Stony Brook University

Design and Optimization on Mobile Data Gathering in Wireless Sensor Networks

A Dissertation Presented

by

Miao Zhao

to

The Graduate School

in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

in

Electrical Engineering

Stony Brook University

December 2010

Stony Brook University

The Graduate School

Miao Zhao

We, the dissertation committee for the above candidate for the Doctor of Philosophy degree, hereby recommend acceptance of this dissertation.

Yuanyuan Yang – Dissertation Advisor
Professor, Department of Electrical and Computer Engineering

Sangjin Hong – Chairperson of Defense
Associate Professor, Department of Electrical and Computer Engineering

Dantong Yu
Adjunct Professor, Department of Electrical and Computer Engineering

Jie Gao
Assistant Professor, Department of Computer Science

This dissertation is accepted by the Graduate School.

Lawrence Martin
Dean of the Graduate School

Abstract of the Dissertation

Design and Optimization on Mobile Data Gathering in Wireless Sensor Networks

by

Miao Zhao

Doctor of Philosophy

in

Electrical Engineering

Stony Brook University

2010

Wireless sensor networks (WSNs) have emerged as a new information-gathering paradigm for taking spatial and temporal measurements of a given set of real-word parameters. In these applications, sensors monitor the environment and route their sensing data back to a static data sink. As the routing task depends purely on sensors themselves, the sensors near the sink need to relay much more packets than the sensors far away from the sink. As the result, it would incur substantial and non-uniform energy consumption among sensors. Therefore, how to efficiently aggregate the information from scattered sensors, generally referred to as *data gathering*, is an important and challenging issue in WSNs as it largely determines network lifetime. Recent studies have shown that significant benefit can be achieved in WSNs by employing mobile collectors for data gathering in WSNs via short-range communications. In such kind of mobile data gatherings, the mobile collectors

roam over the sensing field with controlled mobility, perform appropriate actions to schedule data collection, and transport data back to the data sink, while sensors are engaged in sensing task and only need to relay data for local aggregation if necessary. In this way, energy can be greatly saved at sensors as mobile collectors fully or partially take the burden of routing away from sensors.

This dissertation focuses on scheme design and performance optimization of mobile data gathering in WSNs. We address several important issues and propose a suite of algorithms to improve data gathering performance. First, we consider utilizing spatial-division multiple access (SDMA) to achieve concurrent data uploading from multiple sensors to the mobile collector. The moving tour of the mobile collector is determined based on the tradeoff between the shortest moving path and full utilization of SDMA among sensors. This joint design can lead to prolonged network lifetime as well as shortened data gathering latency. Second, we extend such joint design of mobility and SDMA technique to large sensor network with multiple mobile collectors. A region division and tour planning algorithm is proposed to balance the data gathering time among different regions. Third, we explore inherent tradeoff between energy saving and data gathering latency by proposing bounded relay hop mobile data gathering. In this scheme, multi-hop relay for local data aggregation is incorporated into mobile data gathering, while the relay hop count is constrained to a certain level to limit energy consumption at sensors. Fourth, we optimize the mobile data gathering performance by characterizing the data gathering strategies as a pricing mechanism, where sensors independently adjust their payment for the data uploading opportunity to the mobile collector based on the shadow prices set by the mobile collector. Fifth, we study the problem of how to achieve optimal performance of mobile data gathering based on a flow-level network model. We jointly consider data rate control at sensors, multi-hop routing for data transmissions, and sojourn time allocation for the mobile collector. We propose distributed algorithms to implement these strategies so as to achieve system-wide optimum. Finally, we propose joint design of mobile en-

ergy replenishment and mobile data gathering in wireless rechargeable sensor networks. The mobile entity plays not only as a data collector but also as an energy transporter to deliver energy to sensors via wireless energy transmissions. We present distributed algorithms to provide timely energy recharge to maintain perpetual network operations, meanwhile achieving high-performance data gatherings.

To my beloved husband, Chao, my parents, my sister and my parents-in-law for their love and support.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to show my gratitude to numerous people, who helped and supported me during my Ph.D. study.

First and foremost, I am heartily thankful to my supervisor, Prof. Yuanyuan Yang, for her invaluable guidance and strong support throughout the course of dissertation. She aroused my interests and led me into the networking research field. Her enlightening vision, deep insight, creative thinking, and extensive experience are essential to make my networking research smooth and enjoyable. We have closely worked for more than five years. The high level of professionalism that she demonstrates in her teaching and research will continue to serve as a role model for me in my future career development.

I would also like to thank my defense committees, Prof. Sangjin Hong and Prof. Dantong Yu in Department of Electrical and Computer Engineering, and Prof. Jie Gao in Department of Computer Science. Thanks for their precious time and useful suggestions to improve my dissertation quality.

I am grateful to my colleagues Dr. Zhenghao Zhang, Dr. Ming Ma, Dr. Chi Ma, Dr. Deng Pan, Dr. Min Yang, Lin Liu, Xi Deng, Ji Li, Zhexi Pan, Dawei Gong, Zhiyang Guo, Zhemin Zhang, and Cong Wang in Mobile Computing Laboratory for their friendship and all of their help over years.

I would also like to express my gratitude to many individuals in the department who have made my stay at Stony Brook pleasant and memorable. Staff Assistant for Department's Chairman, Deborah Kloppenburg, and Graduate Program Coordinator, Rachel Ingrassia, have been especially helpful.

Finally and specially, I would like to deeply thank my dear husband, Chao Liang, who has always been a great source of strength to me. He is always my greatest love, my truest friend, and my closest companion. I am forever indebted to my parents Yinglin Zhao and Jianhua Wang for their unyielding love, endless pa-

tience and great encouragement. The achievements of my dissertation are not possible without their tremendous support. I am grateful to my sister, my brother-in-law, and my dear nephew, who always bring much happiness to me. I also appreciate my parents-in-law for their love and encouragement in these years.

Publications

Journal Publications

- M. Zhao and Y. Yang, "Optimization based distributed algorithms for mobile data gathering in wireless sensor networks," under submission.
- M. Zhao, D. Gong and Y. Yang, "Cost minimization for mobile data gathering in wireless sensor networks," under submission.
- M. Ma, Y. Yang and M. Zhao, "A single hop data gathering mechanism with mobile collectors for wireless sensor networks," under submission.
- C. Liang, M. Zhao and Y. Liu, "Optimal bandwidth sharing in multi-swarm multi-party P2P video conferencing systems," under submission.
- M. Zhao and Y. Yang, "Packet scheduling with joint design of MIMO and network coding," under submission.
- M. Zhao and Y. Yang, "Bounded relay hop mobile data gathering in wireless sensor networks," to appear in *IEEE Transactions on Computers*, 2010.
- M. Zhao, M. Ma and Y. Yang, "Efficient data gathering with mobile collectors and space-division multiple access technique in wireless sensor networks," to appear in *IEEE Transactions on Computers*, 2010.
- Z. Zhang, Y. Yang and M. Zhao, "Enhancing downlink performance in wireless networks by simultaneous multiple packet transmission," *IEEE Transactions on Computers*, vol. 58, no. 5, pp. 706-718, 2009.
- M. Zhao, M. Ma and Y. Yang, "Applying opportunistic medium access and multiuser MIMO techniques in multi-channel multi-radio WLANs," *ACM/Springer*

Mobile Networks and Applications (MONET), vol. 14, no. 4, pp. 486-507, August 2009.

- M. Zhao, Y. Yang, H. Zhu, W. Shao and V. O. K. Li, "Priority-based opportunistic medium access control in IEEE 802.11 WLANs," *International Journal of Sensor Networks (IJSN)*, vol. 3, no. 2, pp. 84-94, 2008.

Conference Publications

- M. Zhao, J. Li and Y. Yang, "Joint mobile energy replenishment and data gathering in wireless rechargeable sensor networks," under submission.
- M. Zhao and Y. Yang, "A framework for mobile data gathering with load balanced clustering and MIMO uploading," accepted by *The 30th IEEE International Conference on Computer Communications (INFOCOM)*, April 2011.
- M. Zhao and Y. Yang, "An optimization based distributed algorithm for mobile data gathering in wireless sensor networks," *The 27th IEEE International Conference on Computer Communications (INFOCOM) mini-conference*, March 2010.
- M. Zhao, D. Gong and Y. Yang, "A cost minimization algorithm for mobile data gathering in wireless sensor networks," *The 7th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE MASS)*, Nov. 2010.
- D. Gong, M. Zhao and Y. Yang, "A multi-channel cooperative MIMO MAC protocol for wireless sensor networks," *The 7th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE MASS)*, Nov. 2010.
- M. Zhao and Y. Yang, "Data gathering in wireless sensor networks with multiple mobile collectors and SDMA technique," *IEEE Wireless Communications and Networking Conference (IEEE WCNC)*, April 2010.
- D. Gong, M. Zhao and Y. Yang, "Joint channel assignment and space-division multiple access scheduling in wireless mesh networks," *IEEE Wireless Communications and Networking Conference (IEEE WCNC)*, April 2010.

- M. Zhao and Y. Yang, "Packet scheduling with joint design of MIMO and network coding," *The Sixth IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE MASS)*, Oct. 2009.

- M. Zhao and Y. Yang, "Bounded relay hop mobile data gathering in wireless sensor networks," *The Sixth IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE MASS)*, Oct. 2009.

- M. Zhao, M. Ma and Y. Yang, "Mobile data gathering with space-division multiple access in wireless sensor networks," *The 27th IEEE International Conference on Computer Communications (INFOCOM)*, April 2008.

- M. Zhao, M. Ma and Y. Yang, "Mobile data gathering with multiuser MIMO technique in wireless sensor networks," *IEEE Global Communications Conference (IEEE GLOBECOM)*, Nov. 2007.

- M. Zhao, M. Ma and Y. Yang, "Applying opportunistic medium access and multiuser MIMO techniques in multi-channel multi-radio WLANs," *The 4th IEEE International Conference on Broadband Communications, Networks, and Systems (IEEE BroadNets)*, Sept. 2007.

- M. Zhao, M. Ma and Y. Yang, "Opportunistic medium access control in MIMO wireless mesh networks," *The 20th International Tele-traffic Congress (ITC-20)*, Ottawa, Canada, June 2007.

- M. Zhao and Y. Yang, "A joint design of MIMO-OFDM transceiver and power-saving MAC WLANs," *IEEE Wireless Communications and Networking Conference (IEEE WCNC)*, Hong Kong, March 2007.

- M. Zhao, Z. Zhang and Y. Yang, "Medium access diversity with uplink-downlink duality and transmit beamforming in multiple-antenna wireless networks," *IEEE Global Communications Conference (IEEE GLOBECOM)*, Nov. 2006.

- M. Zhao, H. Zhu, W. Shao, V. O. K. Li and Y. Yang, "Contention-based prioritized opportunistic medium access control in wireless LANs," *IEEE International Conference on Communications (ICC)*, June 2006.

- H. Zhu, V. O. K. Li, Z. Ma and M. Zhao, "Statistical connection admission control framework based on achievable capacity estimation," *IEEE International Conference on Communications (ICC)*, June 2006.
- M. Zhao, H. Zhu, V. O. K. Li and Z. Ma, "A stability-based link state updating mechanism for QoS routing," *IEEE International Conference on Communications (ICC)*, May 2005.

Chapter 1

Introduction

This chapter explains the motivation, design goals, challenges, and contributions of the dissertation.

1.1 Motivation

Wireless sensor networks (WSNs), composed of densely-deployed, low-cost, low-power, multifunctional sensors, have emerged as a new information-gathering paradigm for taking spatial and temporal measurements of a given set of parameters, such as temperature, sound, atmospheric pressure, humidity, or pollutants of a field [1]. They can be used in a wide range of applications, including industrial process control, machine health monitoring, environment and habitat surveillance, healthcare applications, home automation, and traffic control [2]-[8].

In a sensor network, sensors are usually randomly deployed over a field without a pre-configured infrastructure. Each sensor has the capabilities of monitoring the environment, collecting data and routing data back to a data sink [1]. Since sensors are typically battery-powered and has limited capacity, energy consumption becomes a primary concern in a WSN, as it is crucial for the network to functionally operate for an expected period of time. Typically, most energy of a sensor is consumed on two major tasks: sensing the field and uploading data to the data sink. Energy consumption on sensing is relatively stable since it only depends on the sampling rate. However, the situation of energy consumption on data uploading is much more complicated than that of sensing. Data uploading costs significant

amount of energy at sensors for wireless transmissions and the energy expenditure is typically non-uniform among sensors. It strongly depends on the network topology and the location of the destined data sink. As a result, the energy of the sensors near the sink is depleted much sooner than others since these sensors need to relay much more packets from the sensors far away from the sink. Therefore, how to efficiently aggregate the information from scattered sensors, generally referred to as *data gathering*, is an important and challenging issue in WSNs as it largely determines network lifetime.

