strategies can be effectively employed.

Finally, we also compare the performance of the proposed **(M3)** model to a well-known protocol HEED devised specifically to determine CH locations and sensor-to-CH assignments in a WSN. For data routing in HEED, we employ a mathematical optimization model based on our models; and we also slightly modify **(M3)** to obtain a special case, which is still solvable by our algorithm, so that the two approaches can be compared. In our computational studies, we show that our proposed model performs significantly better in this comparison as well.

In summary, our modelling approaches, solution algorithms, and extensive analyses illustrate that **(M3)** has attractive properties capturing important characteristics of integrated topology and routing decisions to improve energy efficiency and prolong lifetime of data-gathering WSNs.

CHAPTER IV

SENSOR NETWORK DESIGN/ROUTING PROBLEM WITH FIXED CLUSTER-HEAD SET-UP COST

In this chapter, we consider an important extension model **(M3-E1)** to the setting of **(M3)** by incorporating the fixed cost associated with locating the Cluster-heads (CHs) into the objective function. In our proposed model, we consider a hierarchical setting where data flow from sensor nodes to the sink nodes occurs via CHs. A CH not only functions to capture information in its vicinity, but also functions as an aggregator/relay node to process and transfer the data generated by other sensors to the sinks. It is well known that CHs consume more energy than regular sensors. By setting a higher fixed cost with lower energy nodes, **(M3-E1)** attempts to avoid some well-positioned sensors, being selected as CHs repeatedly in successive periods and to protect low-energy sensors from quick energy depletion. This will further balance the energy dissipation among the nodes.

Since our mathematical model dictates discrete optimization formulation, even small size instances are highly impractical to be solved using the commercial Branch-and-cut software such as CPLEX. On the other hand, we note that the formulation of **(M3-E1)** is amenable to the efficient Benders decomposition (BD) method. Thus, we develop a BD solution approach that incorporates a simple heuristic algorithm, strengthened Benders cuts and an ε-optimal approach. Computational evidence demonstrates the efficient performance of the BD approach in terms of solution quality and time, especially for large-size instances. In particular, our heuristic algorithm provides good initial upper bounds and facilitates the generation of initial Benders cuts; strengthen Benders cuts and ε-optimal approach accelerate the convergence of classical BD algorithm.

IV.1. The Model

We adopt the same problem setting and model notations as presented in section III. The mathematical formulation for **(M3-E1)** is as follows:

$$\text{Min} \quad t_1(1/|\mathcal{I}|)(\sum_{m\in\mathcal{J}} e^c_m + \sum_{i\in\mathcal{I}} e_i) + (E^R_{max} - E^R_{min}) + t_2(\sum_{m\in\mathcal{J}} z^c_m/E_m) \tag{4.1}$$

subject to

$$\sum_{k\in\mathcal{K}}(w + v\,D^2_{mk})\,T\,x^u_{mk} + \sum_{j\in\mathcal{J}\setminus\{m\}} (w + v\,D^2_{mj})\,T\,x^{cc}_{mj} + \sum_{j\in\mathcal{J}\setminus\{m\}} w\,T\,x^{cc}_{jm} + \sum_{i\in\mathcal{I}}(w + c\,s)\,R_i\,T\,x^c_{im} = e^c_m \;\; \forall\, m \in \mathcal{J} \tag{4.2}$$

$$\sum_{j\in\mathcal{J}}(w + v\,D^2_{ij})\;R_i\,T x^c_{ij} = e_i \qquad \forall\, i \in \mathcal{I} \tag{4.3}$$

$$\sum_{k\in\mathcal{K}} x^u_{mk} + \sum_{j\in\mathcal{J}\setminus\{m\}} x^{cc}_{mj} - \left(\sum_{j\in\mathcal{J}\setminus\{m\}} x^{cc}_{jm} + (1-s)\sum_{i\in\mathcal{I}} R_i\,x^c_{im}\right) = 0\;\forall\, m \in \mathcal{J} \tag{4.4}$$

$$\sum_{j\in\mathcal{J}} x^c_{ij} = 1 \qquad \forall\, i \in \mathcal{I} \tag{4.5}$$

$$x^c_{ij} \le z^c_j \qquad \forall i \in \mathcal{I}, j \in \mathcal{J} \tag{4.6}$$

$$x^{cc}_{mj} \le \sum_{i\in\mathcal{I}} R_i z^c_j \qquad \forall m, j \in \mathcal{J} \tag{4.7}$$

$$x^u_{jk} \le \sum_{i\in\mathcal{I}} R_i z^u_k \qquad \forall j \in \mathcal{J}, k \in \mathcal{K} \tag{4.8}$$

$$x^c_{jk} \le \sum_{i\in\mathcal{I}} R_i z^c_j \qquad \forall j \in \mathcal{J}, k \in \mathcal{K} \tag{4.9}$$

$$\sum_{j\in\mathcal{J}} z^c_j = H \tag{4.10}$$

$$\sum_{k\in\mathcal{K}} z^u_k = U \tag{4.11}$$

$$e_i \leq p\, E_i \qquad \forall i \in \mathcal{I} \tag{4.12}$$

$$e_j^c \leq p\, E_j \qquad \forall j \in \mathcal{J} \tag{4.13}$$

$$z_j^c\, E_j - e_j^c \;\leq\; E_{max}^R \qquad \forall j \in \mathcal{J} \tag{4.14}$$

$$(1 - z_i^c)\, E_i - e_i \;\leq\; E_{max}^R \qquad \forall i \in \mathcal{I} \tag{4.15}$$

$$E_{min}^R \leq E_i - e_i \qquad \forall i \in \mathcal{I} \tag{4.16}$$

$$E_{min}^R \leq E_j - e_j^c \qquad \forall j \in \mathcal{J} \tag{4.17}$$

$$z_j^c,\, z_k^u \in \{0, 1\} \qquad \forall\, i \in \mathcal{I},\;\; j \in \mathcal{J},\;\; k \in \mathcal{K} \tag{4.18}$$

$$x_{ij}^c,\, x_{ij}^{cc},\, x_{jk}^u,\, e_i,\, e_j^c,\, E_{max}^R,\, E_{min}^R \geq 0 \qquad \forall\, i \in \mathcal{I},\;\; j \in \mathcal{J},\;\; k \in \mathcal{K} \tag{4.19}$$

The first term in the objection function represents the weighted (where t_1 is the weight) sum of average energy consumption. The second term gives the range of remaining energy levels and the last term represents the fixed cost associated with locating the CHs. Constraints (4.2) and (4.3) assign the values of the total energy consumed by a CH and a sensor node, respectively. Constraints (4.4) state the data flow balance at each CH node and constraint (4.5) guarantees that each sensor is assigned to one CH. Constraints (4.6)–(4.9) assign the values of binary variables related to CH and sink node location selections. Constraints (4.10) and (4.11) establish the required number of CHs and sink nodes, respectively. Constraint sets (4.12) and (4.13) ensure that the total energy consumed at a node cannot exceed the total available energy at the corresponding sensors. Constraint sets (4.14) and (4.15) give the maximum remaining energy at a sensor node and constraint sets (4.16) and (4.17) give the minimum remaining energy at a sensor node. Finally, (4.18) and (4.19) include the integrality and non-negativity of the decision variables.

IV.2. Benders Decomposition Based Solution Approach

Benders decomposition (Benders, 1962) is a classical solution approach for combinatorial optimization problems and it has been successfully solving a wide array of large-scale mathematical formulations. This technique is based on the idea of exploiting the special structure of the problem so that it can partition the original formulation into two smaller problems, denoted as a *master problem* and a *subproblem*, respectively. The master problem is obtained by removing a number of constraints of the original model and it is expected to provide the optimal solution after the addition of a sequence of cuts, denoted as *Benders cuts*. The master problem accounts for all the integer variables of the original problem and one additional (continuous) auxiliary variable associated with the Benders cuts. On the other hand, the subproblem includes all continuous variables and the associated constraints, and the Benders cuts are derived from the solutions of the dual subproblem. In each iteration, the master problem is resolved to optimality with the addition of a Benders cut. It gives a lower bound for the original problem and a set of values for the integer variables that are then substituted into the subproblem. Next, the dual subproblem is solved to obtain an upper bound and a set of dual variables values that are used to generate a new Benders cut for the master problem in the next iteration. This process is repeated until a termination condition, usually a small optimality gap between the lower bound and the upper bound, is met.

In contrast to the heuristic methods that only give feasible solutions and can not guarantee the solution quality , BD approach provides both lower and upper bounds. At each iteration, the master problem and the subproblem are solved to obtain lower and upper bounds on the objective value of the original problem. Since each iteration of the algorithm adds a new Benders cut to the master problem, the lower bound is

therefore non-decreasing.

Our formulation in **(M3-E1)** employs the binary variables z_j^c and z_k^u (z for brevity) associated with CH and sink nodes selection, continuous variables x_{ij}^c, x_{ij}^{cc} and x_{jk}^u (x for brevity) for routing decisions and energy characteristics related variables e_i, e_m^c , E_{max}^R and E_{min}^R (e for brevity). The structure of our problem presents a natural decomposition scheme for the Benders approach: the routing problem (for fixed z variables) is a linear program which can be solved efficiently, and the master problem (excluding routing decisions) is an integer program involving much smaller numbers of variables and constraints which is easier to solve. Therefore, at each iteration, the solution of the master problem gives a tentative network configuration (the selection of CH and sink locations) for which the subproblem finds the optimal data routing with the given network topology.

In the section that follows, we provide detailed description of each component of BD framework.

IV.2.1. Benders Subproblem and Its Dual

For given binary variables $\hat{z}$ associated with fixed CH and sink nodes locations, the subproblem $SP(x, e|\hat{z})$ is essentially a minimization problem that determines the data routing scheme from sensors to the sinks via CHs. The subproblem $SP(x, e|\hat{z})$ can be stated as follows:

$$\text{Min} \qquad Z_{\text{SP}} = t_1(1/|\mathcal{I}|)(\sum_{m\in\mathcal{J}} e_m^c + \sum_{i\in\mathcal{I}} e_i) + E_{max}^R - E_{min}^R \tag{4.20}$$

$$\sum_{k\in\mathcal{K}}(w+v\,D_{mk}^2)\,T\,x_{mk}^u+\sum_{j\in\mathcal{J}\setminus\{m\}}(w+v\,D_{mj}^2)\,T\,x_{mj}^{cc}$$
$$+\sum_{j\in\mathcal{J}\setminus\{m\}}w\,T\,x_{jm}^{cc}+\sum_{i\in\mathcal{I}}(w+c\,s)\,R_i\,T\,x_{im}^c=e_m^c \qquad \forall\, m\in\mathcal{J} \tag{4.21}$$

$$\sum_{j\in\mathcal{J}}(w+v\,D_{ij}^2)\;R_i\,T x_{ij}^c=e_i \qquad \forall\, i\in\mathcal{I} \tag{4.22}$$

$$\sum_{k\in\mathcal{K}}x_{mk}^u+\sum_{j\in\mathcal{J}\setminus\{m\}}x_{mj}^{cc}-\left(\sum_{j\in\mathcal{J}\setminus\{m\}}x_{jm}^{cc}+(1-s)\sum_{i\in\mathcal{I}}R_i\,x_{im}^c\right)=0\forall\, m\in\mathcal{J} \tag{4.23}$$

$$\sum_{j\in\mathcal{J}}x_{ij}^c=1 \qquad \forall\, i\in\mathcal{I} \tag{4.24}$$

$$x_{ij}^c\le\hat{z_j^c} \qquad \forall i\in\mathcal{I},\forall j\in\mathcal{J} \tag{4.25}$$

$$x_{mj}^{cc}\le\sum_{i\in\mathcal{I}}R_i\hat{z_j^c} \qquad \forall m,j\;\in\mathcal{J} \tag{4.26}$$

$$x_{jk}^u\le\sum_{i\in\mathcal{I}}R_i\hat{z_k^u} \qquad \forall j\in\mathcal{J},\forall k\in\mathcal{K} \tag{4.27}$$

$$x_{jk}^c\le\sum_{i\in\mathcal{I}}R_i\hat{z_j^c} \qquad \forall j\in\mathcal{J},\forall k\in\mathcal{K} \tag{4.28}$$

$$e_i\le p\,E_i \qquad \forall\, i\in\mathcal{I} \tag{4.29}$$

$$e_j^c\le p\,E_j \qquad \forall\, j\in\mathcal{J} \tag{4.30}$$

$$E_{max}^R+e_j^c\;\ge\;\hat{z_j^c}\,E_j \qquad \forall\, j\in\mathcal{J} \tag{4.31}$$

$$E_{max}^R+e_i\;\ge\;(1-\hat{z_i^c})\,E_i \qquad \forall\, i\in\mathcal{I} \tag{4.32}$$

$$E_{min}^R\le E_i-e_i \qquad \forall\, i\in\mathcal{I} \tag{4.33}$$

$$E_{min}^R\le E_j-e_j^c \qquad \forall\, j\in\mathcal{J} \tag{4.34}$$

$$x_{ij}^c,\;x_{ij}^{cc},\;x_{jk}^u,\;e_i,\;e_j^c,\;E_{max}^R,\;E_{min}^R\ge 0 \qquad \forall\, i\in\mathcal{I},\;\;j\in\mathcal{J},\;\;k\in\mathcal{K} \tag{4.35}$$

In order to generate the Benders cuts for the master problem, we solve the dual problem of $SP(x, e|\hat{z})$. We define the dual variables A_j, B_i, α_j, β_i, γ_{ij},δ_{jm}, λ_{jk}, μ_{jk}, ρ_i, τ_j, θ_j, η_i, π_i and σ_j corresponding to the constraints (4.21)-(4.34), respectively. Then the dual subproblem, $DSP(\cdot|\hat{z})$ can be stated as follows:

$$\begin{aligned}
\text{Max} \qquad & Z_{\text{DSP}} = \sum_{i\in\mathcal{I}} \beta_i - \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} \hat{z}_j^c \gamma_{ij} - \sum_{j\in\mathcal{J}}\sum_{m\in\mathcal{J}\setminus\{j\}} (\sum_{i\in\mathcal{I}} R_i \hat{z}_j^c)\delta_{jm} \\
& + \sum_{j\in\mathcal{J}} E_j\{\hat{z_j^c}\,\theta_j - \sigma_j - p\,\tau_j\} + \sum_{i\in\mathcal{I}} E_i\{(1-\hat{z_i^c})\,\eta_i - \pi_i - p\,\rho_i\} \\
& - \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} (\sum_{i\in\mathcal{I}} R_i)\{\hat{z}_k^u\,\lambda_{jk} + \hat{z}_j^c\,\mu_{jk}\} \qquad (4.36)
\end{aligned}$$

subject to

$$(w + v\,D_{jk}^2)\,T\,A_j + \alpha_j - \lambda_{jk} - \mu_{jk} \le 0 \qquad \forall\, j \in \mathcal{J},\ k \in \mathcal{K} \qquad (4.37)$$

$$(w + v\,D_{jm}^2)\,T\,A_j + w\,T\,A_m + \alpha_j - \alpha_m - \delta_{jm} \le 0 \qquad \forall\, j, m \in \mathcal{J} \qquad (4.38)$$

$$\begin{aligned}
& (w + c\,s)R_i\,T\,A_j + (w + v\,D_{ij}^2)R_i\,T\,B_i \\
& \qquad - (1-s)\,R_i\,\alpha_j + \beta_i - \gamma_{ij} \le 0 \quad \forall\, i \in \mathcal{I},\ j \in \mathcal{J} \qquad (4.39)
\end{aligned}$$

$$-A_j + \theta_j - \sigma_j - \tau_j \le t_1(1/|\mathcal{I}|) \qquad \forall\, j \in \mathcal{J} \qquad (4.40)$$

$$-B_i + \eta_i - \pi_i - \rho_i \le t_1(1/|\mathcal{I}|) \qquad \forall\, i \in \mathcal{I} \qquad (4.41)$$

$$\sum_{j\in\mathcal{J}} \theta_j + \sum_{i\in\mathcal{I}} \eta_i \le 1 \qquad (4.42)$$

$$\sum_{j\in\mathcal{J}} \sigma_j + \sum_{i\in\mathcal{I}} \pi_i \ge 1 \qquad (4.43)$$

$$\gamma_{ij},\ \delta_{jm},\ \lambda_{jk},\ \mu_{jk},\ \theta_j,\ \eta_i,\ \pi_i,\ \sigma_j \ge 0 \qquad \forall\, i \in \mathcal{I},\ j, m \in \mathcal{J},\ k \in \mathcal{K} \qquad (4.44)$$

$$A_j,\ B_i,\ \alpha_j,\ \beta_i \text{ unrestricted} \qquad \forall\, i \in \mathcal{I},\ j \in \mathcal{J} \qquad (4.45)$$