Due to tremendous practical interests, in recent years, much research effort has been devoted to efficient data gathering in WSNs and many schemes have been proposed. In early research, efficient relay routing [9]-[14] or hierarchical infrastructure [17]-[20] are employed to improve the routing efficiency. For the relay routing, data packets are forwarded via multi-hop relays among sensors. Some other factors, such as load balance, schedule pattern and data redundancy, are jointly considered with the routing scheme. The successful relay routing requires connectivity among sensors. And the common feature of these approaches is that the sensors on the critical pathes would deplete their energy faster than others, which lead to the limited network lifetime. A WSN can also be organized into a hierarchical infrastructure instead of flat topology, in which sensors are grouped into clusters and the cluster heads take the responsibility of forwarding data to an outside data sink [17]-[20]. It was shown that the hierarchical infrastructure is an efficient way to handle the scaling issue in WSNs. However, in such hierarchical networks, cluster heads inevitably consume more energy than other sensors. To avoid "hot spots", sensors can become cluster heads rotationally [17]. Since every sensor may possibly become a cluster head, each of them has to be "powerful" enough to handle incoming and outgoing traffic, which increases the overall cost of the network. Furthermore, it may incur high overhead due to frequent information exchange among sensors. To overcome these problems in static networks, in more recent work, mobile data gathering [22]-[37] is introduced. In such schemes, a special type of mobile nodes (usually called mobile collectors) are used for facilitating connectivity among static sensors. The typical scenario of mobile data gathering can be depicted as shown in Fig. 1.1, where a mobile collector roams over the sensing field and moves close enough to the sensors for data collection. In this way, mobile collector takes over the burden of routing from sensors, which is particularly desirable when sensors

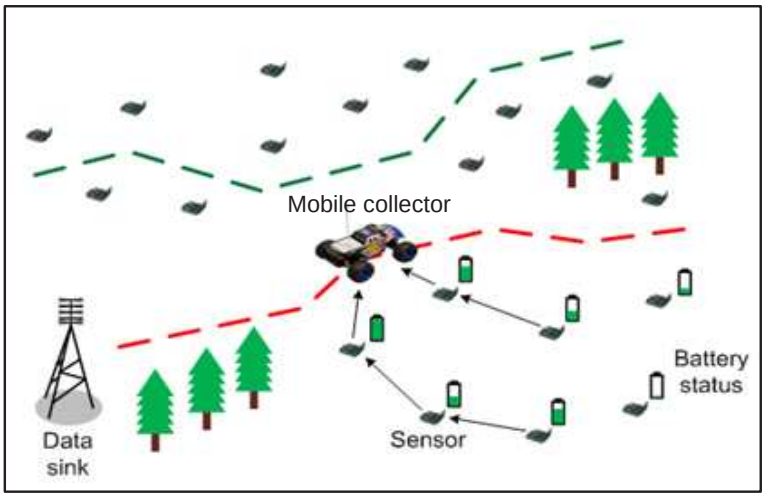


Figure 1.1: Illustration of mobile data gathering.

have limited energy and storage buffer. In general, there are three advantages that make mobility perfectly suitable to data gathering applications in WSNs. First, it alleviates the non-uniformity of energy consumption among sensors since each sensor sends data to the mobile collector via short-range communication when it comes close enough. Second, it works well not only in a connected network, but also in a disconnected network. The moving path of a mobile collector acts as virtual bridge linking up the separated subnetworks, thus the network coverage and connectivity would no longer be a serious problem in packet forwarding. Third, when the possible locations for the mobile collector to stay for data collection are known, its moving tour turns to be predictable, which provides an opportunity to dynamically find an optimal tour to actually achieve efficient data gathering.

Although existing mobile data gathering schemes [22]-[37] can greatly save the energy at sensors, they typically result in an increased data gathering latency. To shorten the latency, most of the work mainly focuses on mobility control to minimize the moving time and rarely takes the sensor behaviors into consideration. Moreover, little work can be found in the literature to properly characterize and model the performance of mobile data gathering. The impact of some system parameters on network performance is still unveiled at current stage. These observations motivate us to propose novel scheme design and optimization strategies to further enhance the performance of mobile data gathering in WSNs.

1.2 Design Goals and Research Challenges

Compared with static data gathering via routing among sensors, mobile data gathering has more service requirements and challenges as we need to consider the behaviors both of the mobile collector and the sensors, as well as coordinating the communication between them.

We now introduce the systematic design goals of the mobile data gathering applications.

- **Prolonged Network Lifetime:** As mobile collectors effectively alleviate the routing burden of sensors, it is extremely expected to prolong the network lifetime as much as possible since sensor batteries typically can not be recharged or replaced after initial deployment. If the rechargeable sensor network available, besides extending network lifetime, it is desirable to maintain the perpetual operations of the network.

- **Shortened Data Gathering Latency:** The data gathering latency is referred to as the time duration of a data gathering tour, which incudes the mobile collector departing from the static data sink, moving along a tour, sojourning at specified locations to collect data, and finally returning to the data sink. Different data application may have different requirement on the time sensitivity. Generally speaking, it is expected to achieve short data gathering latency in order to obtain and process the sensing data as fast as possible.

- **High Network Utility:** Network utility is a properly defined function to characterize the data gathering performance, which quantifies the aggregated "value" of the gathered data from different sensors in a data gathering tour. In practice, the "value" measure can be in terms of information entropy or revenue, which provides the flexibility of modeling user application needs, or a level of "satisfaction" on a certain amount of data from each sensor. Network utility is a direct metric to evaluate the effectiveness of the mobile data gathering. Clearly, higher network utility is always preferred.

- **Low Network Cost:** Network cost is a properly defined function, which is used to quantify the aggregated cost on gathering data from sensors when the mobile collector moves over different locations in a data gathering tour. The

"cost" here physically implies the energy consumption or monetary expense on gathering a certain amount of data from a sensor at a particular location. Network cost is a direct metric to evaluate the efficiency of the mobile data gathering. It is expected to lower the network cost as much as possible for obtaining certain volume of data from the sensing field.

Given the unique characteristics, what we have to handle includes, but not limited to, the following open technical challenges and issues in the scheme design and performance optimization for mobile data gathering in WSNs.

- **Limited Energy at Sensors**. Sensor batteries are generally with limited capacity, which can not be recharged or replaced after initial deployment in the conventional sensor networks. Even in the rechargeable sensor networks, the recharging rate is quite low and the renewable power sources are not always available for use. Therefore, how to save energy at sensors to sustain functional operations is one of the most critical issues in WSNs.

- **High Latency**. As we have indicated, mobile data gathering commits itself to saving energy at sensors by making the mobile collector move close enough to sensors for short-range communication. This would adversely prolong the data gathering latency compared to pure routing among sensors. This prolonged time is due to two reasons, one is the low moving velocity of mobile collector, the other is because of the low efficiency of transmission schedule among the sensors. Therefore, how to lower the data gathering latency and balance the tradeoff between energy saving and latency is an interesting and open issue in mobile data gathering.

- **Network Dynamics**. As the mobile collector moves over different locations in the sensing field, the network topology actually changes from time to time. Due to the mobility, each sensor has the options to use different rate and any set of possible routes to reach the mobile collector at different locations for data uploading. Such versatility makes the settings of data gathering strategies much more complicated than that of static data gathering. How to adjust data rate, link schedule and routing path to achieve the system-wide optimum in terms of network utility or network cost is an interesting and promising research topic.

- **Flexible Mobility and Multi-functionality of Mobile Collector**. Since the mobile collector has the freedom to move to different locations over the sensing field. Different moving trajectory planning would make great impact on the achievable data gathering performance. Moreover, it is possible for the mobile collector to play multiple roles in the sensing field. For example, in the rechargeable sensor networks, the mobile collector can also serve as an energy transporter to deliver energy for the sensors who need the renewable power supply. In this way, mobile energy replenish and mobile data gathering can be executed simultaneously.

1.3 Contributions

In this dissertation, we would focus on the topic of mobile data gathering in WSNs and address several important issues in it. Our contributions can be summarized as follows.

- **Efficient Data Gathering with Mobile Collectors and Space-Division Multiple Access Technique in WSNs** [108]-[110]. We employ a mobile collector, which works like a mobile base station in the sensing field and polls each sensor for data collection while traversing their transmission range. We also consider applying spatial-division multiple access (SDMA) technique to data gathering by equipping the mobile collector with two antennas. By employing SDMA, two distinct compatible sensors may successfully make concurrent data uploading to the mobile collector, which cuts the data uploading time into half in the ideal situation. In this work, we find a tradeoff between the shortest moving tour of the mobile collector and the full utilization of SDMA among sensors so as to minimize the total time cost including the moving time and data uploading time. Therefore, our design can achieve prolonged network time as well as shortened data gathering latency.

- **A Region Division and Tour Planning Algorithm for Mobile Data Gathering with Multiple Mobile Collectors and SDMA Technique** [111]. We extend the joint design of mobility and SDMA technique to the case of employing multiple mobile collectors, in which the sensing field is divided into several non-overlapping regions, each having a mobile collector. We focus

on minimizing the maximum time of a data gathering tour among different regions over the sensing field. Our results show that our proposed algorithm with two available mobile collectors can achieve at least 56% time saving with respect to the non-SMDA+single mobile collector scheme.

- **Bounded Relay Hop Mobile Data Gathering in WSNs** [113][114]. In order to explore the inherent tradeoff between the energy saving and data gathering latency in mobile data gathering, we propose a mechanism named bounded relay hop mobile data gathering (BRH-MDG). In BRH-MDG, we incorporate multi-hop relay into mobile data gathering, while the relay hop count is constrained to a certain level to limit the energy consumption at sensors. Specifically, a subset of sensors will be selected as the polling points that buffer the locally aggregated data and upload the data to the mobile collector when it arrives. In the meanwhile, when sensors are affiliated with these polling points, it is guaranteed that the local relaying of any packet is bounded within a given number of hops. We provide two efficient algorithms to select the polling points among the sensors. It is observed in our simulations that when the local relays are required to complete within two hops for all sensors, our proposed algorithms can result in at least 38% shorter tour length on average compared to the single hop data gathering scheme with shortened moving tour.

- **A Pricing-based Distributed Algorithm to Minimize Network Cost for Mobile Data Gathering in WSNs** [115][116]. We study how to optimize the performance of anchor-based range traversing data gathering scheme, where a mobile collector roams over the sensing field and sojourns at some locations, called anchor points, on its moving tour to directly collect data from each sensor in a single hop. We formulate the performance optimization as a cost minimization problem constrained by the channel capacity, the minimum amount of data gathered from each sensor, and the bound of total sojourn time at all anchor points. This global problem can be decomposed into two subproblems to be solved by each sensor and the mobile collector, respectively. We show that such decomposition can be characterized as a pricing mechanism, in which each sensor independently adjusts its payment for the data uploading opportunity to the mobile collector based on the shadow prices of

different anchor points.

- **Distributed Network Utility Maximization Algorithms for Mobile Data Gathering in WSNs** [117][118]. We study the problem on how to achieve the optimal performance of the anchor-based mobile data gathering, where the mobile collector gathers data from nearby sensors via multi-hop transmission at each anchor point. We formalize the problem as network utility maximization problem under the constraints of guaranteed network lifetime and bounded data gathering latency. To achieve this objective, we jointly address the issues of data rate control, flow routing, and sojourn time allocation, which critically affect the data gathering performance. To efficiently solve these issues, we correspondingly decompose the utility maximization problem into several subproblems and solve them in a distributed manner, which facilitates the scalable implementation of the algorithms. Extensive numerical results demonstrate that the algorithms can achieve fast convergence with the variables nearly reaching their optimal values after only 100 iterations.

- **Joint Mobile Energy Replenishment and Data Gathering in Wireless Rechargeable Sensor Networks** [119]. In this work, we extend our study to wireless rechargeable sensor networks. In order to provide steady and high recharging rates, and achieve efficient data gathering simultaneously, we propose to utilize mobility for the joint design of energy replenishment and data gathering. In particular, a multi-functional mobile entity, is employed, which serves not only as a data collector that roams over the field to gather data via short-range communication but also as an energy transporter that charges static sensors on its migration tour via wireless energy transmissions [65][66]. Our proposed design can guarantee the perpetual operations of the network and achieve high-utility data gatherings.

We conclude the tasks, design goals and contributions of this dissertation as shown in Table 1.1. Our work combines algorithm design, mathematical modeling, performance optimization, analysis and simulation techniques to conduct comprehensive studies on the above issues. The proposed research will have a significant impact on fundamental design principles and infrastructures for the development of future sensor networks. The outcome of these work can be applicable to a wide

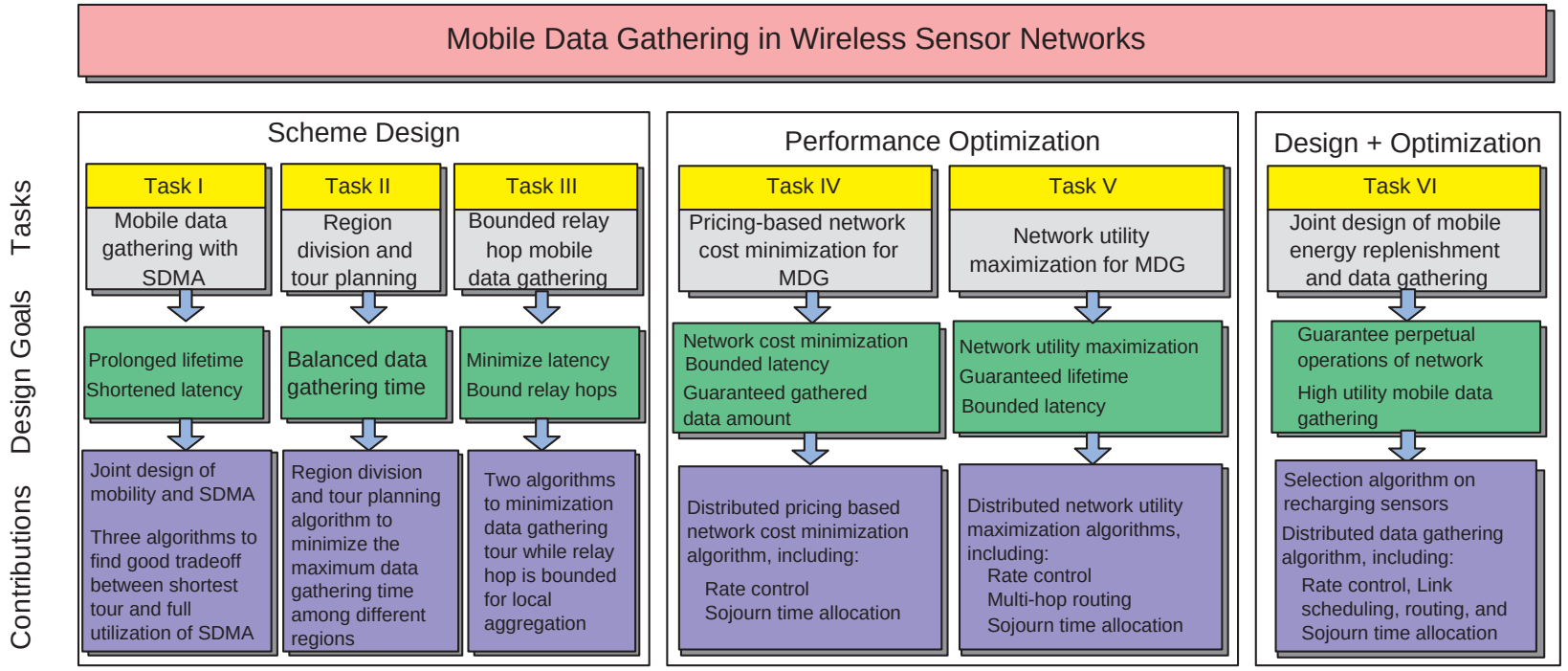


Table 1.1: The tasks, design goals and contributions of this dissertation.

spectrum of applications, including environmental monitoring, field surveillance, human-unattended exploration, industry control, and other commercial areas.

1.4 Dissertation Outline

The rest of the dissertation is organized as follows. Chapter 2 proposes three efficient algorithms with the joint design of mobility and SDMA technique for mobile data gathering. Chapter 3 extends the joint design of mobility and SDMA to the case of multiple mobile collectors. A region division and tour planning algorithm is presented, in which data gathering time is balanced among different regions. Chapter 4 proposes bounded relay hop mobile data gathering scheme and gives two efficient algorithms to implement the design. Chapter 5 studies the cost minimization problem of mobile data gathering and correspondingly proposes a pricing-based distributed algorithm to achieve the minimum network cost. Chapter 6 sets up a flow-level model to characterize the performance of mobile data gathering and presents distributed algorithms to achieve optimal network utility. Chapter 7 proposes a joint design of mobile energy replenishment and mobile data gathering in wireless rechargeable sensor networks, which not only provides system-wide optimal data gathering performance but also achieves perpetual operations of the network. Finally, Chapter 8 concludes the dissertation.

The mobile collectors used for data gathering could be mobile robots or vehicles equipped with powerful transceivers and batteries. For convenience of presentation, we simply call them *SenCars* in the rest of this dissertation.

Chapter 2

Efficient Mobile Data Gathering with Space-Division Multiple Access Technique in WSNs

This chapter presents a joint design of mobility and spatial-division multiple access (SDMA) technique for mobile data gathering in WSNs. The mobility we refer to here is to deploy a mobile collector, i.e., a SenCar, in a sensing field, which works like a mobile base station and collects data from sensors via single hop transmissions so as to achieve uniform energy consumption. We also consider applying SDMA technique to data gathering by equipping the SenCar with multiple antennas such that distinct compatible sensors may successfully make concurrent data uploading to the SenCar. To investigate the utility of the joint design of controlled mobility and SDMA technique, we formulate this design into an integer linear problem (ILP), named *mobile data gathering with SDMA*, or *MDG-SDMA* for short, which aims to minimize the total data gathering time including the moving time of the SenCar and the data uploading time of sensors. Correspondingly, we propose three efficient algorithms to provide practically good solutions to the problem. Extensive simulations demonstrate that our proposed algorithms can result in at least 35% savings on the data gathering time compared to the non-SDMA algorithm with minimum additional overhead.

The rest of this chapter is organized as follows. Section 2.1 summarizes the existing work in this area. Section 2.2 discusses the basic principles of SDMA

technique. Section 2.3 provides the formulation of MDG-SDMA problem and Section 2.4 presents three algorithms to solve it. Section 2.5 gives extensive simulation results that reveal the impact of the joint design of controlled mobility and SDMA technique on network performance. Finally, Section 2.6 concludes the chapter.

2.1 Introduction and Related Work

Due to tremendous practical interests, in recent years, much research effort has been devoted to mobile data gathering in WSNs and some schemes have been proposed [22]-[37]. In such schemes, one or multiple mobile collectors are introduced to take over the routing burden from static sensors. Shah, et al. [22] exploited a type of mobile collectors, called data mules, with random mobility in sparse sensor networks. Data mules pick up data from nearby sensors, buffer the data and then drop them off to a wired access point. This scheme substantially reduces the amount of energy consumption at sensors, but its random moving trajectory is difficult to manage and packet delay cannot be controlled. In [24] and [25], public transportation vehicles were adopted as mobile collectors. In [26], Jea, et al. further proposed a scheme in which data mules move along parallel straight lines and collect data from nearby sensors with multi-hop transmissions. This scheme works well in uniformly distributed sensor networks. However, it may not be necessary or possible for data mules to move only along straight lines. To obtain more flexible data gathering tours for mobile collectors, Ma and Yang [27] proposed a moving path planning algorithm by a divide and conquer method, which recursively determines the turning point for load balancing and organizes each part of the network into a cluster. More recently, they further proposed a single-hop data gathering scheme [28], in which a mobile collector pauses at certain locations to gather data from sensors in the proximity via single-hop transmissions. Zhao, et al. [29] considered mobility control and developed algorithms to generate ferry routes that meet traffic demand and minimize weighted packet delay. Somasundara, et al. [30] proposed an algorithm for scheduling mobile elements to ensure no data loss due to buffer overflow. Ekici, et al. [31] gave an offline heuristic algorithm, which computes periodic trajectories of mobile elements based on the knowledge of data generation rate of sensors and their locations to avoid data loss at low mobile speeds. Luo and Hubaux [32] studied how routing can be fine-tuned to leverage the trajectory of the mobile collector, in par-

ticular, how to better exploit transmission capabilities of the nodes located at the periphery of the network. Nakayama, et al. [33] presented a data gathering scheme, where sensors are clustered and the migration route of the mobile sink is found by an approximate solution to Traveling Salesman Problem (TSP) among cluster centers. Nesamony, et al. [34] studied the minimum distance route problem by letting the transmission range of the calibrating mobile sink along its traversal touch all sensors and defining the determination of the mobile route as a problem belonging to the class of TSP. Basagni, et al. [35] provided a Mixed Integer Linear Programming (MILP) analytical model whose solution determines the sink trajectories that maximize network lifetime, and proposed a distributed heuristic scheme, in which the mobile element moves towards the area where nodes have the highest residual energy. Xing, et al. [36] proposed a rendezvous design to minimize the distance in multi-hop routing paths for local data aggregation under the constraint that the tour length of the mobile collector is no more than a threshold. Finally, Dantu, et al. provided a hardware and software design of a mobile robotic testbed in [37] and experimentally validated some data gathering applications in mobile environments.

Although the aforementioned mobile data gathering schemes can greatly save energy at sensors, they typically result in significantly increased data gathering latency. To overcome this problem, most existing work mainly focused on minimizing the moving length of the mobile collector and did not take sensors' data uploading time into account. In practice, the data uploading time could be a significant part of total data gathering time, especially in a densely-deployed WSN where the time cost on data uploading from sensors to mobile collectors could be comparable to or even more than the moving time. These observations motivate us to design a scheme that can optimize the total data gathering time including both the moving time of the mobile collector and the data uploading time of sensors.

Besides mobility, we also adopt an advanced physical layer technique in wireless communications, SDMA technique, for data gathering. SDMA belongs to the category of multiuser multiple-input and multiple-output (MIMO) technology, specifically with multiple receive antennas [87]. By equipping multiple antennas and specific filters at the receiver, SDMA makes it possible for multiple senders to simultaneously transmit data to a receiver. SDMA was originally used in wireless local area networks (WLANs) and cellular networks [48][49], and sometime was also combined with orthogonal-frequency division multiplexing (OFDM) to

improve the channel capacity and tackle the difficulties of limited available bandwidth. In our study, we have observed that SDMA matches well with the communication pattern in sensor networks since the prominent feature of data gathering traffic in a WSN is many-to-one, where data need to be converged to a collector from multiple sensors. To elaborate, if each SenCar is equipped with two antennas and each sensor still has a single antenna, two compatible sensors can make concurrent data uploading to the SenCar by utilizing SDMA technique when the SenCar arrives at their proximity. The SenCar will separate the multiplexed signal upon receiving and successfully decode the distinct information from different sensors. As a result, the data uploading time would be cut into half in the ideal situation. Since data uploading time is part of the total data gathering time, applying SDMA technique to data gathering would lead to dramatically shortened latency.

In this work, we focus on the joint design of mobility and SDMA technique for mobile data gathering with the purpose of minimizing data gathering time. In particular, we deploy a SenCar with controlled mobility for a short moving tour that alleviates and balances the energy consumption among sensors, and at the same time, utilize SDMA technique to efficiently schedule data transmissions so as to shorten the data uploading time. The main contributions of this work can be summarized as follows. (1) Consider the data uploading time, which was largely ignored in existing work, as part of the total data gathering time, and apply SDMA technique to shorten it, while most existing work [28, 36] was only concerned with minimizing the moving time of the SenCar. (2) Introduce a joint design of controlled mobility and SDMA technique for data gathering, and characterize it as MDG-SDMA problem, respectively. (3) Formulate the MDG-SDMA problem into an integer linear program (ILP) and prove its NP-hardness. Propose three algorithms to solve this problem. (4) Carry out extensive simulations to validate the efficiency of proposed algorithms. The results demonstrate that the proposed algorithms can reduce the data gathering time by at least 35% in a densely-deployed sensor network compared with the non-SDMA algorithm.

2.2 SDMA: Linear Decorrelator Strategy

In this section, we briefly explain the principles of SDMA technique. In the literature [87], the use of multiple receive antennas in the uplink is often called SDMA. In

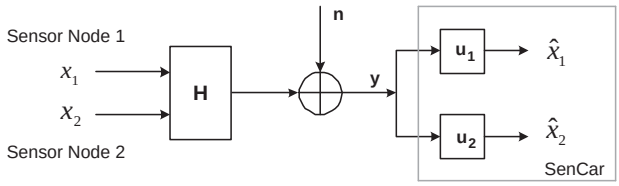


Figure 2.1: SDMA with linear decorrelator strategy.

the application of mobile data gathering in WSNs, a SenCar is the receiver equipped with multiple antennas and sensors acting as the source nodes are the senders, each having a single antenna to upload sensing data to its associated SenCar. We will mainly consider the case when the SenCar is equipped with two antennas, because it is not hard to mount two antennas on a SenCar, while it will likely become difficult and even infeasible to mount more antennas due to the constraint on the distances between antennas to ensure independent fading.

There are some transceiver architectures that can be used as SDMA strategies. For example, each sensor's signal can be demodulated by using a linear decorrelator or a minimum mean square error (MMSE) receiver at the SenCar [87]. Linear decorrelator, known as interference nulling or zero-forcing receiver with low-complexity detection, is the linear filter that maximizes the output signal-to-noise ratio (SNR) subject to the constraint that the filter nulls out the interference from all other data streams. The MMSE receiver is the optimal compromise between maximizing the signal strength from the sensor of interest and suppressing the interference from other sensors. For the simplicity of a SenCar, we will use the linear decorrelator as the SDMA strategy in our scheme.