The Benders cut

After solving the dual subproblem $DSP(\cdot|\hat{z})$, the Benders cuts (BCuts) can be generated using the values of dual variables and an auxiliary continuous variable B as follows:

$$\begin{aligned} \mathrm{B} \geq & \sum_{i\in\mathcal{I}} \hat{\beta}_i - \sum_{j\in\mathcal{J}} E_j\,(\hat{\sigma}_j + p\,\hat{\tau}_j) + \sum_{i\in\mathcal{I}} E_i\,(\hat{\eta}_i - \hat{\pi}_i - p\,\hat{\rho}_i) - \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} \hat{\gamma}_{ij} z_j^c \\ & - \sum_{j\in\mathcal{J}} \sum_{m\in\mathcal{J}\setminus\{j\}} (\sum_{i\in\mathcal{I}} R_i\,\hat{\delta}_{jm}\,z_j^c) + \sum_{j\in\mathcal{J}} E_j\hat{\theta}_j\,z_j^c - \sum_{i\in\mathcal{I}} E_i\,\hat{\eta}_i\,z_i^c \\ & - \sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}} (\sum_{i\in\mathcal{I}} R_i)\{\hat{\lambda}_{jk}\,z_k^u + \hat{\mu}_{jk}\,z_j^c\} \end{aligned} \tag{4.46}$$

IV.2.2. Benders Master Problem

For given the values of all dual variables from the dual subproblem $DSP(\cdot|\hat{z})$, the Benders master problem $MP(z|\cdot)$ is essentially a minimization problem that gives a tentative network configuration (the selection of CH and sink locations) and a lower bound of the original model **(M3-E1)**. The master problem $MP(z|\cdot)$ can be stated as follows:

$$\text{Min} \qquad Z_{\text{MP}} = t_2(\sum_{j\in\mathcal{J}} z_j^c/E_j) + B \tag{4.47}$$

subject to

$$\sum_{j\in\mathcal{J}} z_j^c = H \tag{4.48}$$

$$\sum_{k\in\mathcal{K}} z_k^u = U \tag{4.49}$$

(constraints for the set of *BCuts*) (4.50)

$$z_j^c,\ z_k^u \in \{0,1\},\ B \geq 0 \qquad\qquad \forall\, j\in\mathcal{J},\ k\in\mathcal{K} \tag{4.51}$$

At each iteration, $MP(z|\cdot)$ incorporates a new Benders cut and solve an integer program to obtain the values of the binary variables z_j^c and z_k^u. In particular, constraints (4.50) are the same as constraints (4.46). At each iteration, we obtain a new dual solution of $DSP(\cdot|\hat{z})$, substitute it into constraints (4.46) and then add it to the $MP(z|\cdot)$ and resolve the master problem.

IV.3. Approaches for Accelerating the BD Algorithm

We observe that, the direct implementation of classical BD approach in our model **(M3-E1)** often converges slowly. This is due to the following reasons: (1) BD approach starts the iterative procedure by solving the master problem without any Benders cuts. However, the initial selection of cuts can have a profound effect upon the performance of Benders algorithm (Magnanti and Wong, 1981). (2) Due to the degeneracy of the subproblem $SP(x, e|\hat{z})$, there exists multiple dual optimal solutions for $DSP(\cdot|\hat{z})$. The first obtained optimal solution to $DSP(\cdot|\hat{z})$ may not lead to a strong cut. (3) The master problem MP$(z|\cdot)$ must be solved each time a new Benders cut is added. As the number of iteration increases, the complexity and the size of MP$(z|\cdot)$ increases dramatically, and consequently makes solving MP$(z|\cdot)$ time-consuming. In order to circumvent these difficulties, we explore several techniques (as discussed below) to accelerate the convergence of the BD algorithm.

IV.3.1. The Upper Bound Heuristic Algorithm

In this section, we present an efficient heuristic algorithm that provides a good feasible solution in reason above mentioned. The aim of our heuristic algorithm is to find a good upper bound and facilitate the generation of good initial Benders cuts. Specifically, we use the solution obtained from the heuristic as an input solution and

solve the dual subproblem $DSP(\cdot|\hat{z})$ for generating an initial Benders cut so that it can be added to the master problem $MP(z|\cdot)$ in the next iteration. This is in contrast to initially solving the $MP(z|\cdot)$ without any cuts in a typical BD implementation.

In this heuristic, we attempt to avoid coincidentally well-positioned sensors being selected as CHs repeatedly in successive periods and to protect low-energy sensors from being selected as CHs. For this purpose, we consider only a subset of sensors with higher-energy as the set of candidate CHs $\mathcal{J}$. This is preferable from an energy dissipation minimization point-of-view. We denote this subset as $\mathcal{I}^R$ since it is a subset of sensor set $\mathcal{I}$. Specifically, to determine the $\mathcal{I}^R$ set, we use a threshold value TH_Ψ calculated as $\Psi\%$ of the average initial energy level at the sensors, i.e., $TH_\Psi = (\Psi/100) * (\sum_{i\in\mathcal{I}} E_i/|\mathcal{I}|)$ and $\mathcal{I}^R = \{i \in \mathcal{I} : E_i \geq TH_\Psi\}$.

In `the Upper Bound Heuristic`, given in Algorithm 6, we proceed as follows. First, we note that its core algorithm (lines 3-10) works in an iterative fashion. At each iteration, we determine the set $\mathcal{I}^R$ based on a threshold value TH_Ψ (line 3); solve the model **(M3-E1)** assuming $\mathcal{J} = \mathcal{I}^R$ (line 4). Thus, we obtain the current solution $\mathcal{S}^c$ represented by the CH and sink locations $\mathcal{C} = \{j \in \mathcal{I} : z^c_j = 1\}$ and $\mathcal{D} = \{k \in \mathcal{K} : z^u_k = 1\}$, along with the objective value $Z(\mathcal{S}^c)$ (line 4). While solving **(M3-E1)** , we employ a stopping criterion given by a `TiLim` (CPLEX parameter) time limit to alleviate the problem of excessive runtimes. If an improvement $\hat{G}*$ (line 5) over the best solution $\mathcal{S}^b$ is obtained, then $\mathcal{S}^c$ becomes the new $\mathcal{S}^b$ (lines 6-8). We decrease Ψ by a constant gradient g (line 10), update the set $\mathcal{I}^R$ and then resolve the problem. The algorithm terminates when no improving solution is found or it reaches the maximum iteration $MaxIter$. In our numerical studies, given in Section IV.4, we set the values of *Maxiter*, initial Ψ, and g as 30, 140 and 10, respectively.

Algorithm 6 `The Upper Bound Heuristic`

initialize *Maxiter*, Ψ, g, $Z(\mathcal{S}^b) = \infty$;
while *Maxiter*> 0 and $G^* > 0$ **do**
  $\mathcal{I}^R = \{i \in \mathcal{I} : E_i \geq TH_\Psi\}$;
  $\mathcal{J} = \mathcal{I}^R$, solve (**M3**$-$**E1**) with `TiLim` to obtain $Z(\mathcal{S}^c)$;
  $G^* = Z(\mathcal{S}^b) - Z(\mathcal{S}^c)$;
  if $G^* > 0$ **then**
    $\mathcal{S}^b = \mathcal{S}^c$, $Z(\mathcal{S}^b) = Z(\mathcal{S}^c)$;
  end if
  Maxiter $=$ *Maxiter*-1;
  $\Psi = \Psi - g$;
end while
Return $\mathcal{S}^b$ and $Z(\mathcal{S}^b)$

Although this procedure is simple, it is very effective in terms of solution quality and serves the purpose of generating the initial Benders cut with inexpensive computational times. As illustrated later in section IV.4, combining the upper bound heuristic and BD framework promotes faster convergence, especially for larger instances.

IV.3.2. Strengthening the Benders Cuts

Due to the degeneracy of the subproblem $SP(x, e|\hat{z})$, there exists multiple dual optimal solutions for $DSP(\cdot|\hat{z})$, each defining a different Benders cut; some cuts dominate the others. Hence, it is important to identify the optimal dual solution corresponding to a stronger Benders cut. Magnanti and Wong (1981) define the strongness (or dominance) of a Benders cut for a general optimization problem given by $Min_{y\in Y, z\in R}\{z : z \geq f(u) + y\, g(u), \forall u \in U\}$ as follows: The cut $z \geq f(u^1) + yg(u^1)$ dominates or is stronger than the cut, $z \geq f(u) + y\, g(u)$ if

$f(u^1) + yg(u^1) \geq f(u) + y\, g(u), \forall u \in U$ with a strict inequality for at least one point $y \in Y$. The use of the strong Benders cuts can facilitate better lower bounds and increase the algorithm efficiency, as shown for various problem settings in Magnanti and Wong (1981), Roy (1986), Wentges (1996), Üster et al. (2007).

For our problem, we adopt a two-phase approach presented in Üster et al. (2007) to strengthen the Benders cuts. This is based on the observation that, in the cut given in (4.46), if $\bar{z}_j = 0$, one can modify its coefficient without changing the objective function value, provided feasibility is maintained. Hence, to strengthen the Benders cuts, we aim to modify the values of dual variables associated with $\bar{z}_j = 0$ to make the z_j coefficient larger. Specifically, in the first phase, for solving $DSP(\cdot|\hat{z})$, we only obtain the values of the dual variables for which the associated binary variables z_j^c and z_k^u have values equal to 1. Note that, for the rest of the dual variables, the associated z_j^c and z_k^u values are 0. Hence, the elimination of the remaining dual variables in the first phase cannot affect the objective function value (4.36). In the second phase, we fix the values of the dual variables obtained from the first phase and then solve for other dual variables using a modified version of $DSP(\cdot|\hat{z})$ given in (4.52). The detailed description of two-phase approach is given as follows.

In Phase I, we only obtain the values of the dual variables for which the associated binary variables z_j^c and z_k^u have values equal to 1. We denote the reduced $\mathcal{J}$ set $\mathcal{J}_R$ and the reduced $\mathcal{K}$ set $\mathcal{K}_R$ as $\mathcal{J}_R = \{j \in \mathcal{J} : z_j^c = 1\}$ and $\mathcal{K}_R = \{k \in \mathcal{K} : z_k^u = 1\}$. We solve the dual subproblem $DSP(\cdot|\hat{z})$, assuming that $\mathcal{J} = \mathcal{J}^R$ and $\mathcal{K} = \mathcal{K}^R$. In Phase II, we focus on computing the dual variables for which the associated binary variables z_j^c and z_k^u have values equal to 0. To this end, we solve the following linear programming problem for strong Benders cuts.

$$\text{Max} \sum_{j\in\mathcal{J}} E_j\theta_j - \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}}\gamma_{ij} - \sum_{j\in\mathcal{J}}\sum_{m\in\mathcal{J}\setminus\{j\}}(\sum_{i\in\mathcal{I}} R_i)\,\delta_{jm} - \sum_{i\in\mathcal{I}} E_i\,\eta_i$$
$$-\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{K}}(\sum_{i\in\mathcal{I}} R_i)\{\lambda_{jk}+\mu_{jk}\} \tag{4.52}$$

subject to (4.37)-(4.45)

Note that, in the problem (4.52), the objection function represents the sum of all the coefficient associated with $\bar{z}_j = 0$ given in the Bender cut (4.46) and the constraints are the same as $DSP(\cdot|\hat{z})$. Also, in order not to affect the objective function value in $DSP(\cdot|\hat{z})$, the values of the dual variables associated with $\bar{z}_j = 1$ in Phase II need to remain the same as in Phase I. Specifically, the values of the dual variables found in Phase I are substituted in the problem (4.52). Then we solve the problem (4.52) to obtain the values of the dual variables for which the associated binary variables z_j^c and z_k^u have values equal to 0.

IV.3.3. ε-Optimal Approach

In the BD algorithm, we add a new Benders cut into the master problem MP$(z|\cdot)$ at each iteration. As the number of iterations increases, the complexity and the size of MP$(z|\cdot)$ increases dramatically, and consequently makes MP$(z|\cdot)$ difficult to solve. In order to decrease the solution time of MP$(z|\cdot)$, we utilize the ε-optimal approach introduced in Geoffrion and Graves (1974). Specifically, we add one additional constraint in the MP$(z|\cdot)$, given as

$$t_2(\sum_{j\in\mathcal{J}} z_j^c/E_j) + B \;\le\; UB(1-\varepsilon) \tag{4.53}$$

where UB and ε denote the best upper bound and the acceptable optimality gap, respectively. In an iteration, instead of solving the MP($z|\cdot$) to optimality, we stop the branch-and-cut (using CPLEX) once a feasible solution is obtained. Using the values of the z variables given by this feasible solution, we then solve $DSP(\cdot|\hat{z})$ and generate new Benders cuts. By doing so, the runtime for the MP($z|\cdot$) can be substantially reduced at each iteration. Note that the feasible solution obtained is no longer a valid lower bound and the algorithm terminates when $MP(z|\cdot)$ cannot find a feasible solution, which verifies that the best upper bound is within ε from optimality.

IV.3.4. ε-Optimal BD Framework

In order to improve the computational efficiency of the typical BD algorithm, our algorithm brings together the above components including the upper bound heuristic, strengthening the Benders cuts and ε-optimal approach to speed up the convergence of the BD algorithm. We outline the overall framework of the ε-Optimal BD Algorithm given in Algorithm 7. We denote *Iterno*, *UB*, *LB*, and $(\bar{\mathbf{x}}_{\mathbf{best}}, \bar{\mathbf{e}}_{\mathbf{best}}, \bar{\mathbf{z}}_{\mathbf{best}})$ as the number of iterations, the best upper bound, the best lower bound, and the best feasible solution, respectively.

In particular, we first apply the upper bound heuristic to obtain a feasible solution (an upper bound) and solve the dual subproblem $DSP(\cdot|\hat{z})$ for generating an initial Benders cut so that it can be added to the master problem $MP(z|\cdot)$ in the beginning (line 1-6). This is in contrast to initially solving the $MP(z|\cdot)$ without any cuts in a typical BD implementation. Then we incorporate ε-optimal approach, i.e., instead of solving the MP($z|\cdot$) to optimality, we stop the branch-and-cut using CPLEX once a feasible solution is obtained (line 6 and 16). Using the values of the z variables given by this feasible solution, we then solve $DSP(\cdot|\hat{z})$ and generate new Benders cuts via a two-phase approach (line 4-5, 9 and 15). The best upper bound UB and the best

solution $\bar{z}_{best}$ are updated if improved by the current solution $\hat{z}$ (line 10-14), and a new iteration is started. The algorithm terminates when the MP$(z|\cdot)$ cannot find a feasible solution, which verifies the best upper bound UB is within ε from the optimal solution. Once the iterations are completed, we solve the subproblem $SP(x, e|\bar{z}_{best})$ to obtain the values of continuous variables (line 18). Upon the completion of the algorithm, we report the best feasible solution along with the best upper bound (line 19).

Algorithm 7 ε-Optimal BD Algorithm

1: **initialize** Algorithmic parameters for `The Upper Bound Heuristic`, `Iterno` $= 0$;
2: Apply `The Upper Bound Heuristic` to $\mathcal{S}^b$ and record $Z(\mathcal{S}^b)$;
3: Set $UB = Z(\mathcal{S}^b)$ and $\bar{z}_{best} = \mathcal{S}^b$;
4: Solve $DSP(\cdot|\hat{z})$ to obtain the values for all dual variables;
5: Generate the initial Benders cut and incorporate it into $MP(z|\cdot)$;
6: Solve $MP(z|\cdot)$ to obtain the values for $\hat{z}$ and Z_{MP};
7: **while** $MP(z|\cdot)$ has a feasible solution **do**
8: `Iterno` = `Iterno` + 1;
9: Solve $DSP(\cdot|\hat{z})$ to obtain the values for all dual variables and Z_{DSP}.
10: **if** $Z_{\text{MP}} - B + Z_{\text{DSP}} < UB$ **then**
11: $UB = Z_{\text{MP}} - B + Z_{\text{DSP}}$;
12: $\bar{z}_{best} = \hat{z}$;
13: Update the incumbent value UB in constraint (4.53);
14: **end if**
15: Generate *BCuts* and incorporate them into $MP(z|\cdot)$;
16: Solve $MP(z|\cdot)$ to obtain the value for $\hat{z}$ and Z_{MP}.
17: **end while**
18: Solve $SP(x, e|\bar{z}_{best})$ for all continuous variables $\bar{x}_{best}$ and $\bar{e}_{best}$;
19: Return $(\bar{x}_{best}, \bar{e}_{best}, \bar{z}_{best})$ and the UB.

Note that, if we do not incorporate ε-optimal approach into the BD framework, we will solve the master problem MP($z|\cdot$) to optimality at each iteration. The algorithm terminates when the optimality gap, ((UB - LB)/ LB), is no greater than $\varepsilon \geq 0$.

IV.4. Computational Results

In this section, we conduct a computational study to establish the performance of Benders decomposition algorithm in a single-period setting. The comparisons illustrate the benefit of utilizing the upper bound heuristic, strengthened Benders cuts and ε-optimal framework. The computational experiments are performed on a machine with two 2.66-GHz Intel XEON precessors and 12.0 GB RAM and the algorithms are implemented in C++ utilizing STL (Standard Template Library) and Concert Technology when CPLEX 11 was used.

IV.4.1. Random Test Instance Generation

We generate our random test instances in such a way that a wide range of input data value for the problem parameters is considered. In particular, we generate test instances under two data settings (Setting I – Small instances and Setting II – Large instances) by varying the number of sensors $|\mathcal{I}|$, the number of candidate sinks $|\mathcal{K}|$, the number of required CHs H, and the number of required sinks U. We provide 48 problem classes, as shown in Table 6. For each of problem class, we generate 10 random instances. In all of the instances, we assume that the initial energy levels at the sensors are uniformly distributed in the range [0.1, 0.5] J.

Unless stated otherwise, all of the input and algorithmic parameter values are set as mentioned previously. Furthermore, we randomly generate $|\mathcal{I}|$ sensor coordinates

uniformly distributed in a square of size $N(m)$. Also, the candidate sites for sinks, $\mathcal{K}$, are generated randomly on the periphery of the sensor field. We set the period length as $T = 4000$ *time-units*; and the aggregation ratio as $s = 0.3$. In addition, we set the weight $t = 5$ due to some empirical testing.

IV.4.2. Computational Experiments

We consider a single-period setting and evaluate the performance of Benders decomposition algorithm on the basis of solution quality and time, where the optimality gap is within 2%. In addition, we evaluate the performance of our upper bound heuristic via utilizing two different benchmarks: (1) For Setting I – small instances, we utilize the exact solutions for benchmarking. We obtain the optimal solution for model **(M3-E1)** by using the exact branch-and-cut implementation in CPLEX 11 with default parameters. (2) For Setting II – large instances, we resort to obtain the lower bound from BD approach as another benchmark solution to evaluate the effectiveness of the heuristic algorithm.

Table 7 summarizes the computational results for Setting I – small instances. The values reported in columns 2 to 6 are averaged over ten instances. The second column (T^O_{ave}) concerns the exact solution times (in seconds) for these instances using CPLEX 11. The third column (T^H_{ave}) includes the average solution times obtained using our upper bound heuristic given in Algorithm 6. We observe that they are very significantly lower than the solution times for exact solutions. In terms of solution quality, we calculate the optimality gap as $\Delta^O = 100 * (Z^H - Z^O)/Z^O$, where Z^O and Z^H represent the optimal objective function value and heuristic value for an instance, respectively. The average optimality gaps, Δ^O_{ave}, are reported in the fourth column of Table 7. We observe that the gaps are significantly small for all the instances, illustrating that the upper bound heuristic approach provides near-optimal solutions

Table 6: Problem setting used in computational testing

Setting I – Small instances					Setting II – Large instances				
Class	$\|\mathcal{I}\|$	$\|\mathcal{K}\|$	H	U	Class	$\|\mathcal{I}\|$	$\|\mathcal{K}\|$	H	U
SS1	50	8	3	1	LS1	150	8	9	1
SS2				2	LS2				2
SS3			6	1	LS3			18	1
SS4				2	LS4				2
SS5		16	3	1	LS5		16	9	1
SS6				2	LS6				2
SS7			6	1	LS7			18	1
SS8				2	LS8				2
SS9	75	8	5	1	LS9	200	8	12	1
SS10				2	LS10				2
SS11			9	1	LS11			24	1
SS12				2	LS12				2
SS13		16	5	1	LS13		16	12	1
SS14				2	LS14				2
SS15			9	1	LS15			24	1
SS16				2	LS16				2
SS17	100	8	6	1	LS17	250	8	15	1
SS18				2	LS18				2
SS19			12	1	LS19			30	1
SS20				2	LS20				2
SS21		16	6	1	LS21		16	15	1
SS22				2	LS22				2
SS23			12	1	LS23			30	1
SS24				2	LS24				2

compared with the exact solutions.

The fifth column (T_{ave}^{BD}) in Table 7 concerns the average solution time using BD framework reinforced with the upper bound heuristic and the strengthened Benders cuts. In the last column, we report the average runtimes performed by ϵ-Optimality BD approach given in Algorithm 7. As mentioned before, the termination criterion is set to 2% optimality gap. We observe that they are very significantly lower than the solution times for exact solutions.

As the run time increases exponentially with the instance size, only small instances can be solved to optimality within a reasonable time. Table 8 summarizes the computational results for Setting II – large instances. In this setting, we found that CPLEX (using default setting) cannot be solved to optimality for all the instances within 4 hours. As in Table 7, the values reported in columns 2 to 5 are averaged over ten instances. The second (T_{ave}^{H}) and third (Δ_{ave}^{LB}) columns concern the average solution times using the upper bound heuristic and the average optimality gaps, respectively. Similar to Table 7, we calculate the optimality gap as $\Delta^{LB} = 100 * (Z^H - Z^{LB})/Z^{LB}$, where Z^{LB} and Z^H represent the lower bound from Benders approach and the upper bound from the heuristic approach, respectively. In the last two columns, we report the average runtimes performed by BD algorithm and ϵ-Optimality BD approach, respectively. Again, the termination criterion is set to 2% optimality gap. Based on the results in Table 8, we fist note that the optimality gap Δ_{ave}^{LB} is significantly small for all the problem instances. This demonstrates that the upper bound heuristic provides consistently good quality solutions with small amount of time. Second, the computational times for all the instances in the BD approach are much less than CPLEX, in which the instances are not solved in 4 hours of runtime. This illustrates the effectiveness of our BD approach in addressing a large-size problem. Third, the ε-Optimal BD approach performs better than the BD approach

Table 7: Computational results for small-size problems

Class	T_{ave}^{O}	T_{ave}^{H}	Δ_{ave}^{O}	T_{ave}^{BD}	$T_{ave}^{\varepsilon-Opt}$
SS1	19.90	1.79	0.51	6.97	2.41
SS2	12.69	1.52	0.42	4.07	2.17
SS3	8.11	0.88	0.18	3.71	2.76
SS4	6.57	0.65	0.16	3.24	2.83
SS5	37.12	2.24	0.42	7.88	2.40
SS6	11.13	1.13	0.17	4.11	3.29
SS7	20.00	1.74	0.49	4.54	2.26
SS8	8.49	0.83	0.11	3.79	2.89
SS9	179.74	0.44	0.20	3.47	3.56
SS10	200.24	0.74	0.23	12.89	9.28
SS11	59.04	0.40	0.15	2.97	3.37
SS12	60.86	0.66	0.14	10.41	11.60
SS13	269.70	0.57	0.25	3.82	3.75
SS14	349.99	1.10	0.22	14.35	9.48
SS15	73.86	0.49	0.14	3.19	3.67
SS16	94.43	1.27	0.12	12.48	11.99
SS17	1185.85	6.64	0.21	13.33	6.41
SS18	976.98	18.17	0.05	94.56	29.89
SS19	421.41	5.10	0.09	10.57	6.75
SS20	291.35	10.72	0.05	58.32	33.76
SS21	2058.64	8.79	0.16	15.76	7.49
SS22	1609.08	44.48	0.05	126.85	31.28
SS23	669.96	8.33	0.09	15.61	7.28
SS24	295.45	17.56	0.05	77.31	38.40

Table 8: Computational results for large-size problems

Class	T_{ave}^{H}	Δ_{ave}^{LB}	T_{ave}^{BD}	$T_{ave}^{\varepsilon-Opt}$
LS1	48.11	2.03	177.90	126.49
LS2	25.38	1.98	221.02	114.88
LS3	18.18	2.08	952.34	304.56
LS4	14.34	2.07	1294.80	368.78
LS5	52.86	2.01	212.89	153.95
LS6	40.40	2.02	230.22	152.19
LS7	29.79	2.09	950.64	321.52
LS8	32.78	2.05	953.85	339.17
LS9	1.62	2.27	109.76	124.86
LS10	1.35	2.13	173.92	159.55
LS11	47.75	2.14	941.55	568.72
LS12	31.32	2.05	1785.13	799.03
LS13	1.97	2.20	135.48	100.73
LS14	2.01	2.08	185.45	167.95
LS15	50.79	2.11	1349.47	551.26
LS16	53.19	2.05	1516.96	831.53
LS17	22.59	2.04	701.30	287.12
LS18	24.85	2.06	593.62	394.33
LS19	68.08	2.14	1021.50	359.28
LS20	68.81	2.03	1544.10	757.07
LS21	45.28	2.01	686.81	277.00
LS22	53.94	2.03	707.67	412.63
LS23	71.00	2.11	1610.62	444.78
LS24	74.06	2.10	2149.65	640.31

in terms of solution times .

In general, we can conclude that: (1) Our upper bound heuristic approach provides high-quality solutions with much less runtime than CPLEX; (2) Benders decomposition method, which amalgamates various problem and solution characteristics, is very effective in addressing a rather complex problem; (3) ϵ-Optimality framework further improves the performance of the algorithm; (4)Combining the heuristics, ε-Optimal approach and BD framework promotes faster convergence, especially for larger instances.

IV.5. Summary and Conclusions

In this chapter, we study an integrated topology control and routing problem in WSNs, while incorporating the fixed cost associated with locating the CHs into the objective function. We develop a mixed-integer linear programming (MILP) model to determine the sink and cluster-head locations as well as the data flow, over a time horizon. We adopt the objective as the combination of the average energy usage, the range of remaining energy distribution and the fixed cost associated with locating the CHs. By setting a higher fixed cost with lower energy nodes, our model attempts to avoid some well-positioned sensors, being selected as CHs repeatedly in successive periods and to protect low-energy sensors from quick energy depletion.

On the methodology side, we develop an effective Benders decomposition solution approach that incorporates an upper bound heuristic algorithm, the strengthened Benders cuts and an ε-optimal approach. Specifically, we devise a simple efficient heuristic algorithm that provides a good feasible solution (an upper bound) so as to facilitate the generation of initial Benders cuts. We adopt a two-phase approach to strengthen the Benders cuts and utilize the ε-optimal approach to decrease the

solution time of the master problem at each iteration. We note that the optimal solutions obtained by CPLEX and the lower bounds obtained by the Benders approach verify the high quality of the heuristic solutions for small and large instances, respectively. The availability of good lower bounds is facilitated by the good initial Benders cut and the strengthened Benders cuts. Furthermore, ϵ-Optimality framework significantly improves the performance of the BD algorithm. Computational evidence demonstrates that the upper bound heuristic approach provides high-quality solutions in a timely manner. Combining the heuristics, ε-Optimal approach and BD framework promotes faster convergence, especially for larger instances.

In summary, our proposed BD algorithm, which amalgamates various problem and solution characteristics, is very effective in addressing a rather complex problem.

CHAPTER V

SENSOR NETWORK DESIGN/ROUTING PROBLEM WITH SINGLE-SOURCING ASSIGNMENTS

In this chapter, we consider another important extension model **(M3-E2)** to the setting of **(M3)** by incorporating the single-sourcing requirements for CH assignments and explicitly specifying the transmission ranges of sensor nodes. In the light of WSN characteristics, the motivation for the extension model studies in this section is twofold.

1. As the number of sensor nodes in WSNs can be in the order of hundreds or thousands, system scalability is an important factor, i.e., it is crucial to ensure that the network performance does not significantly degrade as the network size increases. According to Tilak et al. (2002); Akkaya and Younis (2005), hierarchical routing and data aggregation can greatly contribute to the overall system scalability and energy efficiency. Specifically, in a cluster-based structure, organizing sensor networks into disjoint clusters can facilitate scalability by constraining the communications between the sensors to a CH within a cluster. For this purpose, we consider the single-sourcing assignments, i.e., each sensor's data is handled by only one CH.

2. WSNs usually exhibit short transmission ranges and their limited operating power also imposes restrictions on the maximum allowable distance between the sensors and the sink nodes (Raghavendra et al., 2006). To overcome this difficulty, we explicitly specify the transmission ranges of sensor nodes for better topology control.

Note that, the objective in (M3) directly addresses the energy usage and the variation in the remaining energy distribution, thus it implicitly provides us the ability to efficiently manage CH selections associated with energy levels of sensors. Therefore, for simplicity, we do not consider the fixed cost associated with locating the CHs in model (M3-E2).

V.1. The Model

We next give the additional notations and the mathematical formation for **(M3-E2)**.

Additional Notations

r Maximum transmission distance (transmission range)

x^c_{ij} 1 if a node i is assigned to a CH j, 0 o.w.

Formulation

$$\textbf{(M3-E2)} \qquad \text{Minimize} \qquad t(1/|\mathcal{I}|)(\sum_{m\in\mathcal{J}} e^c_m + \sum_{i\in\mathcal{I}} e_i) + (E^R_{max} - E^R_{min}) \quad (5.1)$$

subject to

$$\sum_{k\in\mathcal{K}}(w + v\,D^2_{mk})\,T\,x^u_{mk} + \sum_{j\in\mathcal{J}\setminus\{m\}}(w + v\,D^2_{mj})\,T\,x^{cc}_{mj} + \sum_{j\in\mathcal{J}\setminus\{m\}} w\,T\,x^{cc}_{jm} + \sum_{i\in\mathcal{I}}(w + c\,s)\,R_i\,T\,x^c_{im} = e^c_m \qquad \forall\, m \in \mathcal{J} \quad (5.2)$$

$$\sum_{j\in\mathcal{J}}(w + v\,D^2_{ij})\;R_i\,T x^c_{ij} = e_i \qquad \forall\, i \in \mathcal{I} \quad (5.3)$$

$$\sum_{k\in\mathcal{K}} x^u_{mk} + \sum_{j\in\mathcal{J}\setminus\{m\}} x^{cc}_{mj} - \left(\sum_{j\in\mathcal{J}\setminus\{m\}} x^{cc}_{jm} + (1-s)\sum_{i\in\mathcal{I}} R_i\,x^c_{im}\right) = 0\;\forall\, m \in \mathcal{J} \quad (5.4)$$

$$\sum_{j\in\mathcal{J}} x^c_{ij} = 1 \qquad \forall\, i \in \mathcal{I} \quad (5.5)$$

$$x_{ij}^{c} \leq \lfloor r/D_{ij} \rfloor z_j^c \qquad \forall i \in \mathcal{I}, j \in \mathcal{J} \tag{5.6}$$

$$x_{mj}^{cc} \leq \sum_{i \in \mathcal{I}} \lfloor r/D_{mj} \rfloor R_i z_j^c \qquad \forall m, j \in \mathcal{J} \tag{5.7}$$

$$x_{jk}^{u} \leq \sum_{i \in \mathcal{I}} \lfloor r/D_{jk} \rfloor \, R_i z_k^u \qquad \forall j \in \mathcal{J}, k \in \mathcal{K} \tag{5.8}$$

$$x_{jk}^{c} \leq \sum_{i \in \mathcal{I}} \lfloor r/D_{jk} \rfloor \, R_i z_j^c \qquad \forall j \in \mathcal{J}, k \in \mathcal{K} \tag{5.9}$$

$$\sum_{j \in \mathcal{J}} z_j^c = H \tag{5.10}$$

$$\sum_{k \in \mathcal{K}} z_k^u = U \tag{5.11}$$

$$e_i \leq p \, E_i \qquad \forall i \in \mathcal{I} \tag{5.12}$$

$$e_j^c \leq p \, E_j \qquad \forall j \in \mathcal{J} \tag{5.13}$$

$$z_j^c \, E_j - e_j^c \; \leq \; E_{max}^R \qquad \forall j \in \mathcal{J} \tag{5.14}$$

$$(1 - z_i^c) \, E_i - e_i \; \leq \; E_{max}^R \qquad \forall i \in \mathcal{I} \tag{5.15}$$

$$E_{min}^R \leq E_i - e_i \qquad \forall i \in \mathcal{I} \tag{5.16}$$

$$E_{min}^R \leq E_j - e_j^c \qquad \forall j \in \mathcal{J} \tag{5.17}$$

$$x_{ij}^c, \; z_j^c, \; z_k^u \in \{0, 1\} \qquad \forall \, i \in \mathcal{I}, \;\; j \in \mathcal{J}, \;\; k \in \mathcal{K} \tag{5.18}$$

$$x_{ij}^{cc}, \; x_{jk}^u, \; e_i, \; e_j^c, \; E_{max}^R, \; E_{min}^R \geq 0 \qquad \forall \, i \in \mathcal{I}, \;\; j \in \mathcal{J}, \;\; k \in \mathcal{K} \tag{5.19}$$

The objective function minimizes the weighted sum of average energy consumption and range of remaining energy levels. Constraints (5.2) and (5.3) assign the values of the total energy consumed by a CH and a sensor node, respectively. Constraints (5.4) state the data flow balance at each CH node and constraint (5.5) guarantees that

each sensor is assigned to one CH. Constraints (5.6)–(5.9) assign the values of binary variables related to CH and sink node location selections. Note that these constraints also ensure that only the nodes within the transmission range can communicate with each other. Constraints (5.10) and (5.11) establish the required number of CHs and sink nodes, respectively. Constraint sets (5.12) and (5.13) ensure that the total energy consumed at a node cannot exceed the total available energy at the corresponding sensors. Constraint sets (5.14) and (5.15) give the maximum remaining energy at a sensor node and constraint sets (5.16) and (5.17) give the minimum remaining energy at a sensor node. Finally, (5.18) and (5.19) include the integrality and non-negativity of the decision variables.