To use the linear decorrelator, the SenCar makes the data received from one sensor appear as zero at the received data from other sensors. This is possible when the SenCar applies different filters for each sensor on the received signals such that the signals will add up constructively or destructively as desired. To guarantee that the decorrelator operation is successful, we need to limit the number of simultaneous data streams to no more than the number of receive antennas. In other words, since the SenCar is equipped with two receive antennas, at most two sensors can simultaneously send data to a SenCar at a time. Fig. 2.1 shows the transceiver architecture of SDMA with the linear decorrelator. For simplicity, we will use $\mathbf{h}_i$ to denote $[h_{i1}, h_{i2}]^T$, which represents the complex channel coefficient vector between sensor i and the two receive antennas of the SenCar. $\mathbf{h}_1$ and $\mathbf{h}_2$ are the two columns of the channel coefficient matrix $\mathbf{H}$. Suppose sensor 1 wants to upload data x_1 and

sensor 2 wants to upload data x_2 to the SenCar. The received vector at the SenCar can be written as

$$\mathbf{y} = \mathbf{h}_1 x_1 + \mathbf{h}_2 x_2 + \mathbf{n}, \tag{2.1}$$

where $\mathbf{n}$ is i.i.d $\mathcal{CN}(0, \sigma^2 \mathbf{I}_2)$ channel noise. We can see from Eq. (2.1) that each data stream faces an extra source of interference from the other data stream. A method that can be used to remove this inter-stream interference from an interested sensor is to project the received signal $\mathbf{y}$ onto the subspace orthogonal to the one spanned by the other channel vector. That is, we choose $\mathbf{u}_1$ and $\mathbf{u}_2$ as the filter vectors for sensor 1 and sensor 2, respectively, which satisfies $\mathbf{u}_1^* \mathbf{h}_2 = 0$ and $\mathbf{u}_2^* \mathbf{h}_1 = 0$. Hence, the received signal can be decoded as

$$\begin{cases} \hat{x}_1 = \mathbf{u}_1^* \mathbf{y} = \mathbf{u}_1^* \mathbf{h}_1 x_1 + \mathbf{u}_1^* \mathbf{n} \\ \hat{x}_2 = \mathbf{u}_2^* \mathbf{y} = \mathbf{u}_2^* \mathbf{h}_2 x_2 + \mathbf{u}_2^* \mathbf{n}. \end{cases} \tag{2.2}$$

After processed this way, the inter-stream interference nulling can be achieved, i.e., x_1 and x_2 are separated from each other. $\mathbf{u}_1$ can be any vector that lies in V_1 which is the space orthogonal to $\mathbf{h}_2$, however, to maximize the received signal strength, $\mathbf{u}_1$ should lie in the same direction as the projection of $\mathbf{h}_1$ onto V_1. $\mathbf{u}_2$ should be similarly chosen. $\mathbf{u}_1$ and $\mathbf{u}_2$ can be unit vectors as follows since increasing their length will not increase the SNR.

$$\begin{cases} \mathbf{u}_1 = \frac{1}{\sqrt{|h_{22}|^2 + |h_{21}|^2}} \left[h_{22}^*, -h_{21}^*\right]^T \\ \mathbf{u}_2 = \frac{1}{\sqrt{|h_{12}|^2 + |h_{11}|^2}} \left[h_{12}^*, -h_{11}^*\right]^T. \end{cases} \tag{2.3}$$

From Eq. (2.2), we can see that the signal part of $\hat{x}_1$ and $\hat{x}_2$ are $\mathbf{u}_1^* \mathbf{h}_1 x_1$ and $\mathbf{u}_2^* \mathbf{h}_2 x_2$, respectively. Since $|\mathbf{u}_1^* \mathbf{h}_1|^2 \leq \|\mathbf{h}_1\|^2$ and $|\mathbf{u}_2^* \mathbf{h}_2|^2 \leq \|\mathbf{h}_2\|^2$, the projection operation always reduces the length of $\mathbf{h}_i$ unless $\mathbf{h}_i$ is already orthogonal to the channel vector of the other data stream. This is the overhead for nulling out the interference. Hence, the effective channel for x_1 would be in deep fading whenever the projection of $\mathbf{h}_1$ onto $\mathbf{u}_1$ is small. A similar situation is also applicable to x_2. Thus, for given transmission power of each sensor, not any two sensors can successfully transmit data to the SenCar simultaneously. To ensure the SenCar can successfully decode

the received signal, the following criteria should be satisfied

$$\begin{cases} \text{Pr}_1 = P_t|\mathbf{u}_1^*\mathbf{h}_1|^2 \geq \delta_0, \ \text{SNR}_1 = P_t|\mathbf{u}_1^*\mathbf{h}_1|^2/\sigma^2 \geq \delta_1 \\ \text{Pr}_2 = P_t|\mathbf{u}_2^*\mathbf{h}_2|^2 \geq \delta_0, \ \text{SNR}_2 = P_t|\mathbf{u}_2^*\mathbf{h}_2|^2/\sigma^2 \geq \delta_1, \end{cases} \tag{2.4}$$

where Pr_1, Pr_2, SNR_1 and SNR_2 are the received power and SNR of the data from two sensors, respectively, P_t denotes the transmission power of each sensor, and δ_0 is the receive sensitivity while δ_1 is the SNR threshold for the SenCar to correctly decode the received data. Any two sensors that satisfy this criteria can successfully make concurrent data uploading to a SenCar. Such two sensors are said to be *compatible*.

We have explained how SDMA with the linear decorrelator works. When applied to mobile data gathering in WSNs, SDMA has the following benefits. First, since SDMA technique enables the concurrent data uploading from any two compatible sensors to the SenCar, the data uploading time would be significantly reduced. Second, it is commercially appealing that no additional hardware is needed at sensors for performing SDMA. All intelligent operations take place at the SenCar and sensors will simply act as if the SDMA technique is not employed. The SenCar, on the other hand, will need more hardware, such as filters, to process received data. This fits into WSNs quite well because we want the sensors, which are the senders in SDMA, to be as simple as possible, and on the other hand, it is feasible to equip the SenCar with more complex and powerful transceivers. These advantages motivate us to find an optimal solution to harvest the gain of SDMA in the mobile data gathering, which is just the task of this work.

2.3 Design Overview and MDG-SDMA Problem Formulation

2.3.1 Design Overview

In this subsection, we outline the joint design of mobility and SDMA technique. We assume that the SenCar is equipped with two antennas while every sensor has a single antenna and is statically scattered over the field. Before delving into details, for a clear presentation, we define some terms that will be used in the following and

illustrate the joint design as shown in Fig. 2.2.

While a SenCar is moving through the sensing field, it will stop at certain positions to poll nearby sensors. We define the positions that the SenCar can stop for polling as *anchor points*. When the SenCar moves to a anchor point, it polls nearby sensors with the same transmission power as sensors, such that sensors that receive the polling messages can upload data to the SenCar within a single hop. Note that since the SenCar always coordinates nearby sensors and collects data locally, there is no need for global synchronization. The disk-shaped area centered at a anchor point with the radius equal to the sensor transmission range is defined as the *coverage area* of a anchor point. All sensors in the coverage area of a anchor point form the *neighbor set* of this anchor point. For generality, we do not make any assumption on the distribution of sensors or the location-aware capability of nodes. The SenCar obtains the neighbor set information of all anchor points by visiting them at the setup phase of the network. Any two sensors in the same neighbor set are considered to be compatible as defined in Section 2.2 if they satisfy the criteria in Eq. (2.4). Since each sensor needs to be polled only once during a data gathering tour, it is *associated* with only one anchor point even though it may be located within the coverage areas of multiple anchor points. In other words, the associated sensors of a anchor point are not necessarily to be all sensors in its neighbor set. Different association patterns correspond to different compatibility relationship among sensors because the channel state varies from the association patterns. If two sensors are compatible when associated with the same anchor point, they are qualified to be a *compatible pair* to be scheduled to upload data simultaneously when the SenCar arrives. A SenCar does not need to visit every anchor point in the field. However, the anchor points on the tour of the SenCar should cover all sensors in the field. We call these anchor points *selected anchor points*. The SenCar arriving at a selected anchor point collects data from all associated sensors and then moves straightly to the next selected anchor point on the tour. Thus, the moving tour of the SenCar consists of a number of selected anchor points and the straight line segments connecting them. Let $\mathcal{P}'$=$\{p_1, p_2, \ldots, p_t\}$ denote a set of selected anchor points and DS be the static data sink. Then, a possible moving tour of the SenCar can be represented by $DS \rightarrow p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_t \rightarrow DS$. Thus, the problem of finding an optimal solution for a data gathering tour can be considered as jointly solving the following tightly coupled sub-problems: finding compatible pairs among sensors, determin-

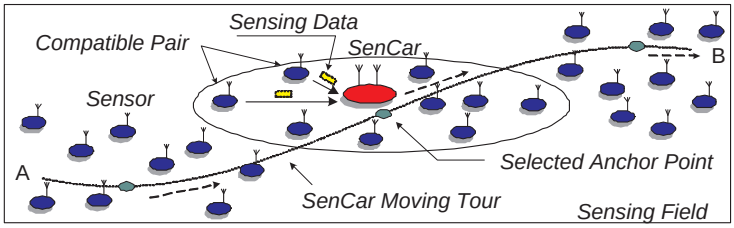
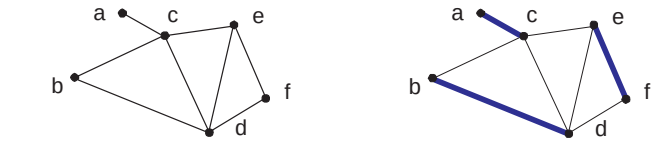


Figure 2.2: Illustration of the joint design of mobility and SDMA technique for mobile data gathering in WSNs.

ing sensor association pattern, and finding locations of selected anchor points and the order for the SenCar to visit them.

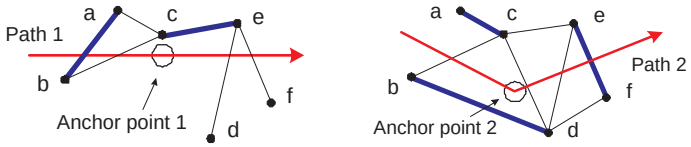
To apply SDMA technique to WSNs and enjoy the benefit it will bring, we have to solve a series of challenging problems. First, the SenCar must be able to determine whether two sensors are compatible. If the possible locations of anchor points are given, the SenCar can receive probe signals from sensors nearby to detect the channel vectors, then determine the compatibility among the sensors in the neighborhood of each location. This information can be measured at the initial setup phase of the network and updated periodically. Second, to collect data as fast as possible, the SenCar should find the maximum number of compatible pairs among sensors. This can be formalized as a matching problem in a compatibility graph, where each vertex represents a sensor, and two vertices are adjacent to each other if two sensors are compatible. For example, the graph in Fig. 2.3(a) shows the compatibility relationship of six sensors around a anchor point, where the anchor point is not shown in the figure. A group of compatible pairs correspond to a set of vertex-disjoint edges in the graph, which is defined as a matching in graph theory [90]. The largest group of compatible pairs corresponds to a maximum matching in the compatibility graph. Fig. 2.3(b) gives three compatible pairs among the sensors, represented by bold edges as a maximum matching in the corresponding compatibility graph. Maximum matching of a graph can be found by algorithms in polynomial time. For example, the efficient implementation of the Edmonds' Blossom Algorithm takes $O(N^3)$ time, where N is the number of vertices in the graph [90, 91, 92].

Note that since the SenCar is mobile, we actually have the freedom to choose the anchor points, where the SenCar will pause for data gathering. The channel



(a) Compatibility graph with 6 sensors. (b) A maximum matching.

Figure 2.3: A maximum matching in the compatibility graph.



(a) Path 1. (b) Path 2.

Figure 2.4: Two possible moving paths of the SenCar.

vectors could vary significantly when the pausing location of the SenCar changes. Thus, intuitively, it is better to let the SenCar visit locations where more sensors are compatible, such that the data can be collected in shorter time. Fig. 2.4 shows two possible moving paths of the SenCar for the example in Fig. 2.3. Path 1 in Fig. 2.4(a) is a straight line, therefore is the shortest path. The compatibility relationship of the six sensors is also shown in Fig. 2.4(a) if the SenCar visits anchor point 1 traveling along this path. We can see that at most two compatible pairs can be found among the sensors (i.e., the size of the maximum matching among the six vertices in the corresponding compatibility graph is 2). In total, 4 time slots are needed for every sensor to upload a packet. However, if the SenCar visits another anchor point along path 2 as shown in Fig. 2.4(b), the six sensors have different compatibility relationship. In this case, there can be found three compatible pairs and it requires only 3 time slots for data uploading. Thus, to complete data gathering as fast as possible, it may be better to take path 2 though it is not the shortest path.

We can see that after mobility and SDMA technique are introduced, the problem of finding a good data gathering tour, which is referred to as the MDG-SDMA problem, becomes more complex. The main benefit of SDMA is to effectively shorten the data uploading time of sensors. However, as mentioned earlier, to better enjoy this benefit, the SenCar may have to visit some specific locations where more sensors are compatible, which may adversely prolong the moving tour. Since our

Table 2.1: Notations used in formulation of MDG-SDMA Problem.