In the model **(M3-E2)**, having single-source constraints $x_{ij}^c \in \{0,1\}$ is one of the critical reasons to make the problem difficult to solve. If we replace them by the simpler requirement that $x_{ij}^c \geq 0$, we get a relaxation problem, denoted by **(MR)**, which, in turn, provide a lower bound for the overall model. On the other hand, model **(M3-E2)** can be viewed as a combination of three energy metrics: total energy consumption, maximum remaining energy and minimum remaining energy. To gain insights into the characteristics of the model and solve larger and realistic instances efficiently as discussed later, we introduce the following reduced models.

(RP1) Minimize the total energy used in the network:

$$\text{Min} \sum_{m \in \mathcal{J}} e_m^c + \sum_{i \in \mathcal{I}} e_i \qquad \text{s.t.} \quad (5.2) - (5.19).$$

(RP2) Minimize the maximum remaining energy at a sensor node:

$$\text{Min } E_{max}^R \qquad \text{s.t.} \quad (5.2) - (5.19).$$

(RP3) Maximize the minimum remaining energy at a sensor node:

$$\text{Max } E^R_{min} \qquad \text{s.t. } (5.2) - (5.19).$$

Note that for reduced model (RP1), constraint sets (5.14)–(5.17) are redundant and can be omitted from the formulation. Similarly, for (RP2) and (RP3), we have redundant constraint sets (5.14)–(5.15) and (5.16)–(5.17), respectively.

V.2. Parallel Heuristic Algorithm

Due to the wide practical applications in WSN, solving the problem in a reasonable amount of time is important. Our mathematical model dictates discrete optimization formulation, even small size problems are highly impractical to be solved using exact optimization methods. In fact, a heuristic method often is the only practical approach for solving complex problems in realistic scale. However, the computation times associated with the exploration of the solution space during the search procedure may be very large. Consequently, we may encounter the difficulty of offering a consistently high level of performance over a wide variety of problem settings and characteristics. Therefore, we aim to adapt the heuristic solution approach developed in section III.5 to a parallel computing architecture in order to address the above issue. Our goal is to find near-optimal solutions in a short amount of time and to exploit parallelism to improve the algorithm effectiveness and robustness.

Although the original model (M3-E2) , the reduced models (RP1), (RP2), and (RP3) and relaxed model (MR) differ structurally, they embody the same sets of binary variables. For this purpose, we utilize the same solution representations presented in section III.5.1. First, we represent a solution by a finite set of fixed CH locations, which we denote it as the set $\mathcal{C} = \{j \in \mathcal{J} : z^c_j = 1\}$ and $|\mathcal{C}| = H$.

We fix the corresponding binary variables z_j^c at 1 by using $\mathcal{C}$, and solve associated subproblem, denoted by (**SubC**), to evaluate the goodness of such a solution. Similarly, in the second case, we represent a solution by the fixed sink locations, i.e., $\mathcal{D} = \{k \in \mathcal{K} : z_k^u = 1\}$ and $|\mathcal{D}| = U$, and, in turn, we have the associated subproblem (**SubU**) obtained by fixing binary variables z_k^u at 1 from the set $\mathcal{D}$.

The algorithm presented in the paper has three main computing tasks: (1) the heuristic search procedure ; (2) cut generation; and (3) model relaxation. Below, we provide the details of how each of the computing tasks is performed as well as the parallel implementation.

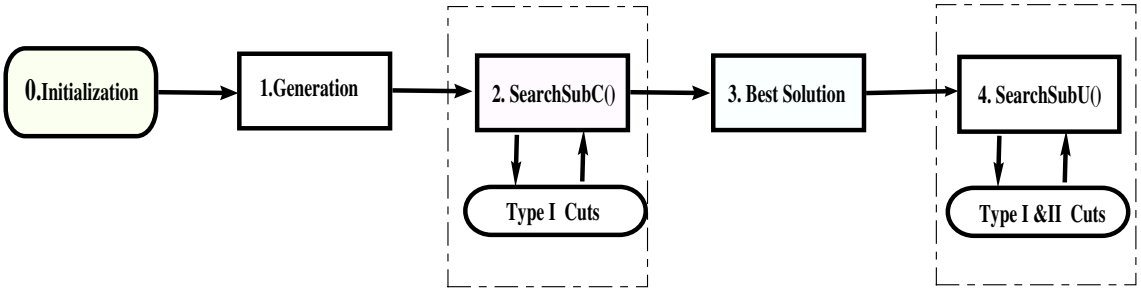
V.2.1. The Heuristic Search Procedure

The general heuristic search procedure that we use in this chapter is based on a variance of the solution approach presented in section III.5. The difference is summarized as follows. (1) To simplify the heuristic method, we exclude step 6 and 7 in the complete procedure presented in section III.5.3. (2) In addition to the objective function value-based cuts, we also incorporate the cut inequalities generating from the reduced models (RP1), (RP2), and (RP3) to the models.

We consider a combination of multiple neighborhoods based on two solution representations and employ a multi-start scheme to instill diversification in searching the solution space in the start. We generate Q initial solutions where each is represented by a set of CHs, $\mathcal{C}$. One of these Q solutions is found by using the construction heuristic and the others are generated randomly. In the next step, we apply the solution improvement procedure which is devoted to improving the initial solutions. In particular, based on two solution representations, we devise two search procedures `Search`**SubC**`()` and `Search`**SubU**`()`, each is also reinforced with cut generation. The order in which the components are executed is indicated in Figure 10. Specifically,

we apply the neighborhood search procedure `SearchSubC()` to each of Q solutions independently. Upon completion of the search procedures, we identify the best solution $\mathcal{S}^b$ among the Q solutions. Then we apply the `SearchSubU()` procedure and also utilize the best solution $\mathcal{S}^b$ as the input solution. We also incorporate cut generation for the purpose of effectiveness. Next, we discuss how each of the components is performed.

Figure 10: The Heuristic Search Structure



Construction Heuristic

In WSN, communication is normally carried from multiple data sources to the sink node (i.e., many to one). Thus, it is preferable to have some CHs close to the sink nodes. Since we assume that the sink nodes are around the periphery of the sensor field, the basic topology desired in data-gathering is to ensure that some CHs are selected close to the periphery while the remaining CHs are chosen from the center area of the sensor field. Also, the model aims to distribute the energy usage across the network uniformly and, in turn, the high energy level nodes are more promising to be selected as CHs at a period.

To achieve the goals mentioned above, we consider the subset $\mathcal{I}^R$ (a high-energy

subset of $\mathcal{I}$) as the set of candidate CHs, $\mathcal{J}$, and also identify two disjoint sets of candidate CHs as (1) $\mathcal{F}_1 \subset \mathcal{J}$, close to the center of the sensor field, and (2) $\mathcal{F}_2 = \mathcal{J} \setminus F_1$, close to periphery. We define a parameter α where $\alpha = |\mathcal{F}_2|/(|\mathcal{F}_1|+|\mathcal{F}_2|)$. The construction heuristic works in an iterative fashion by successively replacing the current solution with a better solution within a certain iterations. At each iteration, it builds a feasible solution starting by randomly picking a node from F_2 as a CH and determines a total of $\lfloor \alpha * H \rfloor$ CHs from F_2, one at a time at the median distance from the current $\mathcal{C}$. Then, the rest of the CHs are picked from set F_1 similarly. The best overall solution is determined as the final initial solution.

The `SearchSubC()` Procedure

In our first procedure, we characterize a solution $\mathcal{C}$ based on the CH selection and accordingly utilize the subproblem (**SubC**) to evaluate the goodness of a solution. We adopt the same procedure as the algorithm 3 presented in section III.5.2.

We employ an h-exchange neighborhood function starting with a 1-exchange, h is increased up to h_{max} under the condition that the solution can be monotonously improved. As the high energy level nodes are preferred to be selected as CHs, we also consider the subset $\mathcal{I}^R$ (a high-energy subset of $\mathcal{I}$) as the set of candidate CHs, $\mathcal{J}$ in the improvement stage. We apply the best-improving strategy to this search procedure, in other words, all neighbors are investigated and the current solution is replaced by the best neighbor. However this may be a time-consuming process due to the complex and extensive search space. Therefore, we are motivated to pursue the use of a restricted candidate list. Specifically, for a CH to be exchanged, we consider only the non-CH candidates in $\mathcal{J}$ which are within its ρ radius. Then, the neighborhood for each node is clearly restricted to a subset of $\mathcal{J}$ and it is very likely to be different in each period, since the set $\mathcal{J}$ will be updated based on the

sensors' remaining energy information at the end of each period. Moveover, in order to quickly evaluate the possible solutions in the neighborhood, we incorporate a simple objective function value based on cut inequality (Type I cuts) into the subproblems for computing efficiency. The detail about the cut generation will be discussed later.

The `SearchSubU()` Procedure

In our second procedure, we characterize a solution $\mathcal{D}$ based on the sink node selection. The solutions are found by a greedy randomized construction method. In particular, we generate a solution by picking the first sink location randomly and the others by ensuring a good separation between them at each iteration. Employing multiple sink nodes in a well-spread pattern is desirable in terms of energy efficiency due to the fact that, such setting will promote a data flow converging to well-apart locations and the energy drainage in the network is more evenly distributed to the sensors. As the optimum solution of (**SubU**) for a given set of sink nodes $\mathcal{D}$ is generally a time-consuming process, we employ a stopping criterion given by a θ optimality gap or τ time limit , whichever is reached first. We update the best solution, if necessary, in an iterative fashion. This is a simple procedure, but it appears very effective in terms of solution quality and serves the purpose of exploring the solution space efficiently with inexpensive computational times. This is especially true when it incorporates two types of cut inequalities that we describe next.

V.2.2. Cut Generation

In our algorithm, we use two types of valid inequalities that have been shown effectiveness in improving the computing efficiency and solution quality.

Type I Cuts

The first type of cut inequalities is based on simple objective function value as presented in section III.5. Moreover, we can apply Type I Cuts into (RP1), (RP2), and (RP3) and their associated subproblems. In particular, let $\bar{Z}_{RP1}$ and $\bar{Z}_{RP2}$ denote an upper bound in (RP1) and (RP2) respectively and let $\underline{Z}_{RP3}$ denote a lower bound in (RP3), then we can utilize the following cut inequalities:

$$\sum_{m\in\mathcal{J}} e_m^c + \sum_{i\in\mathcal{I}} e_i \leq \bar{Z}_{RP1} \tag{5.20}$$

$$E_{max}^R \leq \bar{Z}_{RP2} \tag{5.21}$$

$$E_{min}^R \geq \underline{Z}_{RP3} \tag{5.22}$$

For (M3-E2) or associated subproblems, supposing that $\bar{Z}$ is an upper bound, we have

$$t\left(\sum_{m\in\mathcal{J}} e_m^c + \sum_{i\in\mathcal{I}} e_i\right)/|\mathcal{I}| + E_{max}^R - E_{min}^R \leq \bar{Z}, \tag{5.23}$$

Furthermore, an additional cut inequality is given by

$$t\left(\sum_{i\in\mathcal{I}} E_i/|\mathcal{I}| - E_{max}^R\right) \leq \bar{Z} \tag{5.24}$$

Type II Cuts

The second type of inequalities used in our algorithm generates from the reduced models that we have presented in section V.1. Model (M3-E2) and the reduced models (RP1), (RP2), and (RP3) mainly differ in their objective functions and share the same feasible region. They have very useful properties, as shown in Proposition 1,2 and 3, that we can utilize in generating the valid inequalities for Type II cuts.

Proposition 1

Let Z_{LR}^1 denote the optimal solution (or a lower bound) in model (RP1), then

$$\sum_{m \in \mathcal{J}} e_m^c + \sum_{i \in \mathcal{I}} e_i \geq Z_{LR}^1 \tag{5.25}$$

is a valid inequality for model (M3-E2) and other models (RP2) and (RP3).

Proposition 2

Let Z_{LR}^2 denote the optimal solution (or a lower bound) in model (RP2), then

$$E_{max}^R \geq Z_{LR}^2 \tag{5.26}$$

is a valid inequality for model (M3-E2).

Proposition 3

Let Z_{UR}^3 denote the optimal solution (or an upper bound) in model (RP3), then

$$E_{min}^R \leq Z_{UR}^3 \tag{5.27}$$

is a valid inequality for model (M3-E2).

Proof

The proofs of Proposition 1, 2 and 3 are similar, based on the following observation. Consider an optimization problem (Q) over a discrete set X, given as

$$\textbf{(Q)} \quad \text{Min } f(x) + g(y)$$
$$x, y \in \text{X}$$

Furthermore, consider a reduced problem (Q1):

$$\textbf{(Q1)} \quad \text{Min } f(x)$$
$$x, y \in \text{X}$$

Let (x, y) be any feasible solution in problem (Q) and it is easy to show that x is also a feasible solution in (Q1). Let $Z(Q1)$ be the optimal objective value (or a lower bound) of (Q1), then

$$f(x) \geq Z(Q1)$$

In particular, let $f(x)$ and $Z(Q1)$ represent the mathematical terms $\sum_{m \in \mathcal{J}} e_m^c + \sum_{i \in \mathcal{I}} e_i$ and Z_{LR}^1, respectively. It is straightforward to prove that (5.25) is a valid inequality for model (M3-E2). Similarly, let $f(x)$ represent the term E_{max}^R (or $-E_{min}^R$), and let $Z(Q1)$ represent Z_{LR}^2 (or $-Z_{UR}^3$), we have Proposition 2 and 3 respectively.

Next we prove that (5.25) is also a valid inequality for the models (RP2) and (RP3). We consider another reduced problem (Q2):

$$\textbf{(Q2)} \quad \begin{aligned} &\text{Min } g(y) \\ &x, y \in \text{X} \end{aligned}$$

Since the problems (Q), (Q1) and (Q2) share the same feasible region, (x, y) can be an arbitrary feasible solution in (Q2) which in turn, guarantees that x is also a feasible solution in (Q1). Thus, we can complete the proof of Proposition 1 following the same logic reasoning discussed above.

Preliminary Analysis of the Reduced Models

To gain insights into the characteristics of the reduced models, we solve a set of small size instances to optimality using the exact branch-and-cut implementation (as in CPLEX). We have the following important observations:

1. For model (RP1), it is much easier to be solved compared with (RP2) and (RP3). If we relax the binary variables z_j^c and x_{ij}^c to flow variables, the optimality gap between the optimal solution and the lower bound is significantly small (within

5%). This observation is important as we might obtain a strong lower bound for (RP1) in a relative short time by solving the relaxed model of (RP1).

2. There is no substantial difference among the models (RP1),(RP2) and (RP3) in terms of mean remaining energy values, showing that cut inequality (5.25) might be a strong cut for models (M3-E2), (RP2) and (RP3).

3. The high quality solutions (represented by the selected CH set, $\mathcal{C}$) from (RP2) and (RP3) are also good candidate solutions for the original problem (M3-E2), which shows that cut inequalities (5.26) and (5.27) might be strong cuts for model (M3-E2). This is expected since both of the objectives attempt to use the residual energy information at nodes as a routing metric to balance the energy consumption over the network.