Indices:	
$\mathcal{S} = \{1, 2, \ldots, N_s\}$	A set of sensors.
$\mathcal{P} = \{1, 2, \ldots, N_p\}$	A set of anchor points.
Constants:	
$f_{n,i} = \{0, 1\}$ $\forall n \in \mathcal{S}, \forall i \in \mathcal{P}$	Location indicator. If sensor n is in coverage area of anchor point i, $f_{n,i} = 1$, otherwise, $f_{n,i} = 0$.
$c_{m,n,i} = \{0, 1\}$ $\forall m, n \in \mathcal{S}, \forall i \in \mathcal{P}$	Indicator of compatibility relationship. If sensors m and n are compatible when they are both in coverage area of anchor point i, $c_{m,n,i} = 1$, otherwise, $c_{m,n,i} = 0$.
$d_{i,j} \geq 0$ $\forall i, j \in \mathcal{P}$	Length of arc $a_{i,j}$, i.e., distance between anchor point i and anchor point j.
$q > 0$	Size of the sensing data of each sensor.
$v_d > 0$	Effective data uploading rate of a sensor.
$v_m > 0$	Moving velocity of the SenCar.
Variables:	
$I_i = \{0, 1\}$ $\forall i \in \mathcal{P}$	Indicator of selected anchor point. If anchor point i is selected into $\mathcal{P}'$, $I_i = 1$, otherwise, $I_i = 0$.
$x_{n,i} = \{0, 1\}$ $\forall n \in \mathcal{S}, \forall i \in \mathcal{P}$	Indicator of sensor association. If sensor n is associated with anchor point i, $x_{n,i} = 1$, otherwise, $x_{n,i} = 0$.
$u_{m,n,i} = \{0, 1\}$ $\forall m, n \in \mathcal{S}, \forall i \in \mathcal{P}$	Indicator of compatible pair. If sensors m and n are selected as a compatible pair when they are both associated with anchor point i, $u_{m,n,i} = 1$, otherwise, $u_{m,n,i} = 0$.
$e_{i,j} = \{0, 1\}$ $\forall i, j \in \mathcal{P}$	Indicator of selected line segment in moving tour. If moving tour contains arc $a_{i,j}$, $e_{i,j} = 1$, otherwise, $e_{i,j} = 0$.
$y_{i,j} \geq 0$ $\forall i, j \in \mathcal{P}$	Flow from anchor point i to anchor point j on arc a_{ij}.

objective is to have sensors turn on their radios for a short time, we will focus on minimizing the total data gathering time, which consists of the moving time of the SenCar and the data uploading time of sensors. Thus, the resulting optimal solution does not necessarily have the shortest moving tour or the maximum number of compatible pairs, instead, it is a tradeoff between the shortest moving tour and the full utilization of SDMA.

Note that to solve this problem optimally, we will need the compatibility relationship of sensors for *every* location the SenCar can visit. However, this is impossible in practice, because it is very hard to estimate the channel vectors of sensors at all locations. Thus, we will only consider a *finite* set of anchor points, at which the compatibility relationship among sensors is known (which can be obtained by the SenCar periodically). We denote such a set of anchor points as $\mathcal{P}$. The problem will then be reduced to finding a subset of $\mathcal{P}$, denoted as $\mathcal{P}'$, such that by visiting anchor points in $\mathcal{P}'$ all data can be collected in minimum time. The anchor points in $\mathcal{P}'$ are called selected anchor points.

2.3.2 MDG-SDMA Problem Formulation

We are now in the position to formally formulate the MDG-SDMA problem in WSNs. Given a set of sensors $\mathcal{S} = \{1, 2, \ldots, N_s\}$ and a set of anchor points $\mathcal{P} = \{1, 2, \ldots, N_p\}$, find sensor association patterns and compatible pairs, determine the selected anchor points and the sequence to visit them, such that the sensing data of every sensor in $\mathcal{S}$ can be collected in minimum time. Without loss of generality, we assume that the position of anchor point 1 is also the location of the static data sink, which is the starting and ending points of a data gathering tour. For a clearer presentation, the notations we use are summarized in Table 2.1.

$$\textbf{Minimize } \frac{q}{v_d}\left(|\mathcal{S}| - \frac{1}{2}\sum_{m\in\mathcal{S}}\sum_{n\in\mathcal{S}}\sum_{i\in\mathcal{P}} u_{m,n,i}\right) + \frac{\sum_{i\in\mathcal{P}}\sum_{j\in\mathcal{P}} d_{i,j}e_{i,j}}{v_m} \tag{2.5}$$

Subject to

$$x_{n,i} \leq f_{n,i} \cdot I_i, \ \forall n \in \mathcal{S}, \forall i \in \mathcal{P} \tag{2.6}$$

$$\sum_{i \in \mathcal{P}} x_{n,i} = 1, \forall n \in \mathcal{S} \tag{2.7}$$

$$\sum_{n \in \mathcal{S}} x_{n,i} \geq I_i, \forall i \in \mathcal{P} \tag{2.8}$$

$$u_{m,n,i} \leq \left(\frac{x_{m,i} + x_{n,i}}{2} \right) c_{m,n,i}, \forall m, n \in \mathcal{S}, \forall i \in \mathcal{P} \tag{2.9}$$

$$\sum_{i \in \mathcal{P}} \sum_{m \in \mathcal{S} \setminus \{n\}} u_{m,n,i} \leq 1, \ \forall n \in \mathcal{S} \tag{2.10}$$

$$\sum_{i \in \mathcal{P}} \sum_{n \in \mathcal{S} \setminus \{m\}} u_{m,n,i} \leq 1, \ \forall m \in \mathcal{S} \tag{2.11}$$

$$u_{m,n,i} = u_{n,m,i}, \forall m \in \mathcal{S}, \ \forall n \in \mathcal{S}, \forall i \in \mathcal{P} \tag{2.12}$$

$$\sum_{i \in \mathcal{P}, i \neq j} e_{i,j} = I_j, \ \forall j \in \mathcal{P} \tag{2.13}$$

$$\sum_{j \in \mathcal{P}, j \neq i} e_{i,j} = I_i, \ \forall i \in \mathcal{P} \tag{2.14}$$

$$y_{i,j} \leq |\mathcal{P}| \cdot e_{i,j}, \ \forall i, j \in \mathcal{P} \tag{2.15}$$

$$\sum_{i \in \mathcal{P} \setminus \{1\}} y_{i,1} = \sum_{i \in \mathcal{P} \setminus \{1\}} I_i \tag{2.16}$$

$$\sum_{i \in \mathcal{P} \setminus \{j\}} y_{j,i} - \sum_{k \in \mathcal{P} \setminus \{j\}} y_{k,j} = I_j, \ \forall j \in \mathcal{P} \setminus \{1\} \tag{2.17}$$

Given notations in Table 2.1, the MDG-SDMA problem in WSNs can be formulated as an integer linear program labeled from (2.5) to (2.17). In the formulation, objective function (2.5) minimizes the data gathering time, which consists of the data uploading time of sensors and the moving time of the SenCar. Constraints (2.6)-(2.8) ensure that a sensor should be associated with one and only one selected anchor point within the coverage area the sensor is located, so that its sensing data can be collected during the tour. Constraint (2.9) guarantees that any two sensors that are qualified to be a compatible pair must be associated with the same anchor point and be compatible in the coverage area of this anchor point. Constraints (2.10)-(2.12) enforce that each sensor belongs to at most one compatible pair. Constraints (2.13)-(2.14) ensure the fact that each selected anchor point should have

one arc pointing towards it and another arc pointing away from it. Constraint (2.15) restricts that flow can take place only when the arc is on the moving tour of the SenCar. Constraint (2.16) specifies that the units of flow entering anchor point 1 are equal to the number of selected anchor points since anchor point 1 is the ending point of the tour. Constraint (2.17) enforces that for each selected anchor point, the units of outgoing flow are one unit more than that of the incoming flow [93]. It has been shown in [89] that constraints (2.15)-(2.17) can exclude the solutions of the moving tour with loops and can also prohibit the tour that does not include the given starting and ending anchor points.

We have the following theorem concerning the NP-hardness of the MDG-SDMA problem.

Theorem 1. *The MDG-SDMA problem in WSNs is NP-hard.*

Proof. The NP-hardness of MDG-SDMA problem can be shown by a polynomial-time reduction from the well-known *Traveling Salesman Problem (TSP)* problem to a special case of MDG-SDMA. Given a complete graph $G = (V, E)$ as an instance of TSP, we construct an instance of MDG-SDMA on graph $G' = (V', E')$, which is topologically identical to G. The vertex set of G' includes all anchor points and the data sink, and each edge in G' represents the distance between the two corresponding anchor points. Then assume that no two sensors are compatible with each other and the sensors can only be covered by visiting all anchor points, which can be achieved by imposing some constraints on channel states and sensor transmission power. This reduction is straightforward and can certainly be done in polynomial time. Hence, in this case the SenCar has to visit all anchor points to collect data from each sensor one by one. Since the data uploading time is a constant for a given number of sensors, finding the optimal tour for data gathering is equivalent to finding the shortest round trip that visits each anchor point once. Thus, the TSP in G will have a tour with minimum cost in distance if and only if the same tour in G' is the tour with the minimum length for MDG-SDMA. Hence, MDG-SDMA is NP-hard. □

2.4 Algorithms for MDG-SDMA Problem

We have shown that the MDG-SDMA problem is NP-hard. In this section, we develop three heuristic algorithms to give practically good solutions to the problem in different situations, which are called *maximum compatible pair (MCP) algorithm*, *minimum covering spanning tree (MCST) algorithm*, and *revenue-based (RB) algorithm*, respectively. We describe them one by one next. It is worth pointing out that the solution exploration procedure for each algorithm only needs to be executed when the channel state information is updated or the topology of the network changes, thus does not need to be frequently repeated.

We model the sensing field as a graph $G = (\mathcal{S}, \mathcal{E}, \mathcal{P}, \mathcal{A})$. $\mathcal{S}$ and $\mathcal{P}$ are the sets of sensors and anchor points, respectively, and each element in the sets is a vertex of the graph. $\mathcal{E}$ is the set of edges among the vertices in $\mathcal{S}$. Two vertices in $\mathcal{S}$ are adjacent if the two sensors are compatible in the coverage area of a anchor point. In order to keep the graph simple, there is at most one edge between any two vertices in $\mathcal{S}$ even if the two sensors are compatible in the coverage areas of multiple anchor points. $\mathcal{A}$ is the set of arcs between any two vertices in $\mathcal{P}$. The solution of the MDG-SDMA problem finds a set of selected anchor points $\mathcal{P}'$, which is a subset of $\mathcal{P}$, a matching in $\mathcal{E}$ that represents compatible pairs among sensors and a moving tour of the SenCar with arcs in $\mathcal{A}$ connecting the vertices in $\mathcal{P}'$ in the graph such that the data gathering can be done in minimum time. A possible solution to the MDG-SDMA problem is to break it into two subproblems. The first subproblem is to find a subset of $\mathcal{P}$, $\mathcal{P}'$, that meets certain requirements. This operation is crucial to the solution since it determines the compatibility pattern and the range of the moving tour. The second subproblem is to find the shortest round trip connecting the selected anchor points in $\mathcal{P}'$, which is exactly the well-known TSP problem. As the second subproblem has been well studied before, we will focus on the first subproblem.

2.4.1 Maximum Compatible Pair (MCP) Algorithm

Our first algorithm aims to find a set of selected anchor points that can achieve the maximum number of compatible pairs among sensors, thus is called maximum compatible pair (MCP) algorithm. Based on this objective, $\mathcal{P}'$ should be chosen to

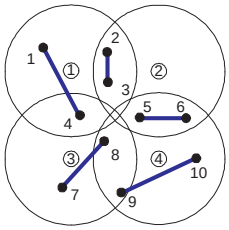
satisfy the following three requirements.

- By visiting the selected anchor points in $\mathcal{P}'$, every sensor can be covered, i.e., all sensors are in the neighbor sets of the selected anchor points in $\mathcal{P}'$.
- Selected anchor points in $\mathcal{P}'$ will allow as many sensors to use SDMA technique as possible, i.e., achieve the maximum number of compatible pairs.
- $\mathcal{P}'$ has the minimum number of selected anchor points that satisfy the above two requirements, which will likely result in a short moving tour.

Theorem 2. *Finding the selected anchor points that satisfy the above three requirement is NP-hard.*

Proof. Assume that there is no compatible pair in the network. Then $\mathcal{P}'$ only needs to satisfy the first and third requirements. This restricted case is simply to find the minimum number of neighbor sets of the anchor points in $\mathcal{P}$ so that the selected sets contain all sensors in the neighbor sets, which is a known NP-complete problem, *Minimum Set Cover* (MSC) problem. Thus, it is clearly NP-hard. □

Fortunately, there are existing approximate algorithms to solve the MSC problem, which can be utilized in our algorithm. The basic idea of the MCP algorithm is to find the minimum number of selected anchor points that can achieve the maximum compatible pairs among sensors. It can be roughly divided into four steps. Next, we explain how the MCP algorithm works by a simple example in Fig. 2.5. Assume that the network has a total of 10 sensors in $\mathcal{S}$ (denoted as S_1-S_{10} and plotted as labeled dots), and 4 anchor points in $\mathcal{P}$ (denoted as P_1-P_4 and plotted as small numbered circles). The disk-shaped area represents the coverage area of the anchor point at the center. Specifically, though S_2 and S_3 are in the coverage areas of both P_1 and P_2, they are compatible only in P_1. On the contrary, S_5 and S_6 are compatible in the coverage areas of both P_2 and P_4. MCP finds the solution in following four steps. In the first step, we find the maximum compatible pairs among all sensors, which is equivalent to finding a maximum matching in the corresponding compatibility graph. Specifically, based on the compatibility relationship (not shown in the figure) with all anchor points in $\mathcal{P}$, we find 5 compatible pairs as shown by solid lines. In the second step, the neighbor set of each anchor point is updated based on the compatible pairs obtained in the first step by deleting the



(a) The maximum matching is 5.

Original Neighbor Set :
P1 : {S1,S2,S3,S4} P2 : {S2,S3,S5,S6}
P3 : {S4,S7,S8} P4 : {S5,S6,S8,S9,S10}
Updated Neighbor Set :
P1 : {S1,S2,S3,S4} P2 : {S5,S6}
P3 : {S7,S8} P4 : {S5,S6,S9,S10}
Selected Polling Point & Sensor Association :
P1 : {S1,S2,S3,S4} P3 : {S7,S8}
P4 : {S5,S6,S9,S10}

(b) The neighbor set updating.