The reduced models are useful for two reasons. First, they help generate valid inequalities and improve time performance and alleviate the memory difficulties associated with CPLEX implementation. Second, models (RP2) and (RP3) can provide the high quality solutions. We solve the subproblem (**SubC**) of model (M3-E2) by utilizing the solutions obtained from (RP2) and (RP3), which, in turn, gives good upper bounds for model (M3-E2). Since this is a totally different mechanism for obtaining a good solution from the heuristic search approach mentioned above, combining together will significantly improve the algorithm robustness.

In order to solve model (RP1), we generate a random initial solution and apply `Search`**SubC()** procedure. Then we solve this problem to optimality using CPLEX via incorporating Type I cut inequality (5.20). For a large-size problem, we relax the binary variables z_j^c and x_{ij}^c to flow variables, and solve the associated relaxation problem to optimality using the same approach. Therefore, we obtain the optimal solution (or a lower bound) Z_{LR}^1 in model (RP1) and generate Type II cut inequality

(5.25) for the models (M3-E2), (RP2) and (RP3). To solve models (RP2) and (RP3), we first start with a random initial solution and apply `SearchSubC()` procedure to obtain the heuristic solution for (RP2) and (RP3) respectively. Then, we employ a stopping criteria given by a time limit to solve (RP2) and (RP3) using CPLEX, while incorporating Type I cut inequality (5.21) for (RP2), (5.22) for (RP3) and Type II cut inequality (5.25) for computing efficiency.

V.2.3. Model Relaxation

As the model (M3-E2) is an NP-hard problem, we set out to find good lower and upper bounds. The lower bounds are based on relaxation of single-source constraints x_{ij}^c. Our numerical studies indicate that the effect of relaxing the binary constraint on x_{ij}^c is beneficial in terms of solution characteristics in two ways. First, the optimality gap between the optimal solution and the lower bound is significantly small (within 5%). This can be interpreted as most of the sensors preferably choosing the only one CH to transmit the collected data even without mandatory requirement. Second, the solution of the problem (MR) (represented by the selected CH set, $\mathcal{C}$) exhibits the same or very similar pattern as the optimal solution in the model (M3-E2). This is an important characteristic because we can obtain a good upper bound for model (M3-E2) by utilizing the solution from (MR) and solving the subproblem (**SubC**) of model (M3-E2).

To solve **(MR)**, we use the similar method applied in model **(M3)**, but in a simpler way. First, we generate an initial solution by the construction heuristic and apply `SearchSubC()` to improve the solution. Next, using the best solution from `SearchSubC()` as the input solution, we apply the `SearchSubU()` procedure. Upon the completion of the search procedures, we identify the best solution. Then we employ a stopping criteria given by a time limit to solve **(MR)** using CPLEX

reinforced with objective value based cut inequalities. By doing this, **(MR)** provides both a good lower bound and a high quality solution for **(M3-E2)**.

V.2.4. Parallel Implementation

In this section, we discuss how to parallelize the solution procedure described in the previous sections. Parallel computers have two basic architectures: *distributed memory* and *shared memory* (Cung et al., 2001). Distributed memory parallel computers are essentially a collection of serial computers (nodes) working together to solve a problem. In a shared memory computer, multiple processor units share access to a global memory space via a high-speed memory bus. With a shared-memory multiprocessor, different processors can access the same variables. Message Passing Interface (MPI), is an interface for a set of library functions that processors in a distributed-memory multiprocessor can use to communicate with each other (Pacheco, 1996). In this research, we consider a mixed shared/distributed memory architecture. We parallelize the sequential algorithm via *Master-Worker-Model* which bases on distributed memory architecture. On the other hand, in each subproblem, we solve it using ILOG parallel CPLEX. The CPLEX parallel optimizer is built based on Symmetric Multiprocessor (SMP) shared-memory systems (ILOG, 2006), which takes advantage of multiple processors to solve large linear and difficult mixed integer programs in substantially reduced time.

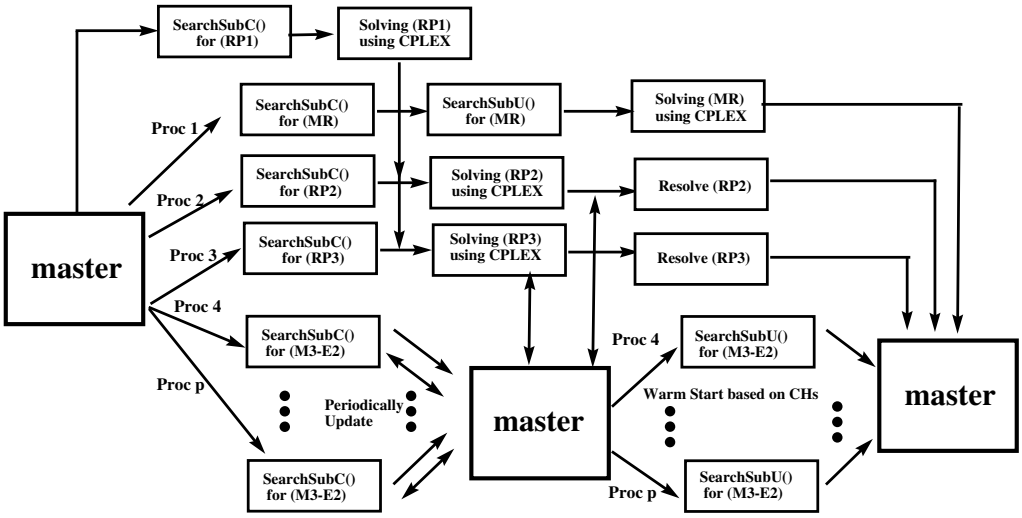
Crainic and Toulouse (2003) have classified the parallelization strategies applied to meta-heuristics into three categories: (1) low-level parallelism; (2) domain decomposition; and (3) multiple search. Low-level parallelism aims solely at speeding up computations by executing in parallel one or several computing-intensive tasks within one or multiple iterations of the method. Domain decomposition is generally implemented by partitioning the vector of decision variables. The partitioning

reduces the size of the solution space, but it needs to be repeated to allow the exploration of the complete space. The first two parallelization strategies yield a single search path. Parallelization approaches that consist of several concurrent searches in the solution space are classified as multiple search. Concurrent searches may or may not execute the same heuristic method, and may start from the same or from different initial solutions. They may share information during the search, which are often called cooperative multi-search methods, or only at the end to identify the best overall solution, in which they are known as independent search methods. For many difficult optimization problems, multiple search method has been implemented and been shown to be effective (Adenso-Díaz et al. (2006); Crainic et al. (2004)).

Our algorithm has three main computing tasks: (1) the heuristic search procedure; (2) cut generation; and (3) model relaxation. Initially, they are performed in a sequential manner. However, we observe that there is a great opportunity to reduce the amount of synchronization by performing the multiple search parallelism. Solving the reduced models (RP1), (RP2) and (RP3) and the relaxed model (MR) can be implemented in the independent processors. As the heuristic search procedure employs a multi-start scheme starting from different initial solutions, it also fits perfectly into multiple concurrent explorations of the solution space.

We use the multiple search parallelism in our algorithm. Specifically, we employ the Master-Worker paradigm where a master process delegates tasks to worker processes, and the workers perform the algorithmic components , then report the results back to the master. The parallel framework is depicted in Figure 11. To begin the algorithm, the master sends initial solutions to p individual worker processes and each worker receives its initial solution, selects a neighborhood and explores it followed by the associated `SearchSubC()` procedure. The master is responsible for managing the overall flow of the heuristic and solving **(RP1)**. Once the procedure for solving

Figure 11: Parallel Framework



(RP1) is completed, the master will send the value of Z_{LR}^1 to worker 2 and 3 for them to generate Type II cut inequality (5.25). The worker process 1 ,2 and 3 are assigned to solve **(MR)**, **(RP2)** and **(RP3)** respectively. Also, worker 2 and 3 will send the values of Z_{LR}^2 and Z_{UR}^3 to master process respectively, once they receive the request from the master. The remaining worker processes are delegated to perform the heuristic search procedure in parallel. Below, we provide the details of how this computing task is parallelized.

As the heuristic search procedure employs a multi-start scheme, each worker (from process 4 to process p) can apply `SearchSubC()` for **(M3-E2)** using different initial solutions. For the search procedure to be effective, it is important that each worker can share the information that can be used to guide the heuristics. To avoid unnecessary information overhead, each worker only passes the best solution obtained to the master periodically. The master then keeps and updates the best solution and send it back to each worker. Once each worker receives the overall best solution, the search is then continued starting from this solution. Upon the completion of the

`SearchSubC()` procedure for worker 4 to worker p, the master updates the current overall best solution so that it can be used as an input solution for `SearchSubU()`. On the other hand, the master will request the values of Z^2_{LR} and Z^3_{UR} from worker processes 2 and 3 as they can be used to generate Type II valid inequalities (5.26) and (5.27). Furthermore, we employ independent search method to solve `SearchSubU()` of **(M3-E2)**. In particular, the master generates different combination of initial solution $\mathcal{D}$ from candidate set of sink nodes, assigns them to each worker (process 4 to p) and broadcasts the current best solution and the values of Z^1_{LR}, Z^2_{LR} and Z^3_{UR} . Using the current best solution as an input solution, each worker then incorporates Type I & II cut inequalities and apply `SearchSubU()` independently.

Upon the completion of parallel implementation, the master will receive four solutions from **(MR)**, **(RP2)** and **(RP3)**, and the heuristic search procedure for **(M3-E2)** respectively. The master, in turn, picks the best one and record the associated objective function value. In doing so, it will significantly help us to exploit information to find good solutions on the various processes.

V.3. Computational Results

To examine the computational performance of the proposed parallel heuristic approach, we carry out a computational study, which is performed on Datastar machine in San Diego Supercomputer Center. The computational environment was a cluster of 176 8-way SMP nodes using IBM POWER4+ chip, where each node has 8 processors and 16GB of memory. The nodes are interconnected by Federation high-performance network switch, support both intranode (shared-memory and MPI) and internode (MPI). In this study, we use 8 nodes where the master process is run on one node, and worker processes are run on the remaining nodes. Each node was coded in C++

and process communications was handled by MPI. Furthermore, we solve the sub-problems using parallel CPLEX 10.2 and use 8 threads to solve one MIP problem during the parallel optimization.

V.3.1. A Comparison of Parallel and Sequential Methods

To verify the effectiveness of the high performance computing technique, we solve a set of instances using both the sequential and parallel method. Note that, unless stated otherwise, all of the input and algorithmic parameter values are set as mentioned in the previous sections. We assume that the initial energy levels at the sensors are equal to 0.5 J and the N is 7 for settings with $|\mathcal{I}|$ equal to 30.

Table 9: A comparison of parallel and sequential Methods

		Sequential Performance			Parallel Performance			
$\|\mathcal{I}\|$	H	T_S^O	T_S^H	$\Delta_S(\%)$	T_{P1}^O (8 threads)	T_{P2}^O (32 threads)	T_P^H	$\Delta_P(\%)$
30	6	917.60	152.84	0.55	131.08	32.60	11.80	0.00
	9	5555.30	258.15	0.42	761.44	268.10	27.30	0.21
	12	3501.60	202.68	0.09	583.50	493.80	36.10	0.09

Table 9 summarizes the computational results using the sequential and parallel methods. The values reported in columns 3 to 9 are averaged over ten instances. The third column (T_S^O) and fourth column (T_S^H) concern the exact solution times (in seconds) using CPLEX and the heuristic solution times using the sequential algorithm, repespectively. The fifth column (Δ_S) represents the optimality gap between the exact solution and the sequential heuristic solution. The sixth column (T_{P1}^O) and the

seventh column (T_{P2}^{O}) concern the solution times for 8 threads and 32 threads using parallel CPLEX, respectively. The eighth column (T_{P}^{H}) represents the parallel heuristic solution times and the ninth column (Δ_P) represents the optimality gap between the exact solution and parallel heuristic solution. Based on the computational results, we can conclude that: (1) Parallel CPLEX significantly improves solution time. (2) Our parallel implementation not only achieves a speed-up of the computations, but also yields better solutions compared with the sequential algorithm. (3) We can generate a larger test-bed and obtain more benchmark results via cyberinfrastructure resources.

V.3.2. Random Test Instance Generation

We generate our random test instances in such a way that a wide range of input data value for the problem parameters is considered. We randomly generate $|\mathcal{I}|$ sensor coordinates uniformly distributed in a square of size $N\,(m)$. Also, the candidate sites for sinks, $\mathcal{K}$, are generated randomly on the periphery of the sensor field. We present our results for a number of problem settings as shown in Table 10. For each problem setting, we consider three levels for H and hence we have 36 different problem classes. For each of problem class, we generate 10 random instances. We set the period length as $T = 4000$ *time-units*; the maximum transmission distance as $r = 30$ m; and the aggregation ratio as $s = 0.3$. In addition, we set the weight $t = 5$ due to some empirical testing. For the energy dissipation related parameters, we set the values similar to the studies in Heinzelman et al. (2000) as $v = 100$ $pJ/bit/m^2$, $w = 50$ nJ/bit, and $c = 5$ nJ/bit. In all of the instances, we assume that the initial energy levels at the sensors are uniformly distributed in the range [0.3, 0.5] J. Note that, our algorithm can be extended to account for nonuniform energy nodes since we use energy-based threshold in the solution approach.

Table 10: Problem setting used in computational testing

<table>
<tr><th>Setting</th><th>$|\mathcal{I}|$</th><th>$|\mathcal{K}|$</th><th>U</th><th>N</th><th>R_i</th></tr>
<tr><td>S1</td><td>30</td><td rowspan="8">8</td><td rowspan="8">2</td><td rowspan="8">50</td><td rowspan="5">10</td></tr>
<tr><td>S2</td><td>40</td></tr>
<tr><td>S3</td><td>50</td></tr>
<tr><td>S4</td><td>60</td></tr>
<tr><td>S5</td><td>70</td></tr>
<tr><td>S6</td><td>80</td><td rowspan="3">9</td></tr>
<tr><td>S7</td><td>90</td></tr>
<tr><td>S8</td><td>100</td></tr>
<tr><td>S9</td><td>150</td><td rowspan="4">16</td><td rowspan="4">3</td><td rowspan="2">75</td><td>8</td></tr>
<tr><td>S10</td><td>200</td><td>7</td></tr>
<tr><td>S11</td><td>250</td><td rowspan="2">100</td><td>6</td></tr>
<tr><td>S12</td><td>300</td><td>5</td></tr>
</table>

V.3.3. Computational Experiments

We consider a single-period setting and evaluate the performance of our algorithm on the basis of solution quality and time via utilizing two different benchmarks:

- Exact solutions

  We obtain the optimal solution for the original model (M3-E2) by using the exact branch-and-cut implementation in parallel CPLEX 10.2 with default parameters. To obtain more benchmark results from (M3-E2), we also solve the problem to optimality with parallel CPLEX, while incorporating two different cuts as described in (5.23), (5.24), (5.25), (5.26) and (5.27).

- Lower bounds from solving the relaxed model (MR).

  As the run time increases exponentially with the instance size, only small instances can be solved to optimality within a reasonable time. We resort to obtain the lower bound from the model (MR) as another benchmark solution

to evaluate the effectiveness of the parallel algorithm.

Our first set of problems is of the size that parallel CPLEX, yields an optimal solution to the test instance within 4 hours of CPU time, while using MIP branch-and-bound routine reinforced with cut generation. Our goal is to see how the results can be influenced by the data structure and to evaluate the quality of heuristic solutions in terms of their deviation from the optimum and from the best lower bound as well as the solution time. The results of these problem sets are summarized in Table 11.

Table 11: Computational results for small-size problems

Setting	H	T^{O}_{ave}	T^{O-C}_{ave}	T^{H}_{ave}	Δ^{O}_{ave}	Δ^{LB}_{ave}
S1	3	7.3	50.1	13.0	0.00	0.03
	4	4.0	41.8	17.8	0.00	0.04
	6	2.4	35.0	10.7	0.00	0.26
S2	4	14.7	116.4	29.5	0.00	0.00
	6	11.6	81.7	22.7	0.00	0.15
	8	13.7	61.3	27.0	0.00	0.13
S3	5	35.8	191.2	40.9	0.00	0.05
	7	899.7	179.2	29.9	0.00	0.09
	10	477.7	100.7	69.6	0.00	0.11
S4	6	892.9	192.4	70.5	0.00	0.02
	9	7201.1	340.5	79.1	0.00	0.05
	12	ns	1371.0	104.1	0.00	0.09
S5	7	162.0	153.0	138.0	0.00	0.03
	10	ns	4398.8	106.5	0.12	0.10
	14	ns	6661.7	102.9	0.08	0.05

We consider the problem sizes varying from 30 to 70 sensor nodes and for each size of the problem, we consider three different levels for parameter H (10%, 15% and 20% of $|I|$) regarding the number of clusters. The values reported in columns 3 to 7 of the table are averaged over the ten instances. The third (T^{O}_{ave}) and the fourth (T^{O-C}_{ave}) columns concern the exact solution times (in seconds) for these instances.