Figure 2.5: An example of the MCP algorithm.

sensors in following two cases: (1) Two sensors in a compatible pair are not compatible in the neighbor set of a anchor point. For example, S_2 and S_3 are deleted from the neighbor set of P_2, since S_2 and S_3 form a compatible pair but they are incompatible in the coverage area of P_2. (2) Two sensors of a compatible pair are in different neighbor sets. For example, S_4 is deleted from the neighbor set of P_3 since its compatible peer S_1 is not in the neighbor set of P_3. Similarly, S_8 is also deleted from the neighbor set of P_4. The original neighbor sets and the updated sets for the example are listed in Fig. 2.5(b). The purpose of this updating procedure is to group any two sensors in a compatible pair as a single element in each neighbor set. In the third step, we can utilize the greedy algorithm for *Minimum Set Cover* problem to find minimum updated neighbor sets of the anchor points in $\mathcal{P}$ that can cover all sensors. In the example, P_1, P_3 and P_4 are finally selected into $\mathcal{P}'$. The updated neighbor sets of the selected anchor points implicitly indicate the association patten of sensors. In the last step, we can run an approximate algorithm for the TSP problem to find the shortest moving tour of the SenCar visiting the selected anchor points in $\mathcal{P}'$. The details of the MCP algorithm are described in Table 2.2.

The MCP algorithm results in maximum compatible pairs among sensors, which leads to the minimum data uploading time. However, the moving tour may not be the shortest one, though the number of selected anchor points that the SenCar has to visit has been minimized. Hence, it is suitable for the networks with densely-deployed sensors where the data uploading time is dominant. For a network with a total of N_s sensors and N_p anchor points, the maximum compatible pairs among N_s sensors can be found by the efficient implementation of the Edmonds' Blossom Algorithm which takes $O(N_s^3)$ time, the updating on neighbor sets takes $O(N_s^2 N_p)$ time, the greedy algorithm finding the minimum neighbor sets that cover all sensors takes $O(N_s N_p \min\{N_s, N_p\})$ time, and the approximate shortest tour on selected anchor points can be found in $O(N_p^2)$ time. Thus, the time complexity of the MCP algorithm is $O(N_s^3 + N_s^2 N_p + N_s N_p \min\{N_s, N_p\} + N_p^2)$. In general, if we have

$N_s > N_p$ or $N_s \approx N_p$, the complexity of the algorithm becomes $O(N_s^3)$.

2.4.2 Minimum Covering Spanning Tree (MCST) Algorithm

For a sparsely distributed sensor network, sensors are generally less likely to be compatible with each other. Thus, under such circumstances, more effort should be focused on reducing the moving time of the SenCar. Thus, $\mathcal{P}'$ should be chosen to satisfy the following requirements.

- By visiting the selected anchor points in $\mathcal{P}'$, all sensors can be covered.
- Visiting the selected anchor points in $\mathcal{P}'$ leads to the shortest moving tour of the SenCar.

Clearly, it is NP-hard to find such $\mathcal{P}'$. Thus we propose a greedy algorithm named minimum covering spanning tree (MCST) algorithm for it. The idea of the algorithm can be described as follows. At each stage of the algorithm, a anchor point with the minimum average cost will be selected into $\mathcal{P}'$. The average cost of an unselected anchor point P_i is defined as the minimum distance between P_i and the elements in $\mathcal{P}'$ divided by the number of uncovered sensors its neighbor set contains. All sensors in the neighbor set of a selected anchor point are considered covered. The algorithm terminates when all sensors are covered. Fig. 2.6 gives a simple example with 10 sensors (denoted as S_1-S_{10} and plotted as labeled dots) and 4 anchor points (denoted as P_1-P_4 and plotted as small numbered circles). We use τ to denote the average cost of a anchor point and use d to denote the distance between two adjacent anchor points in horizontal and vertical directions. We also use superscripts to indicate the stages of the algorithm. In the first stage, since the position of P_1 is also the position of the static data sink, the distances among the four anchor points and the static data sink are 0, d, d and $\sqrt{2}d$, respectively, and the number of uncovered sensors in each neighbor set is 4, 4, 4 and 5, respectively. Thus, $\tau^1(P_1) = 0$, $\tau^1(P_2) = d/4$, $\tau^1(P_3) = d/4$, and $\tau^1(P_4) = \sqrt{2}d/5$. Since P_1 has the minimum average cost, it is selected into $\mathcal{P}'$ and the sensors in its neighbor set are considered covered. In the second stage, the numbers of uncovered sensors in the neighbor sets of the unselected anchor points P_2, P_3 and P_4 have been reduced to 2, 3 and 5, respectively. The minimum distance between each unselected anchor point and the elements in $\mathcal{P}'$ is still d, d and $\sqrt{2}d$, respectively. Thus, $\tau^2(P_2) = d/2$,

Table 2.2: Maximum compatible pair (MCP) algorithm.

Inputs:
A set $\mathcal{S}$ containing all sensors
A set $\mathcal{P}$ containing all anchor points
Neighbor family set $\mathcal{F} = \{f_i | i \in \mathcal{P}\}$, f_i is the neighbor set of anchor point i
Distance matrix $\mathcal{D} = \{d_{i,j}\}_{|\mathcal{P}|\times|\mathcal{P}|}$, where $d_{i,j}$ is the length of arc $a_{i,j} \in \mathcal{A}$, which is the arc between anchor points i and j
Compatibility relationship matrix $\mathcal{C}(\mathcal{P}) = \{c_{m,n,i}\}_{|\mathcal{S}|\times|\mathcal{S}|\times|\mathcal{P}|}$
Outputs:
A set $\mathcal{P}'$ containing the selected anchor points and data sink
Compatible pairs among sensors
Moving tour of the SenCar
MCP Algorithm:
Construct the corresponding compatibility graph based on $\mathcal{C}(\mathcal{P})$;
Find maximum compatible pairs as finding a maximum matching in the corresponding compatibility graph;
Record the compatible pairs in a set $\mathcal{M}$;
For all f_i in $\mathcal{F}$ do
  For all v in f_i do
    If v in $\mathcal{M}$
      If v's compatible peer $\widetilde{v}$ is not in f_i
        Remove v from f_i;
      end if
      If v's compatible peer $\widetilde{v}$ is also in f_i and v and $\widetilde{v}$ are incompatible in the coverage area of anchor point i
        Remove v and $\widetilde{v}$ from f_i;
      end if
    end if
  end for
end for
Find the minimum set cover of $\mathcal{F}$ by the greedy algorithm;
Add corresponding anchor point of selected neighbor sets into $\mathcal{P}'$;
Add the static data sink into $\mathcal{P}'$;
Find an approximate shortest tour on selected anchor points in $\mathcal{P}'$.

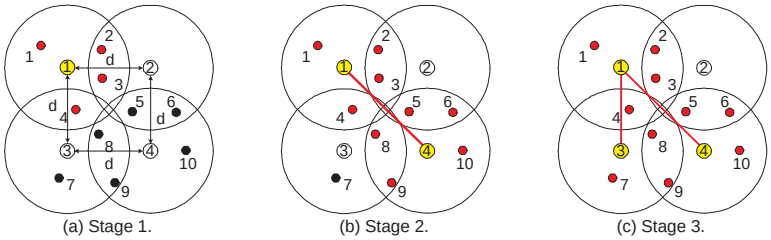


Figure 2.6: An example of the MCST algorithm.

$\tau^2(P_3) = d/3$, and $\tau^2(P_4) = \sqrt{2}d/5$. P_4 has the minimum average cost and is selected. Now $\mathcal{P}' = \{DS, P_1, P_4\}$. Again, the uncovered sensors in the neighbor set of P_4 are considered covered. In the third stage, as there are no uncovered sensors in the neighbor set of P_2, we set $\tau^3(P_2) = \infty$. Since the distance between P_3 and P_4 is equal to that between P_3 and P_1, the minimum distance between P_3 and the elements in $\mathcal{P}'$ is still d. Also, there is only one uncovered sensor in the neighbor set of P_3, thus, $\tau^3(P_3) = d$. Finally, P_3 is selected into $\mathcal{P}'$ and all sensors are covered.

After $\mathcal{P}'$ is found, we can run an approximate algorithm for the TSP problem to find the shortest tour and use the Edmond Blossom algorithm to find the compatible pairs based on the compatibility pattern when sensors are associated with the selected anchor points in $\mathcal{P}'$. Since $\mathcal{P}' \subseteq \mathcal{P}$, the number of compatible pairs obtained here is less than that obtained based on the association pattern with all anchor points in $\mathcal{P}$. The detailed MCST algorithm is given in Table 2.3. MCST algorithm takes $O(N_s N_p \min\{N_s, N_p\})$ time to find a sub-family of neighbor sets which covers all sensors, $O(N_p^2)$ time to determine the approximate shortest tour among selected anchor points, and $O(N_s^3)$ time to find compatible pairs. Thus, its time complexity is $O(N_s N_p \min\{N_s, N_p\} + N_p^2 + N_s^3)$. When $N_s > N_p$ or $N_s \approx N_p$, its time complexity becomes $O(N_s^3)$.

2.4.3 Revenue-Based (RB) Algorithm

In the MCP and MCST algorithms, compatible pairs and the moving tour are separately considered. Now, we propose an algorithm called revenue-based (RB) algorithm, which chooses selected anchor points based on a combined metric of both the number of compatible pairs and the length of moving tour. The basic idea of the RB algorithm is that a selected anchor point is chosen based on the revenue of each unselected anchor point and the algorithm terminates when all sensors are

Table 2.3: Minimum covering spanning tree (MCST) algorithm.

Inputs:
A set $\mathcal{S}$ containing all sensors
A set $\mathcal{P}$ containing all anchor points
Neighbor family set $\mathcal{F} = \{f_i | i \in \mathcal{P}\}$, f_i is the neighbor set of anchor point i
Distance matrix $\mathcal{D} = \{d_{i,j}\}_{|\mathcal{P}|\times|\mathcal{P}|}$, where $d_{i,j}$ is the length of arc $a_{i,j}$, which is the arc between anchor points i and j
Compatibility relationship matrix $\mathcal{C}(\mathcal{P}) = \{c_{m,n,i}\}_{|\mathcal{S}|\times|\mathcal{S}|\times|\mathcal{P}|}$
Outputs:
A set $\mathcal{P}'$ containing the selected anchor points and data sink
Moving tour of the SenCar
Compatible pairs among sensors
MCST Algorithm:
Add the static data sink into $\mathcal{P}'$;
$\mathcal{U} \leftarrow \mathcal{S}$; //The set $\mathcal{U}$ is used to record uncovered sensors.
while $\mathcal{U} \neq \phi$
Find $cost(i)$, $\forall i \in (\mathcal{P} \setminus \mathcal{P}')$, where $cost(i) = \min\{d_{ij} | j \in \mathcal{P}'\}$;
Find the set f_i ($i \in (\mathcal{P} \setminus \mathcal{P}')$) that minimizes $\tau(i) = \frac{cost(i)}{|f_i|}$;
Add the corresponding anchor point i into $\mathcal{P}'$;
Remove sensors in f_i from $\mathcal{U}$;
For each anchor point $j \in (\mathcal{P} \setminus \mathcal{P}')$
Remove sensors in $f_j \cap f_i$ from f_j;
end For
end while
Find an approximate shortest tour on selected anchor points in $\mathcal{P}'$;
Construct corresponding compatibility graph based on $\mathcal{C}(\mathcal{P}') = \{c_{m,n,i}\}_{|\mathcal{S}|\times|\mathcal{S}|\times|\mathcal{P}'|}$, where the elements are obtained from $\mathcal{C}(\mathcal{P})$;
Find compatible pairs as finding a maximum matching in the corresponding compatibility graph.

covered. Specifically, the revenue of an unselected anchor point, say, P_i, is defined as $\mathcal{R}(P_i) = -\alpha\omega(P_i) + \beta\tau(P_i)$, where α and β are positive coefficients, $\omega(P_i)$ is the maximum number of compatible pairs among the uncovered sensors in the neighbor set of P_i, and $\tau(P_i)$ is the average cost of P_i as defined in the MCST algorithm. At each stage, the unselected anchor point with the minimum revenue will be selected. If the selected anchor point is P_i, the sensors in $\omega(P_i)$ compatible pairs will be marked as covered. Other isolated sensors in the neighbor set of P_i are still considered uncovered, so that they could have opportunities to be paired up in a compatible pair within the coverage areas of other anchor points. When

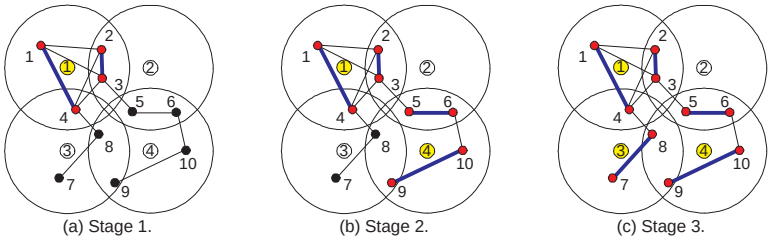


Figure 2.7: An example of the RB algorithm.

there is no more compatible pair that can be found for any unselected anchor point, either all sensors are covered or the uncovered sensors left can no longer be paired up as compatible pairs. In the former case, the algorithm simply terminates since all sensors are covered by existing selected anchor points. In contrast, in the latter case, if the uncovered sensors left are in the neighbor sets of some selected anchor points, they can be randomly associated with one selected anchor point in the current $\mathcal{P}'$ and are considered covered. Otherwise, follow the MCST algorithm to find other anchor points to cover them until all sensors are covered. The detailed RB algorithm can be found in Table 2.5. Since the main complexity of the algorithm is due to the work of finding the maximum compatible pairs among the uncovered sensors in the neighborhood of each unselected anchor point, its time complexity is $O(N_p^2 N_s^3)$ when $N_s > N_p$ or $O(N_s^5)$ when $N_s \approx N_p$.