In the former, we report the average solution times for optimality (ns represents the instances are not solved in 4 hours of runtime), whereas in the latter, we report the same measure, however, this time reinforced with cut generation. Clearly, for an instance, the optimum objective function values, Z^O, for both of these cases are the same. However, the solution times are significantly improved for the problem settings $S4$ and $S5$, as observed in column for T_{ave}^{O-C}, if the heuristic solution is obtained first and utilized in solving a model to optimality. The fifth column (T_{ave}^{H}) includes the average solution times obtained using our parallel heuristic described in Section V.2.4. We observe that they are very significantly lower than the solution times for exact solutions. The sixth (Δ_{ave}^{O}) and the seventh (Δ_{ave}^{LB}) columns concern the optimality gaps based on two different benchmarks for these instances. We calculate the optimality gap between the heuristic solution and the optimum as $\Delta_{ave}^{O} = 100 * (Z^H - Z^O)/Z^O$, and the gap between the upper bound and lower bound as $\Delta_{ave}^{LB} = 100 * (Z^H - Z^{LB})/Z^{LB}$ where Z^O , Z^H and Z^{LB} represent the optimal objective function value, heuristic value and lower bound for an instance, respectively. Based on the results in Table 11, we have the following observations:

- Our heuristic solutions can provide near-optimal solutions compared with the exact solutions. For problem settings varying from $S1$ to $S4$, they can actually find the optimal solutions. Even for problem setting $S5$, the optimality gap Δ_{ave}^{O} is significantly small.

- The optimality gap Δ_{ave}^{LB} is also very small for all the problems. This verifies the effectiveness of relaxing the binary constraint on x_{ij}^{c} in terms of solution characteristics. Most of variables x_{ij}^{c} in the relaxation problem (MR) often has integer solutions or yield integer solutions with relatively little branching in a branch-and-bound procedure.

- Cut generation is also effective in terms of alleviating memory difficulties and runtime restrictions.

In general, we can conclude that our heuristic approach provides high-quality solutions with much less runtime than CPLEX. Our second set of problems is focused on larger size. The scenario that we uses has the number of sensor nodes varying from 80 to 300, the number of potential sink locations between 8 and 16, the number of sinks between 2 and 3 and the sensor field size varying from $50 * 50$ to $100 * 100$ m^2 as shown in Table 10. Similarly, we also consider three different H values (10%, 15% and 20% of $|I|$). For these instances, we found that parallel CPLEX (both using default setting and incorporating cut generation) cannot be solved to optimality within 4 hours. Hence we only consider the lower bound for benchmarking. Again, the third column (T^H_{ave}) concerns the average solution times using the parallel heuristic and the fourth column (Δ^{LB}_{ave}) presents the optimality gap between the upper bound and the lower bound.

Table 12 shows the effectiveness of heuristic method for large size problems in terms of solution time and quality. The optimality gap Δ^{LB}_{ave} is significantly small when we solve the problem up to 200 sensor nodes. For problem settings $S11$ and $S12$, the gap is relatively high, but is still within 6%. This is mainly due to the lower bound procedure in section V.2.3. We observe that solving the relaxed model (MR) requires a considerable computational effort for setting $S11$ and $S12$. Therefore we are not able to obtain a tight lower bound when a stopping criterion (given by a time limit) for solving (MR) is employed.

From the results of our performance experiments, we conclude that our approach, is very effective in addressing a complex problem. Parallelism is helpful in solving large size problem in a reasonable amount of time. The main benefit from parallel

implementation is that we are able to run different computing tasks or the same task starting from different initial solutions concurrently and can share the information that helps to guide the overall heuristic method. Also, the algorithm can find high quality solutions for different classes of instances of the same problem, and increase its robustness by utilizing different combinations of strategies and parameter settings.

Table 12: Computational results for large-size problems

Setting	H	T^H_{ave}	Δ^{LB}_{ave}
S6	8	213.5	0.04
	12	230.6	0.06
	16	267.6	0.01
S7	9	288.2	0.03
	14	360.2	0.06
	18	327.5	0.04
S8	10	316.1	0.02
	15	399.1	0.04
	20	387.2	0.02
S9	15	561.4	0.03
	22	542.1	0.03
	30	739.6	0.43
S10	20	1031.6	0.47
	30	1041.4	0.58
	40	1000.4	1.06
S11	25	1101.4	3.47
	35	1130.7	2.08
	50	1090.1	3.40
S12	30	1201.2	5.95
	45	1187.5	2.03
	60	1235.6	4.40

V.4. Summary and Conclusions

In this chapter, we study an integrated topology control and routing problem in WSNs, while incorporating the single-sourcing requirements for CH assignments and explicitly specifying the transmission ranges of sensor nodes for better topology control. We develop a mixed-integer linear programming (MILP) model to determine the sink and cluster-head locations as well as the data flow, over a time horizon. We adopt the objective as the combination of the average energy usage with the range of remaining energy distribution at the sensor nodes.

Our mathematical model dictates discrete optimization formulation, even small size problems are highly impractical to be solved using exact optimization methods. Therefore, we devise an effective parallel heuristic algorithm, to greatest advantage in utilizing the unique model characteristics, cut generation and model relaxation. We consider a mixed shared/distributed memory architecture. We parallelize the sequential algorithm via *Master-Worker-Model* which bases on distributed memory architecture. On the other hand, in each subproblem, we solve it using parallel CPLEX which is shared-memory based.

Based on the results of our performance experiments, we conclude that our approach, which includes various problems and solution characteristics, is very effective in addressing a complex problem. Our parallel implementation not only achieves a speed-up of the computations, but also yields better solutions as it can explore the solution space more effectively. Furthermore, robust implementations can be obtained by the use of different combinations of strategies and parameter settings at each process, leading to high quality solutions for different classes of instances of the same problem, without too much effort in parameter tuning.

CHAPTER VI

DATA VISUALIZATION

Due to the complexity of large-scale WSN problem, we consider visualization tools to support our optimization efforts. Visualization can be very helpful to solve larger and realistic problems with dynamic nature and gain insights into the problem domain. In particular, for the multi-period setting, it would be helpful to see graphically the network status in the beginning of the period (e.g. the energy levels at sensor nodes) and the solution generated (i.e., the underlying network topology including CH and sink locations for a period and the routing schemes over the network).

In this chapter, we aim to use Prefuse Visualization Toolkit (Heer et al., 2005) to develop the methods that can handle visualization tasks for a wide variety of data. Specifically, we combine the *network diagram* and *scatter plot* methods to visualize the multi-period data on sensor network topology and routing. By integrating and analyzing the data, we can better understand and gain intuition regarding the operational characteristics of the algorithms. This will facilitate to find a better solution in a timely manner.

VI.1. Visualization Features

Prefuse is an interactive graphical open source toolkit that supports a rich set of features for data modeling, visualization, and interaction (Heer et al., 2005). Prefuse is written in Java, using the Java 2D graphics library. The data can be stored in a graph or tree structure (interrelated information) or within a data table (not related data). In our case, the data set includes the locations of sensor and candidate sink nodes and also a sequence of files that correspond to the solutions generated at each period, i.e.,CH and sink location assignment for a period and the routing schemes

over the network.

In this chapter, we aim to develop the methods to visualize the network configuration and routing schemes at each period and how they change over the periods. In particular, we consider the following two different methods via using prefuse toolkit.

- Network Diagram

  In a network diagram, entities are connected to each other in the form of a node and link diagram.

- Scatter Plot

  A scatter plot is a classic statistical diagram that visualizes the relationship between numeric variables.

There are pros and cons of these two methods. In the network diagram, the overall arrangement of nodes in the network is very telling of the structure of the connections between nodes. Also, it is easy to see the network status graphically in the beginning of the period. However, this method may suffer from visual confusion caused by the many nodes and interconnections between them. On the other hand, scatter plots can show correlations between features and their interactions in an intuitive and simple way. The limitation for the scatter plot is that, it cannot provide the connections between nodes and routing schemes over the network. Therefore, we combine these two methods so as to better understand and gain intuition regarding the operational characteristics of the algorithm. In the section that follows, we provide the detail descriptions of each method.

VI.1.1. Network Diagram

To make our data amenable for a network diagram, we first transform it to a graph structure. In our study, we adopt the node-link representation. The data includes

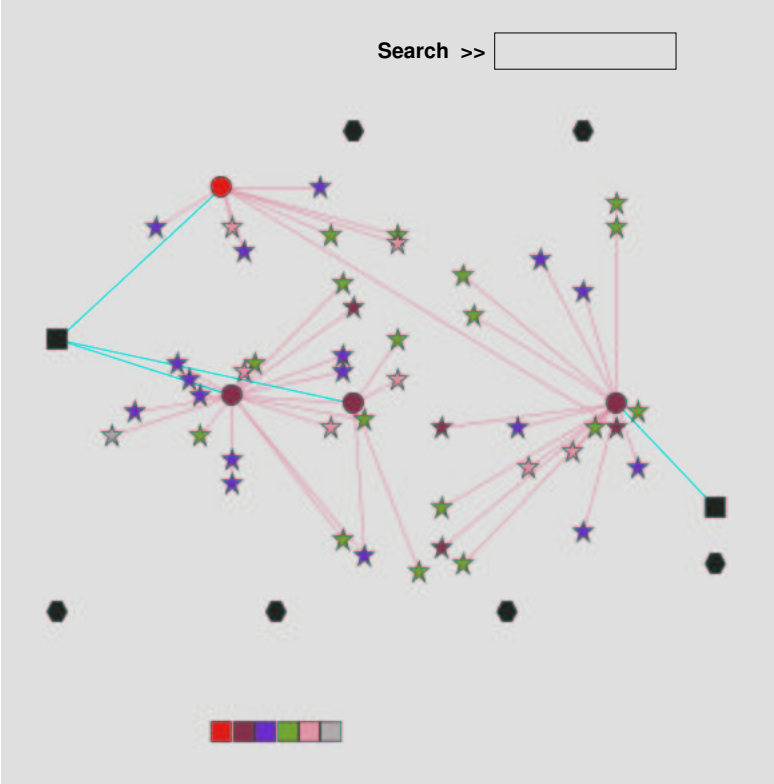
a graph topology and data attributes, which is written in XML format. We define the data attributes as tuples associated with node and link data types. In particular, a node type can be described as "node name ", "node type", "energy level", "x-coordinate", and "y-coordinate"; a link type can have "link type" attribute.

We visualize the data in two-dimensional Euclidean geometry. Specifically, we load a graph data set from a sequence of XML files. Then, we generate the corresponding visual analogues (called VisualItems), which record visual properties such as node name, location and type. Next, individual Renderers for node and link items are created and a default RendererFactory is created to assign these renderers to the appropriate items. Note that, although drawn in two dimensions, network diagram can present many features simultaneously by adding colors, shapes, sizes, textures and so on.

To aid visualization, we include *color maps* for assigning colors to the data elements. As in WSN application, energy efficiency plays a critical role in prolonging network lifetime, we generate the color maps by analyzing the energy level attribute values. Given the sensor energy distribution at each period, we characterize five break points: (1) a – the minimum remaining energy value; (2) b – the average remaining energy value; (3) c – the maximum remaining energy value; (4) a_1 – the energy value is equal to $a * 1.02$; (5) c_1 – the energy value is equal to $c * 0.98$. Furthermore, we define six different energy levels (corresponding to six different colors) based on these 5 break points:

- *Level 1 - gray color*: the node with minimum remaining energy value a;
- *Level 2 - pink*: the node whose remaining energy value is in the interval $(a, a1]$;
- *Level 3 - yellowgreen*: the node whose remaining energy is in the interval $(a1, b]$;
- *Level 4 - purple*: the node whose remaining energy is in the interval $(b, c1)$;

Figure 12: An Example Snapshot of Network Diagram



- *Level 5 - maroon*: the node whose remaining energy is in the interval $[c1, c)$;
- *Level 6 - red*: the node with maximum remaining energy value c.

In addition, we add different *shapes* based on node type. Figure 12 depicts an example snapshot of network diagram. Specifically, the *star* shape represents the regular sensor node, the *eclipse* represents the node selected as a CH, the *rectangle* represents the selected sink node, and the *hexagon* represent the candidate sink. Also, the links are colored depending on different communication linkages. (1) If a sensor sends information to a CH, we have a *pink* link. (2) If a CH sends information to another CH, we have a *brown* link. (3) If a CH sends information to a sink node, we have a *cyan* link.

In node-link visualization applications, we may suffer from visual confusion due to many nodes and interconnections between them. Hence, we embed interactive controls to filter and search among the nodes so that the visualization can be more useful and easy to understand. In particular, we provide controls which includes:

- *Drag*: Click and drag an item to reposition it.
- *Select*: Click an item to place it at the center of the display.
- *Pan*: Left-click and drag the background to pan the display view.
- *Zoom*: Right-click and drag the mouse up or down or use the scroll wheel to zoom the display view.
- *Zoom-To-Fit* : Right-click once to zoom the display to fit the whole graph.
- *Search*: Type in a search box to search for the energy levels over the nodes.

For large network visualization examples, we can zoom and pan to obtain a detailed view of different sections of the graph. Also, we can move the mouse over any node to read the relevant information. In summary, we develop a network diagram whose display features include layout, color maps and interactive controls. It is easy to navigate and use for integrating and analyzing the data.

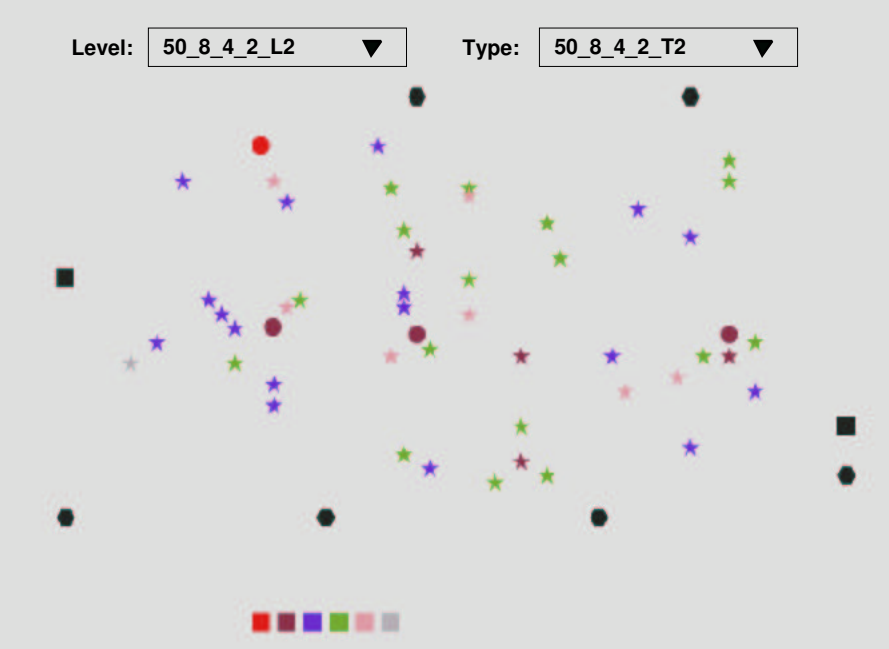
VI.1.2. Scatter Plot

Similar to section VI.1.1, we transform the data to a table structure to make it amenable for a scatter plot method. The input data file is written in txt format. A table organizes a collection of data into rows and columns, each row containing a data record with multiple attributes, and each column containing the corresponding data value for a named attribute. In each row, we define the data attributes as tuples associated with multi-period node characteristics. In particular, each row can have

multiple attributes such as node location ("x-coordinate' and "y-coordinate"), "node type at period 1", "energy level at period 1", "node type at period 2", "energy level at period 2", and so on.

We visualize the data in two-dimensional Euclidean geometry. We first load a data set from a text file and then place the nodes based on their coordinate information provided in the input file. Note that, for each node, we adopt the same color and shape features as section VI.1.1.