Table 2.4: Revenue of anchor points in the example of RB algorithm.

	Stage 1	Stage 2	Stage 3
P_1	-2α	-	-
P_2	$-\alpha + \beta d/4$	$-\alpha + \beta d/2$	∞
P_3	$-\alpha + \beta d/4$	$-\alpha + \beta d/3$	$-\alpha + \beta d/2$
P_4	$-2\alpha + \beta\sqrt{2}d/5$	$-2\alpha + \beta\sqrt{2}d/5$	-

To better understand the RB algorithm, we give an example in Fig. 2.7, where the same network configuration as last two examples is used. The compatibility relationship among sensors is shown in solid lines. Note that S_2 and S_3 are only compatible in the coverage area of P_1 while incompatible in the coverage area of P_2. S_5 and S_6 are compatible in the coverage area of both P_2 and P_4. The revenue of the anchor points in each stage is summarized in Table 2.4. In the first stage, the numbers of uncovered sensors in the neighbor sets of P_1 to P_4 are 4, 4, 4 and 5, respectively. The maximum number of compatible pairs among the uncovered sensors in each neighbor set is $\omega^1(P_1) = 2$, $\omega^1(P_2) = 1$, $\omega^1(P_3) = 1$, and $\omega^1(P_4) = 2$. The superscripts still stand for the running stages of the algorithm.

Since P_1 is also the position of the static data sink, $\tau^1(P_1) = 0$ and accordingly, $\mathcal{R}^1(P_1) = -2\alpha$. As P_1 has the minimum revenue in the first stage, it is selected into $\mathcal{P}'$. The sensors in its neighbor set that are also in the two compatible pairs (i.e., S_1-S_4) are considered covered. Now, $\mathcal{P}' = \{DS, P_1\}$. In the second stage, the number of uncovered sensors in each neighbor set of unselected anchor points P_2, P_3 and P_4 has been reduced to 2, 3 and 5, respectively. The maximum number of compatible pairs among the unselected sensors in each updated neighbor set changes to $\omega^2(P_2) = 1$, $\omega^2(P_3) = 1$ and $\omega^2(P_4) = 2$, accordingly. The minimum distances between P_2, P_3, P_4 and the elements in $\mathcal{P}'$ are d, d, and $\sqrt{2}d$, respectively. Among these three unselected anchor points, P_4 is chosen with the minimum revenue $\mathcal{R}^2(P_4) = -2\alpha + \beta\sqrt{2}d/5$ and now $\mathcal{P}' = \{DS, P_1, P_4\}$. The four sensors in the two corresponding compatible pairs, S_5, S_6, S_9 and S_{10}, in its neighbor set are considered covered. S_8 is left uncovered so that it still has the opportunity to be paired up with other uncovered sensors. In the third stage, since there is no uncovered sensor in the neighbor set of P_2, its revenue is set to infinite. As the distance between P_3 and P_4 is equal to that between P_3 and P_1, the minimum distance between P_3 and the elements in $\mathcal{P}'$ is still d. Moreover, there are two uncovered sensors left in its neighbor set, which also happen to form a compatible pair. Hence, $\omega^3(P_3) = 1$ and $\tau^3(P_3) = d/2$. Finally, P_3 is selected and all sensors are covered. Now, the last step is to run an approximate algorithm for the TSP problem to find the shortest tour on the selected anchor points in $\mathcal{P}'$.

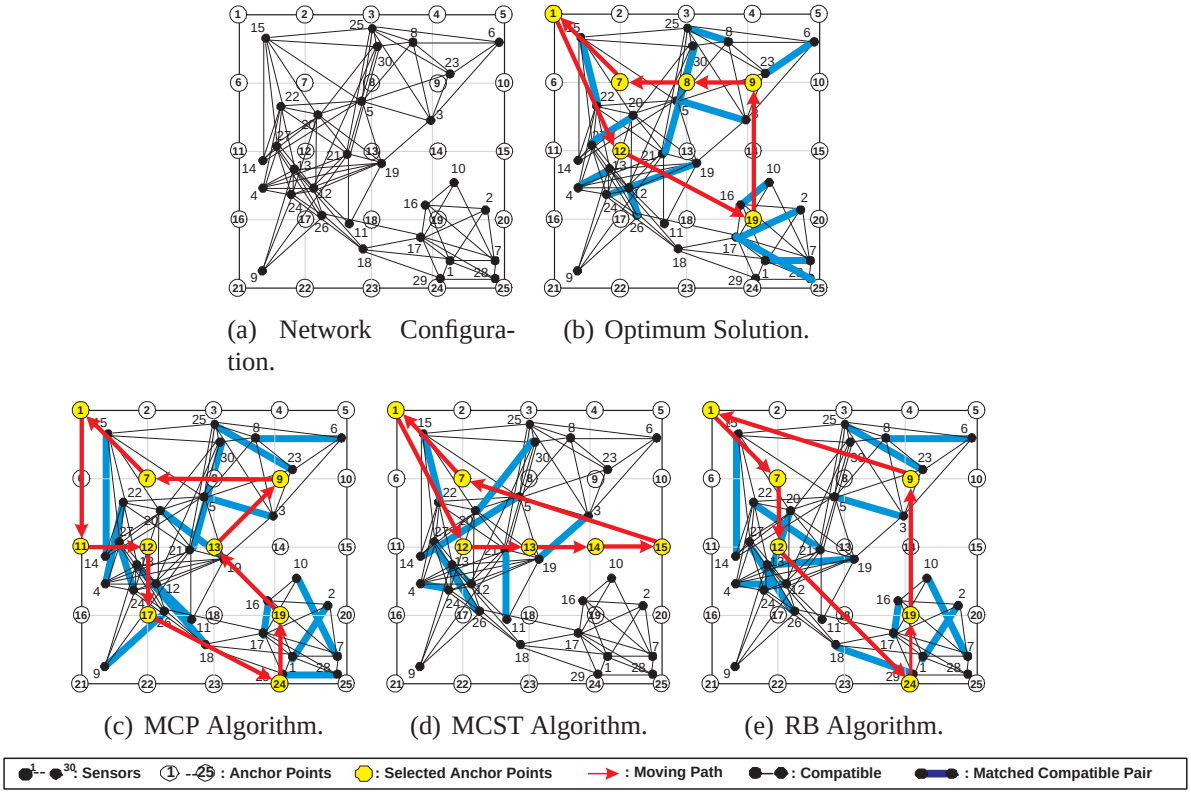
2.5 Performance Evaluation

In this section, we study the performance of proposed algorithms with simulations.

2.5.1 Performance Comparison with Optimum Solution

In this subsection, we investigate the performance of the MCP, MCST and RB algorithms by comparing their results with the optimum solution obtained by CPLEX [94] based on our ILP formulation modeling in AMPL (A Mathematical Programming Language) [95].

We first provide a numerical example to compare our proposed algorithms with the optimum solution. The network configuration is as shown in Fig. 2.8, where a



(a) Network Configuration. (b) Optimum Solution.

(c) MCP Algorithm. (d) MCST Algorithm. (e) RB Algorithm.

Figure 2.8: The solutions of different algorithms.

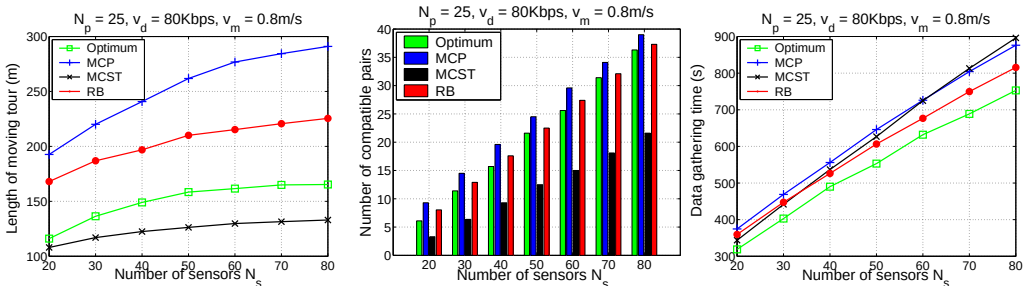
total of 30 sensors scattered over a $60m \times 60m$ square area, and 25 anchor points are located at the intersections of grids and each one is $15m$ apart from its adjacent neighbors in horizontal and vertical directions. We set the radius of the coverage area of each anchor point to $30m$, which is also the transmission range of each sensor. Two sensors are connected by a link in Fig. 2.8(a) if they are compatible in the coverage area of a anchor point. Note that there is only one link between any two compatible sensors even if they are compatible in the coverage areas of multiple anchor points. We assume that the size of the sensing data of each sensor $q = 1Mb$, the effective data uploading rate of each sensor $v_d = 80Kbps$, the moving velocity of the SenCar $v_m = 0.8m/sec$ and $\alpha/\beta = 5$ in the RB algorithm. The solutions of different algorithms are depicted in Fig. 2.8(b)-(e), respectively. It is noticed that the MCP algorithm results in 15 compatible pairs, which has the maximum number of compatible pairs, thus achieves the minimum data uploading time of 188 seconds, while the MCST algorithm focuses on finding the shortest tour that covers all sensors, thus results in the minimum moving time of 184 seconds. In contrast, the RB algorithm pursues the tradeoff between the maximum compatible pairs and the shortest moving tour of the SenCar, and it achieves the shortest total data gathering time of 426 seconds, which is only 3.9% longer than that of the

optimum solution in 410 seconds.

Because of the NP-hardness of the MDG-SDMA problem, the brutal force search method of the optimum solution becomes impossible for a large network. We have managed to obtain optimum solutions for a few small networks for comparing with the MCP, MCST and RB algorithms. In this set of simulations, we measure the length of the moving tour, the number of compatible pairs and the data gathering time of different algorithms when the number of sensors N_s varies from 20 to 80. The sensors are randomly deployed over the sensing field and all other parameter settings are the same as aforementioned. For each point, the performance is the average of the results obtained in 200 simulation experiments. The comparison results are plotted in Fig. 2.9, from which we can draw some observations. First, the length of the moving tours in all solutions first increases with the number of sensors and then tends to be stable with a slight increase when N_s becomes large. It is reasonable since more selected anchor points need to be visited when the number of sensors initially increases and then these anchor points gradually become sufficient for the further increase of sensors. Second, the number of compatible pairs achieved in different solutions follows the trend that MCP$>$RB$>$Optimum$>$MCST. Third, in terms of data gathering time, the RB algorithm always results in close performance to the optimum solution regardless of the number of sensors. It is also noticed that in a very sparsely deployed network (i.e., the cases with a small number of sensors), the MCST algorithm outperforms the MCP algorithm and achieves very close performance to the optimum solution. However, as the network becomes denser with more sensors, the MCP algorithm would adversely surpass the MCST algorithm due to the saving on the data uploading time by the full utilization of SDMA for concurrent data transmissions.

2.5.2 Performance Comparison with Other Data Gathering Schemes

We now evaluate the performance of the MCP, MCST and RB algorithms by comparing their performance with other two non-SDMA data gathering schemes: (1) Single hop data gathering (SHDG) [28]: a mobile collector moves along a well-planned moving tour, which is found in a similar way to the MCST algorithm. However, each sensor always uploads data to the mobile collector with single-input single-output (SISO) transmissions. (2) Data gathering with selected sensor loca-

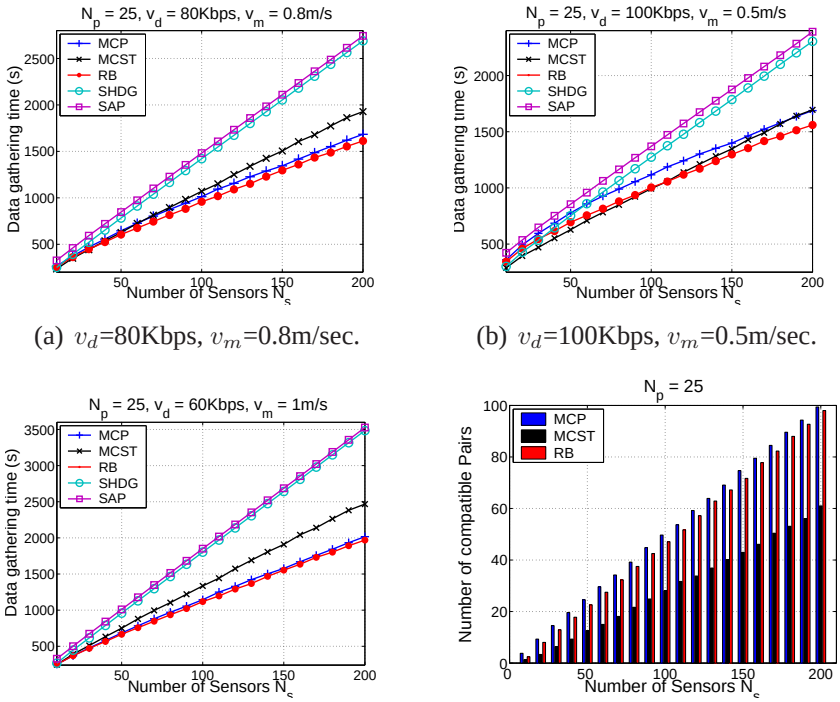


(a) Length of moving tour. (b) Number of compatible pairs. (c) Data gathering time.