Figure 13: An Example Snapshot of Scatter Plot



In the scatter plot visualization, it is difficult to manipulate data set interactively. Therefore, we develop the user interface that allows visual encodings to be changed at different periods via placing two drop-down menus in the layout. By doing so, we can present the node energy levels (represented as different colors) and the node types (represented as different shapes) for each period. Figure 13 depicts an example snapshot of scatter plot at period 2. The name "50_8_4_2_L2", in the first drop down menu, represents "50 sensor nodes, 8 candidate sinks, 4 CHs and 2 sinks, energy level at period 2". Similarly, "50_8_4_2_T2", in the second drop down menu, represents "50 sensor nodes, 8 candidate sinks, 4 CHs and 2 sinks, node type at period 2". The drop down list gives us options such as "50_8_4_2_L1", "50_8_4_2_T1", "50_8_4_2_L5", "50_8_4_2_T5". The powerful part of this visualization is that we can present various algorithm solutions in one simple form, by selecting the energy level and node type for any period from the drop down list. Hence it allows us to study the relationship among data points and how they change over the periods. A scatter plot is a simple approach, but it demonstrates how compelling it could be for data exploration.

VI.2. Multi-period Data Analysis

In order to gain insights into the operational characteristics of the model, we solve model **(M3)** under the setting given by $|\mathcal{I}| = 50$, $|\mathcal{K}|$=8, U=2, N=50, R_i=10, and H=4 values. The initial energy levels at the sensors are uniformly distributed in the range [0.1, 0.5] *J*. We optimally solve model **(M3)** using exact branch-and-cut implementation in CPLEX with default parameters. We use both the *network diagram* and *scatter plot* methods to visualize the multi-period solutions obtained from CPLEX on sensor network topology and routing. Figure 14 and 15 depict the first four period

Figure 14: Different Period Snapshots of Network Diagram

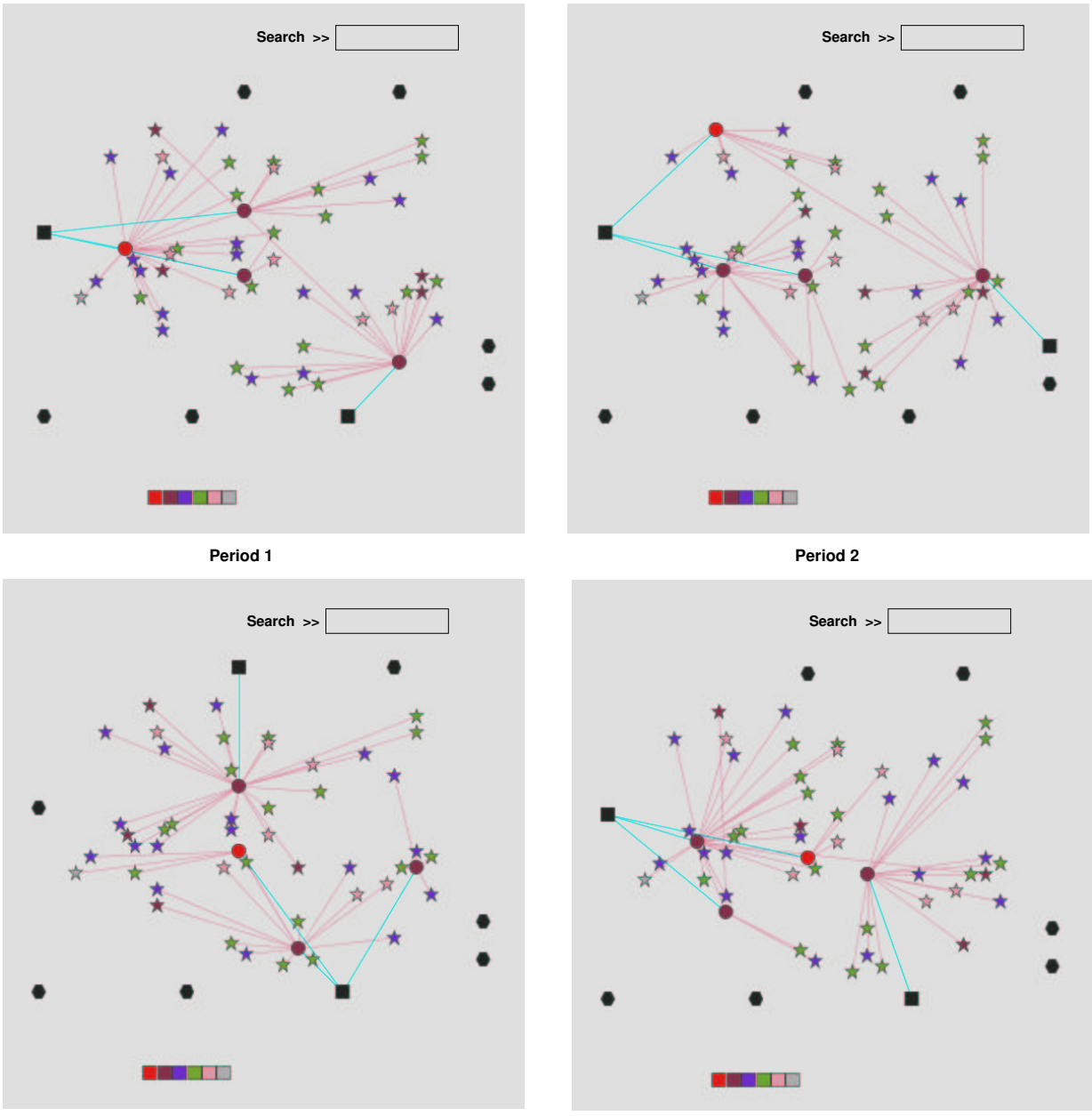
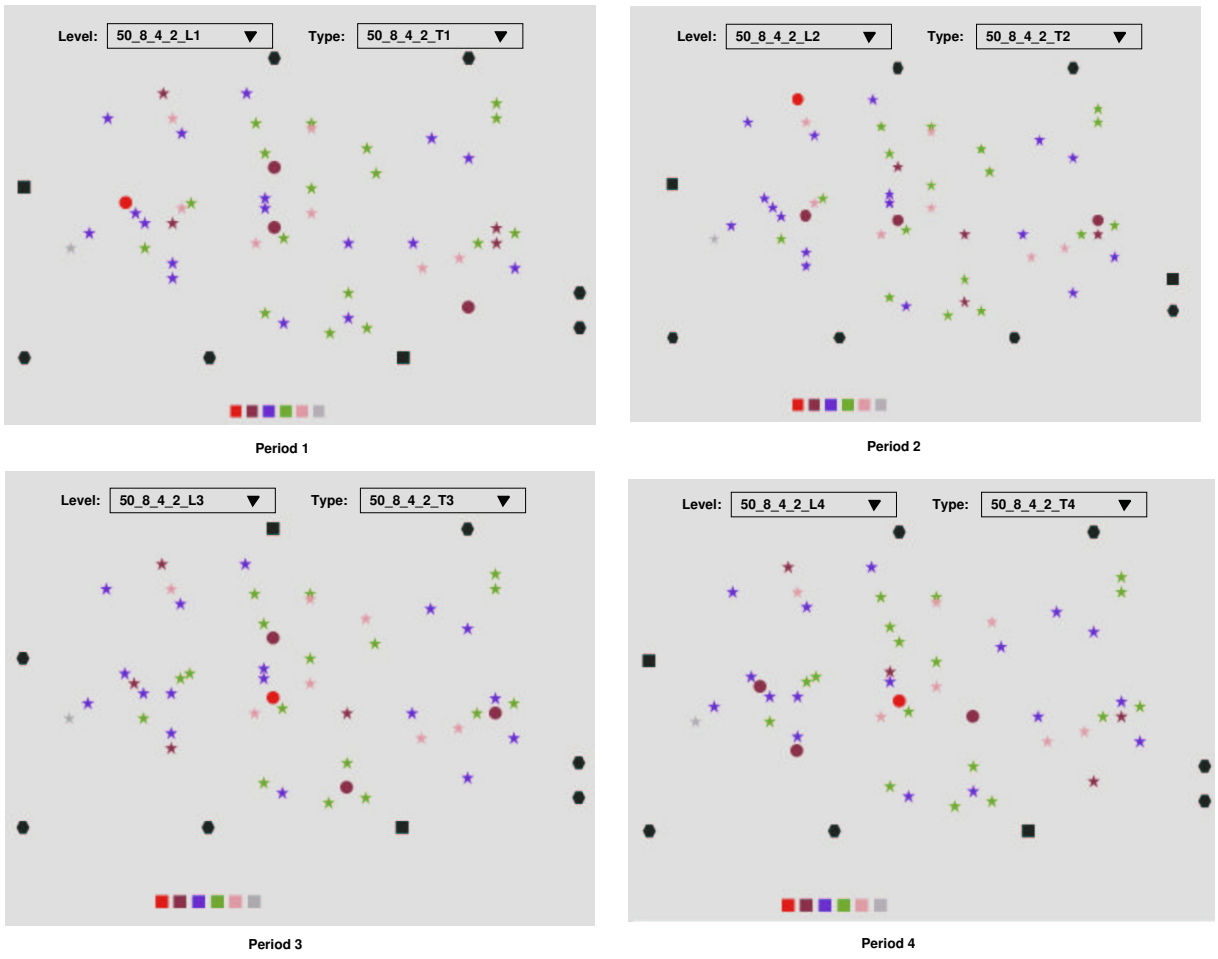


Figure 15: Different Period Snapshots of Scatter Plot



snapshots using the network diagram and scatter plot methods, respectively.

By integrating and analyzing these data, we have the following important observations:

1. The energy efficient topology where some CHs are selected close to the sinks and the remaining CHs are chosen from the center of the sensor field, are commonly observed at different periods. Furthermore, we note that the sensor nodes with higher energy (level 5 & 6) and also close to the periphery are strong candidates for CHs; the sink nodes are selected in a well spread fashion. This will help us develop a simple rule or a localized algorithm on how to choose the CH and sink locations across the network.

2. The network topology changes from period to period, which results in a balance energy dissipation. As illustrated in figure 14 and 15, the CH and sink locations and the routing schemes vary significantly in successive periods.

3. The sensor nodes with higher-energy (level 5 & 6) are more likely selected as CHs. This verifies the effectiveness of our heuristic algorithm proposed in section III.5. By considering only a subset of sensors with higher-energy as the set of candidate CHs, we can significantly reduce the number of variables and constraints in model **(M3)** and hence facilitate to find a better solution in much less time.

4. As shown in figure 14 and 15, the solutions generated from **(M3)** do not necessarily give the equal size clusters. This is an illustration of a pitfall in using the equal size cluster, as it may not promote a balanced energy consumption pattern.

5. The solutions generated from **(M3)** do not necessarily select the most H high-

est energy nodes as CHs. This observation points out the deficiency in some studies (Younis and Fahmy, 2004), as they mainly attempt to choose the highest energy sensors as CHs. This may be biased from the long-term network lifetime perspective.

6. In model **(M3)**, we consider the multi-sourcing assignments, i.e., each sensor can choose more than one CHs to transmit the collected information. As illustrated in figure 14 and 15, most of the sensors preferably choose the only one CH to transmit the collected data even without mandatory requirement. Based on the discussion in chapter V, organizing sensor networks into disjoint clusters may ensure scalability so that the network performance does not significantly degrade with the increase of the network size. This is an important finding as it demonstrates that our solutions generated from **(M3)** facilitate the system scalability even without explicitly considering the single-sourcing assignments.

VI.3. Summary and Conclusions

In this chapter, we aim to utilize the data visualization techniques to support our optimization efforts. We use Prefuse Visualization Toolkit (Heer et al., 2005) to develop the methods that can handle visualization tasks for a wide variety of data. Specifically, we combine the *network diagram* and *scatter plot* methods to visualize the multi-period data on sensor network topology and routing. By integrating and analyzing the data, we gain intuition regarding the operational characteristics of the algorithms and verify the effectiveness of our heuristic algorithm proposed in section III.5. Furthermore, we demonstrate that visualization can be very helpful in facilitating to find a better solution in a timely manner.

CHAPTER VII

CONCLUSIONS AND FUTURE DIRECTIONS

Recent advancements in sensory devices are presenting various opportunities for widespread applications of sensor networks. Wireless sensor networks (WSNs) can be deployed in inhospitable environments and difficult-to-reach terrains such as urban or rural battlefields, borderlines, forest fires; in wild habitats and oceans to monitor and observe natural phenomena; in disaster prevention and relief; in ecological and environmental monitoring. The most challenging issue in WSNs is limited and unrechargeable energy provision. Many research efforts aim at developing energy efficient network topology and routing schemes for prolonging the network lifetime. However, we notice that, in the majority of the literature, topology control and routing problems are handled separately, thus overlooking the interrelationships among them. In particular, if the issues of routing are not taken into consideration in the topology control problem, then the underlying topology might not be suited for supporting an efficient routing scheme.

Considering this deficiency in the current literature, we investigate the integrated mathematical models and their solution algorithms for topology control and routing, along with consideration of multiple sinks, based on optimization techniques for the design of WSNs. To this end, we consider three models that differ mainly in terms of their objective functions. The objectives include minimization of (1) total energy usage in the system, (2) maximum energy used at a sensor node, and (3) a weighted sum of the range of end-of-period remaining energy distribution at the sensor nodes and the average energy used in the system. We determine that the third model captures important characteristics of topology control and routing integration in WSN design and exhibits significantly better performance than the first two models and a

well-known protocol HEED in extending network lifetime.

VII.1. Summary of Contributions

Our contributions in this dissertation can be summarized as follows.

1. We devise three mathematical models for integrated topology and routing decisions for data-gathering WSNs. The first two objectives are commonly considered in devising communication protocols (e.g. Heinzelman et al. (2000, 2002); Lloyd et al. (2005); Ramanathan and Hain (2000); Rodoplu and Meng (1999); Wattenhofer et al. (2001)). However, this has not been done from an integrated mathematical modelling perspective as in our case. We consider these two models as benchmark models for our third proposed model, which minimizes the total energy and the range of remaining energy distribution in the network. The models developed herein provide new insights in the theory of WSNs and introduce new modelling approaches for important practical problems in this developing area.

2. In devising our models, we consider the use of multiple sinks. This is helpful for energy efficiency since multiple sinks create an opportunity for better proximity to sensors, thus saving energy in communication. It is possible to route the data so that the energy drainage in the network is more evenly distributed to the sensors by changing the locations of the sinks and the CHs in each period.

3. We suggest a new approach to achieve topology control via limiting the usable energy at a sensor as a fraction of its total available energy. We show that how this usable energy determined is important and difficult in our two benchmark models. On the other hand, the solution in the proposed third model is in-

sensitive to this characteristic as the control of energy distribution is implicitly accounted in the objective.

4. We consider cases where an overall view of a measure, such as spatial and temporal temperature/humidiy/pressure gradients in a large sensor field deployed for environmental monitoring, is of interest. To this end, we employ a general data aggregation approach at the CHs that represents the elimination of data redundancy. In previous studies with data aggregation (e.g. Heinzelman et al. (2000, 2002); Kalpakis et al. (2003); Younis and Fahmy (2004)), we observe that aggregation of data into a single signal at each CH, i.e., regardless of the amount of data received, is common which is applicable in such cases as monitoring maximum temperature in the sensor field.

5. We consider two important extension models to the setting of the third model, by incorporating the fixed CH set-up cost (as in the first case), and the single-sourcing requirements for CH assignments and the transmission ranges of sensor nodes (as in the second case). On the methodology side, we develop effective solution approaches that are based on Benders decomposition techniques, heuristics and parallel heuristic algorithms. The proposed solution approaches will also contribute to the generalized mixed-discrete optimization problem, especially for the problems with similar characteristics.

6. The data visualization toolkit developed herein will be very helpful in solving larger and realistic problems with dynamic nature.

VII.2. Foundation for Future Research

Research in this dissertation can be extended in several directions in the future, as outlined below.

1. One extension of our work, from the modeling perspective, is to incorporate the coverage problem into the integrated topology control and routing problems, i.e., we exploit the high spatial redundancy of sensors by only allowing a subset of sensors active for a given period of time, whereas all other sensors save energy being in inactive state.

2. This dissertation study focuses on time-driven sensor networks applications pertaining to continuously monitoring ecological habitats (animals, plants, microorganisms). Another interesting extension of our work, from the modeling perspective, is to reformulate the models to suit for the time critical applications.

3. The solution methods that we have developed in this dissertation may also apply to the generalized mixed-discrete optimization problem, such as the capacitated p-center problem, the multicommodity network design problem and the multiobjective optimization problem.

4. For data visualization, we plan to provide additional components and potentially develop a visual environment for animating a time-series of values from a single aggregated input file and displaying real-time data.

REFERENCES

Abbasi, A., M. Younis. 2007. A survey on clustering algorithms for wireless sensor networks. *Computer Communications* **30**(14-15) 2826–2841.

Adenso-Díaz, B., S. García-Carbajal, S. Lozano. 2006. An empirical investigation on parallelization strategies for scatter search. *European Journal of Operational Research* **169** 490–507.

Akkaya, K., M. Younis. 2005. A survey on routing protocols for wireless sensor networks. *Journal of Ad Hoc Networks* **3**(3) 325–349.

Akyildiz, I. F., W. Su, Y. Sankarasubramaniam, E. Cayirci. 2002a. A survey on sensor networks. *IEEE Communications Magazine* **40**(8) 102–114.

Akyildiz, I. F., W. Su, Y. Sankarasubramaniam, E. Cayirci. 2002b. Wireless sensor networks: A survey. *Computer Networks* **38**(3) 393–422.

Al-Karaki, J. N., A. E. Kamal. 2004. Routing techniques in wireless sensor networks: A survey. *IEEE Wireless Communications* **11**(6) 6–28.

Al-Karaki, J. N., R. Ul-Mustafa, A. E. Kamal. 2009. Data agregation and routing in wireless sensor networks: Optimal and heuristic algorithms. *Computer Networks* **53**(7) 945–960.

Alfieri, A., A. Bianco, P. Brandimarte, C.F. Chiasserini. 2007. Maximizing system lifetime in wireless sensor networks. *European Journal of Operational Research* **181** 390–402.

Bandyopadhyay, S., E. J. Coyle. 2003. An energy efficient hierarchical clustering algorithm for wireless sensor networks. *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. San Franciso, CA 1713–1723.

Benders, J. F. 1962. Partitioniing procedures for solving mixed-variables programming problems. *Numerische Mathematik* **4** 238–252.

Bhardwaj, M., A. P. Chandrakasan. 2002. Bounding the lifetime of sensor networks via optimal role assignments. *Proceedings of the IEEE INFOCOM*. New York 1587–1596.

Biagioni, E., K. Bridges. 2002. The application of remote sensor technology to assist the recovery of rare and endangered species. *International Journal of High Performance Computing Applications* **16**(3) 315–324.

Biagioni, E., G. Sasaki. 2002. Wireless sensor placement for reliable and efficient data collection. *Proceedings of the 36th Hawaii International Conference on Systems Sciences*. Big Island, HI.

Bulusu, N., J. Heidemann, D. Estrin. 2000. GPS-less low cost outdoor localization for very small devices. Technical Report 00–729, Department of Computer Science, University of Southern California.

Burri, N., P. von Rickenbach, R. Wattenhofer. 2007. Dozer: Ultra-low power data gathering in sensor networks. *ACM/IEEE Fourth International Symposium on Information Processing in Sensor Networks (IPSN'07)*. Cambridge, Massachusetts 450–459.

Cardei, M., D. Du. 2005. Improving wireless sensor network lifetime through power aware organization. *ACM Wireless Networks* **11**(3) 333–340.

Cerpa, A., J. Elson, D. Estrin, L. Girod, M. Hamilton, J. Zhao. 2001. Habitat monitoring: Application driver for wireless communications technology. *In Proceedings of the 2001 ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean*. San Francisco, CA.

Chang, J., L. Tassiulas. 2004. Maximum lifetime routing in wireless sensor networks. *IEEE/ACM Transactions on Networking* **12**(4) 609–619.

Chang, J.-H., L. Tassiulas. 2000. Energy conserving routing in wireless ad hoc networks. *Proceedings of the IEEE INFOCOM*. Tel Aviv, Israel 22–31.

Chen, G., C. Li, M. Ye, J. Wu. 2009. An unequal cluster-based routing protocol in wireless sensor networks. *Wireless Networks* **15**(2) 193–207.

Cheng, Z., M. Perillo, W. B. Heinzelman. 2008. General network lifetime and cost models for evaluating sensor network deployment strategies. *IEEE Transactions on Mobile Computing* **7**(4) 484C497.

Chong, C. Y., S. P. Kumar. 2003. Sensor networks: Evolution, opportunities, and challenges. *Proceedings- IEEE* **91**(8) 1247–1256.

Clouqueur, T., V. Phipatanasuphorn, P. Ramanathan, K. K. Saluja. 2003. Sensor deployment strategy for detection of targets traversing a region. *Mobile Networks and Applications* **8**(4) 453–461.

Collins, S. L., L. M. A. Bettencourt, A. Hagberg, R. F. Brown, D. I. Moore, G. Bonito, K. A. Delin, S. P. Jackson, D. W. Johnson, S. C. Burleigh. 2006. New opportunities

in ecological sensing using wireless sensor networks. *Frontiers in Ecology and the Environment* **4**(8) 402–407.

Crainic, T. G., M. Gendreau, P. Hansen, N. Mladenović. 2004. Cooperative parallel variable neighborhood search for the p-median. *Journal of Heuristics* **10** 293–314.

Crainic, T. G., M. Toulouse. 2003. Parallel strategies for metaheuristics. *In: Handbook of Metaheuristics*. Glover, F., Kochenberger, G. (Eds.), Kluwer, Boston. 475–513.

Cung, V. D., S. L. Martins, C. C. Ribeiro, C. Roucariol. 2001. Strategies for the parallel implementation of metaheuristics. *In: Essays and Surveys in Metaheuristics*. Ribeiro, C.C., Hansen, P. (Eds.), Kluwer, Boston. 263–308.

Datta, S., T. Woody. 2007. Eight technologies for a green future. *Business 2.0 Magazine* .

Estrin, D., G. Borriello, R. Colwell, J. Fiddler, M. Horowitz, W. J. Kaiser. 2001. *Embedded, Everywhere: A Research Agenda for Networked Systems of Embedded Computers*. National Academy Press, Washington DC.

Gandham, S. R., M. Dawande, R. Prakash, S. Venkatesan. 2003. Energy efficient schemes for wireless sensor networks with multiple mobile base stations. *IEEE Global Telecommunications Conference, GLOBECOM* **1** 377–381.

Geoffrion, A.M., G.W. Graves. 1974. Multicommodity distribution system design by Benders decomposition. *Management Science* **20**(5) 822–844.

Haenggi, M. 2003. Energy-balancing strategies for wireless sensor networks. *IEEE International Symposium on Circuits and Systems (ISCAS03)*. Bangkok, Thailand.

Hart, J. K., K. Martinez. 2006. Environmental sensor networks: A revolution in the earth system science? *Earth-Science Reviews* **78**(3-4) 177–191.

Heer, J., S. K. Card, J. A. Landay. 2005. Prefuse: A toolkit for interactive information visualization. *Conference on Human Factors in Computing Systems (CHI)*. Portland, OR.

Heinzelman, W. 2000. Application-specific protocol architectures for wireless networks. Ph.D. Dissertation, Massachusetts Institute of Technology.

Heinzelman, W. B., A. Chandrakasan, H. Balakrishnan. 2000. Energy efficient communication protocol for wireless micro-sensor networks. *Proceedings of IEEE Hawaii International Conference on System Sciences*. Hawaii. 174–185.

Heinzelman, W. B., A. Chandrakasan, H. Balakrishnan. 2002. An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on Wireless Communications* **1**(4) 660–670.

Heo, N., P. Varshney. 2005. Energy-efficient deployment of intelligent mobile sensor networks. *IEEE Transactions on Systems, Man and Cybernetics, Part A,* **35**(1) 78–92.

Howard, A., M.J. Matarić, G. Sukhatme. 2002. An incremental self-deployment algorithm for mobile sensor networks. *Autonomous Robots* **13**(2) 113–126.

Hua, C., T. P. Yum. 2008. Optimal routing and data aggregation for maximizing lifetime of wireless sensor networks. *IEEE/ACM Transactions on Networking* **16** 892–903.

ILOG. 2006. *ILOG CPLEX 10.0 users manual*. ILOG, Sunnyvale, California.

Kalpakis, K., K. Dasgupta, P. Namjoshi. 2003. Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks. *Computer Networks* **42**(6) 697–716.

Karl, H., A. Willig. 2005. *Protocols and Architectures for Wireless Sensor Networks*. John Wiley and Sons, West Sussex, England.

Khan, M., B. Bhargava, L. Lilien. 2003. Self-configuring clusters, data aggregation, and authentication in microsensor networks. Technical Report. CSD TR 03-003, Department of Computer Science, Purdue University.

Krishnamachari, B. 2005. *Networking Wireless Sensors*. Cambridge University Press, New York.

Kubisch, M., H. Karl, A. Wolisz, L. C. Zhong, J. Rabaey. 2003. Distributed algorithms for transmission power control in wireless sensor networks. *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*. New Orleans 558–563.

Lin, S., B. W. Kernighan. 1973. An effective heuristic algorithm for the traveling salesman problem. *Operations Research* **21** 498–516.

Liu, J., B. Li. 2003. Distributed topology control in wireless sensor networks with asymmetric links. *IEEE Global Telecommunications Conference, GLOBECOM* **3** 1257– 1262.

Liu, M., J. Cao, G. Chen, X. Wang. 2009. An energy-aware routing protocol in wireless sensor networks. *Sensors* **9**(1) 445–462.

Lloyd, E. L., R. Liu, M. V. Marathe, R. Ramanathan, S. S. Ravi. 2005. Algorithmic aspects of topology control problems for ad hoc networks. *Mobile Networks and Applications* **10**(1-2) 19–34.

Madan, R., S. Cui, S. Lall, A. J. Goldsmith. 2007. Modeling and optimization of

transmission schemes in energy-constrained wireless sensor networks. *IEEE/ACM Transactions on Networking* **15**(6) 1359–1372.

Magnanti, T.L., R.T. Wong. 1981. Accelerating Benders decomposition: Algorithmic enhancement and model selection. *Operation Research* **29**(3) 464–484.

Mainwaring, A., J. Polastre, R. Szewczyk, D. Culler. 2002. Wireless sensor networks for habitat monitoring. *In First ACM International Workshop on Wireless Workshop in Wireless Sensor Networks and Applications (WSNA 2002)*. Atlanta, GA.

Nakamura, F. G., F. P. Quintao, G. C. Menezes, G. R. Mateus. 2005. An optimal node scheduling for flat wireless sensor networks. *Proceedings of IEEE International Conference on Networking (ICN05)*. 475 – 483.

Ordónez, F., B. Krishnamachari. 2004. Optimal information extraction in energy-limited wireless sensor networks. *IEEE Journal on Selected Areas in Communications* **22**(6) 1121–1129.

Pacheco, P. 1996. *Parallel Programming With MPI*. Morgan Kaufmann Publishers Inc., San Francisco, CA.

Papadimitriou, I., L. Georgiadis. 2006. Energy-aware routing to maximize lifetime in wireless sensor networks with mobile sink. *Journal of Communications Software and Systems* **2**(2) 141–151.

Pottie, G. J., W. J. Kaiser. 2000. Wireless integrated network sensors. *Communications- ACM* **43**(5) 51–58.

Raghavendra, C. S., K. M. Sivalingam, T. Znati. 2006. *Wireless Sensor NetworkS*. Springer-Verlag, New York.

Ramanathan, R., R. Hain. 2000. Topology control of multihop wireless networks using transmit power adjustment. *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*. Tel-Aviv, Israel 404–413.

Rodoplu, V., T.H. Meng. 1999. Minimum energy mobile wireless networks. *IEEE Journal on Selected Areas in Communications* **17**(8) 1333 – 1344.

Roy, T.J.V. 1986. A cross decomposition algorithm for capacitated facility location. *Operation Research* **34**(1) 145–163.

Rundel, P. W., E. A. Graham, M. F. Allen, J. C. Fisher, T. C. Harmon. 2009. Environmental sensor networks in ecological research. *New Phytologist* **182**(3) 589–607.

Saffo, P. 1997. Sensors: The next wave of innovation. *Communications- ACM* **40**(2) 92–97.

Santi, P. 2005a. *Topology Control in Wireless Ad Hoc and Sensor Networks*. John Wiley and Sons, West Sussex, England.

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

Savvides, A., C-C. Han, M. B. Strivastava. 2001. Dynamic fine-grained localization in ad-hoc networks of sensors. *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*. Rome, Italy 166–179.

Singh, S., M. Woo, C. S. Raghavendra. 1998. Power-aware routing in mobile ad hoc networks. *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking*. Dallas, TX 181–190.

Sohraby, K., D. Minoli, T. F. Znati. 2007. *Wireless Sensor Networks: Technology, Protocols, and Applications*. John Wiley and Sons, New Jersey, NJ.

Soro, S., W. B. Heinzelman. 2009. Cluster head election techniques for coverage preservation in wireless sensor networks. *Ad Hoc Networks* **7**(5) 955–972.

Technology Review. 2003. Ten emerging technologies that will change the world. *MIT's Technology Review* **106**(1) 33–49.

Tian, D., N. D. Georganas. 2005. Connectivity maintenance and coverage preservation in wireless sensor networks. *Ad Hoc Networks* **3**(6) 744–761.

Tilak, S., N. B. Abu-Ghazaleh, W. R. Heinzelman. 2002. A taxonomy of wireless micro-sensor network models. *ACM SIGMOBILE Mobile Computing and Communications Review* **6**(2) 28–36.

Tseng, Y., Y. Chang, B. Tzeng. 2004. Energy-efficient topology control for wireless ad hoc sensor networks. *Journal of Information Science Engineering* **20**(1) 27–38.

Üster, H., G. Easwaran, E. Akcali, S. Cetinkaya. 2007. Benders decomposition with alternative multiple cuts for a multi-product closed-loop supply chain network design mode. *Naval Research Logistics* **54**(8) 890–907.

Wang, Q., H. Hassanein. 2005. A comparative study of energy-efficient (E^2) protocols for wireless sensor networks. *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*. Ilyas, M., Mahgoub, I. (Eds.), CRC Press, Boca Raton, FL. 18:1– 18:22.

Wang, Q., H. Hassanein, K. Xu. 2005. A practical perspective on wireless sensor networks. *Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems*. Ilyas, M., Mahgoub, I. (Eds.), CRC Press, Boca Raton, FL. 9:1– 9:28.

Wattenhofer, R., L. Li, P. Bahl, Y. M. Wang. 2001. Distributed topology control for power efficient operation in multihop wireless ad hoc networks. *20th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. Anchorage, Alaska 1388–1397.

Wentges, P. 1996. Accelerating Benders' decomposition for the capacitated facility location problem. *Mathematical Methods of Operations Research* **44**(2) 267–290.

West Technology Research Solution. 2010. The WTRS wireless sensor network technology trends Q1 2010, single issue. http://MarketResearch.com.

Xing, G., C. Lu, Y. Zhang, Q. Huang, R. Pless. 2005. Minimum power configuration in wireless sensor networks. *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc)*. Urbana-Champaign, IL 390–401.

Xue, Y., Y. Cui, K. Nahrstedt. 2005. Maximizing lifetime for data aggregation in wireless sensor networks. *Mobile Networks and Applications* **10** 853–864.

Ye, F., A. Chen, S. Liu, L. Zhang. 2001. A scalable solution to minimum cost forwarding in large sensor networks. *Proceedings of 10th International Conference on Computer Communications and Networks*. Hong Kong 304 –309.

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

Younis, O., S. Fahmy. 2004. Heed: A hybrid, energy-efficient, distributed clustering approach for ad-hoc sensor networks. *IEEE Transactions on Mobile Computing* **3**(4) 366–379.

Zhang, H., J. C. Hou. 2005. Maintaining sensing coverage and connectivity in large sensor networks. *Ad Hoc & Sensor Wireless Networks* **1**(1-2) 89 – 124.

Zou, Y., K. Chakrabarty. 2003. Sensor deployment and target localization based on virtual forces. *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*. San Francisco, CA.

VITA

Hui Lin received two B.S. degrees in automation and engineering economics in 2001 from Tianjin University, China and her M.S. in electrical engineering from Tianjin University in 2004. She joined the Department of Industrial and Systems Engineering at Texas A&M University in Fall 2005 for doctoral studies and received her Ph.D. in December 2010. Her research interests include wireless sensor network design, applied optimization and high performance computing.

Hui Lin can be reached at:

Department of Industrial & Systems Engineering

241 Zachry, 3131 TAMU

College Station, TX 77843-3131