Figure 2.9: Comparison between the optimum solution and the proposed algorithms in small networks.

tions as the anchor points (SAP) [33][36]: the locations of a subset of the sensors serve as the anchor points, where the mobile collector would pause to collect data from the sensors nearby with SISO transmission pattern. For a fair comparison, the mobile collector would visit the minimum number of sensor locations that can cover all sensors and collect data from each sensor in a single hop. Assume that N_s sensors are randomly scattered in a $D \times D$ square area and 25 anchor points are located at the intersections of grids with equal intervals. For each value of N_s, the performance is the average of the results obtained in 1000 simulation experiments.

Fig. 2.10 plots the data gathering time and the number of compatible pairs obtained by different schemes when N_s varies from 5 to 200 under different settings of effective data uploading rate v_d and the SenCar's moving velocity v_m. D is set to $60m$. It is shown that the MCP, MCST and RB algorithms always outperform the SHDG and SAP algorithms, and the improvement turns to be more evident when the network becomes denser with more sensors. This is reasonable because the denser distribution of the sensors makes the data uploading time gradually become dominant and provides more opportunities for the sensors to utilize SDMA for concurrent data uploading. For example, when N_s increases to above 100, the RB algorithm can shorten data gathering time by at least 35% compared to the SHDG and SAP algorithms as shown in Fig. 2.10(a). We also notice that when the moving velocity is low, such as $v_m = 0.5m/sec$ as shown in Fig. 2.10(b), the MCST algorithm performs better than the MCP algorithm with up to 17% improvement since the time cost on the moving tour becomes the main factor in this scenario. On the contrary, when the effective data uploading rate is low, such as $v_d = 50Kbps$ in Fig. 2.10(c), the data uploading time overwhelms the moving time, especially



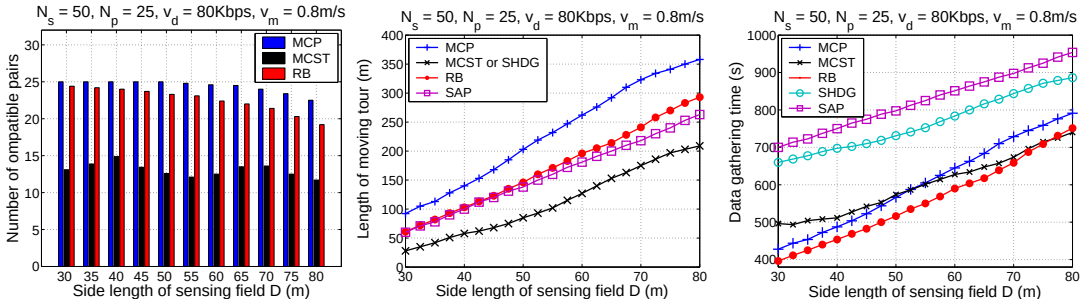
(a) v_d=80Kbps, v_m=0.8m/sec. (b) v_d=100Kbps, v_m=0.5m/sec.

(c) v_d=50Kbps, v_m=1m/sec. (d) Compatible pairs vs N_s.

Figure 2.10: Performance comparisons as the functions of the number of sensors with different settings of v_d and v_m.

with denser sensors. Thus, the MCP algorithm, which is mainly concerned with the maximum compatible pairs, exhibits its advantages in this case, achieving up to 22% improvement compared to the MCST algorithm. In both cases, the RB algorithm always performs best since it jointly considers these parameters in the metric of choosing selected anchor points, thus exhibits a great adaptivity to the variation of these parameters.

Fig. 2.11 plots the number of compatible pairs, the length of moving tour of the SenCar, and the data gathering time obtained by different schemes when the side length of the distributed field D varies from $30m$ to $80m$. N_s is set to 50. It is shown in Fig. 2.11(a) that the number of compatible pairs achieved in the MCP and RB algorithms decreases as D increases. This is intuitive since the compatible opportunities among sensors shrink as they become sparsely distributed. In contrast, the number of compatible pairs in the MCST algorithm fluctuates with the increase of D. This is because that D has a less immediate impact on the compatible pairs in the MCST algorithm since the selection of anchor points tightly depends on shortening the moving tour of the SenCar rather than achieving the maximum compatibility



(a) Number of compatible pairs. (b) Length of moving tour. (c) Data gathering time.

Figure 2.11: Performance comparisons as the functions of the side length D of the distributed field.

among sensors. Fig. 2.11(b) demonstrates that the moving tour is prolonged as D increases for all schemes under investigation. The MCST and SHDG result in the shortest moving tours while the SAP scheme ranks the second. The reason for the longer tour in SAP is because that the SenCar has to visit the exact locations of the specified sensors while it only needs to traverse the transmission range of the sensors in MCST and SHDG. Finally, Fig. 2.11(c) indicates that the MCP, MCST and RB algorithms can greatly shorten the data gathering time with respect to SHDG and SAP since they can cut down data uploading time by utilizing SDMA. For example, when $D = 50m$, the RB algorithm outperforms SHDG and SAP by 31% and 36% on the data gathering time, respectively.

2.6 Conclusions

In this chapter, we introduce efficient mobile data gathering scheme with mobility and SDMA technique. We formulate this problem into an ILP, called MDG-SDMA problem, and prove its NP-hardness. Consequently, we propose three algorithms, named MCP, MCST, and RB algorithms, to provide practically good solutions to the problem. Extensive simulation results demonstrate that the proposed algorithms can achieve much shorter data gathering time than other compared schemes.

Table 2.5: Revenue-based (RB) algorithm.

Inputs:
A set $\mathcal{S}$ containing all sensors
A set $\mathcal{P}$ containing all anchor points
Neighbor family set $\mathcal{F} = \{f_i | i \in \mathcal{P}\}$, f_i is the neighbor set of anchor point i
Distance matrix $\mathcal{D} = \{d_{i,j}\}_{|\mathcal{P}|\times|\mathcal{P}|}$, where $d_{i,j}$ is the length of arc $a_{i,j} \in \mathcal{A}$, which is the arc between anchor points i and j
Compatibility relationship matrix $\mathcal{C}(\mathcal{P}) = \{c_{m,n,i}\}_{|\mathcal{S}|\times|\mathcal{S}|\times|\mathcal{P}|}$
Outputs:
A set $\mathcal{P}'$ containing the selected anchor points and data sink
Compatible pairs among sensors
Moving tour of the SenCar
RB Algorithm:
Add the static data sink into $\mathcal{P}'$;
$\mathcal{U} \leftarrow \mathcal{S}$; //The set $\mathcal{U}$ is used to record uncovered sensors.
while ( )
For each anchor point $i \in (\mathcal{P} \setminus \mathcal{P}')$
Construct compatibility graph from $\mathcal{C}(i) = \{c_{m,n,i}\}_{|f_i|\times|f_i|}$ where the elements can be obtained from $\mathcal{C}(\mathcal{P})$;
Find maximum compatible pairs among the sensors in f_i as finding maximum matching in the compatibility graph;
Use $\omega(i)$ to record the number of compatible pairs;
end For
If $\exists i \in (\mathcal{P} \setminus \mathcal{P}')$ that $\omega(i) \neq 0$
For each anchor point $i \in (\mathcal{P} \setminus \mathcal{P}')$
Find $cost(i) = \min\{d_{ij} | j \in \mathcal{P}'\}$;
Calculate the revenue of anchor point i by:
$\mathcal{R}(i) = -\alpha \cdot \omega(i) + \beta \cdot \frac{cost(i)}{|f_i|}$;
end for
Find the set f_i ($i \in (\mathcal{P} \setminus \mathcal{P}')$) that minimizes $\mathcal{R}(i)$;
Add corresponding anchor point i into $\mathcal{P}'$;
Record corresponding $\omega(i)$ compatible pairs;
Remove the sensors in $\omega(i)$ compatible pairs from $\mathcal{U}$;
For each anchor point $j \in \mathcal{P}$ and $j \neq i$
Remove the sensors in $\omega(i)$ compatible pairs from f_j;
end for
else break;
end if
end while
If $\mathcal{U} \neq \phi$
For each node $v \in \mathcal{U}$
If v is in some neighbor sets of selected anchor points in $\mathcal{P}'$
Randomly associate v with a selected anchor point $i \in \mathcal{P}'$ that can cover it, and update $\mathcal{F}$ and $\mathcal{U}$, accordingly;
end If
end For
end if
If $\mathcal{U} \neq \phi$
Follow MCST algorithm to find new selected anchor points that cover the left sensors;
end If
Find an approximate shortest tour on anchor points in $\mathcal{P}'$.

Chapter 3

A Region Division and Tour Planning Algorithm for Mobile Data Gathering with Multiple Mobile Collectors and SDMA Technique

In this chapter, we extend the joint design of mobility and SDMA technique to large-scale sensor networks, where a certain number of multiple SenCars are available for use. In particular, the sensing field is divided into several non-overlapping regions, each having a SenCar. Each SenCar gathers data from sensors in the region while traversing their transmission ranges. We also consider exploiting SDMA technique by equipping each SenCar with two antennas. With the support of SDMA, two distinct compatible sensors in the same region can successfully make concurrent data uploading to their associated SenCar. We focus on the problem of minimizing maximum data gathering time among different regions, which is referred to as mobile data gathering problem with multiple mobile collectors and SDMA, or MDG-MS for short. Accordingly, we propose a region-division and tour-planning (RDTP) algorithm, in which data gathering time is balanced among different regions. Simulation results show that the RDTP algorithm with two available SenCars achieves at least 56% time saving compared to the non-SDMA+single SenCar scheme.

The rest of this chapter is organized as follows. Section 3.1 outlines the joint design of mobility and SDMA technique when multiple SenCars are available. Sec-

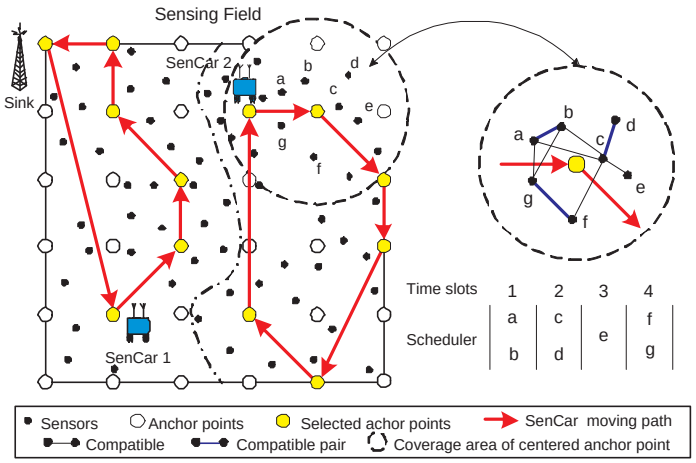


Figure 3.1: Two SenCars are deployed in the sensing field and gather data simultaneously in different regions.

tion 3.2 provides the definition and formulation of MDG-MS problem. Section 3.3 describes and illustrates the details of RDTP algorithm as the solution to the MDG-MS problem. Section 3.4 shows the performance results of RDTP algorithm. Finally, Section 3.5 concludes the chapter.

3.1 Design Overview

In the previous chapter, we have studied how to plan an efficient data gathering tour when a single SenCar and SDMA technique are employed. However, in a large-scale WSN, utilizing only a single SenCar may lead to a long data gathering tour and cause data buffer overflow at sensors. To effectively deal with these problems, in this chapter, we consider deploying multiple SenCars that work simultaneously in a sensing field, each having the capability of exploiting SDMA technique to collect data from scattered sensors in its sub-area.

In the data gathering scheme with multiple SenCars and SDMA technique, the sensing field is divided into several non-overlapping *regions*, each having a SenCar. We assume that each SenCar can forward the gathered data to one of its nearby SenCars, such that all data can be forwarded to the SenCar that will visit the static data sink. The data forwarding among the SenCars can be performed when they complete data gathering in each region or even can be done while they are moving

on the paths except when the SenCars are communicating with their associated sensors. This ensures that such inter-SenCar forwarding would not impact on data gathering between sensors and their associated SenCars. In each region, a SenCar takes the responsibility of collecting data from local sensors, similar to the case of a single SenCar with SDMA technique that we have discussed in the previous chapter. Fig. 3.1 gives an example, where two available SenCars are working on non-overlapping regions and SenCar 1 would visit the data sink on its tour. When a SenCar arrives at a selected anchor point in its region, its associated sensors are scheduled to communicate with the SenCar by utilizing SDMA technique. Two sensors in a compatible pair would upload data simultaneously in a time slot, while an isolated sensor (i.e., a sensor not in any compatible pair) would send data to the SenCar separately.

3.2 MDG-MS Problem

We assume that sensors can turn to sleep mode when the data gathering in the respective regions is completed. Thus, finding optimal data gathering strategies to prolong entire network lifetime and shorten data gathering latency among different regions is equivalent to minimizing the maximum data gathering time among different regions. This problem is referred to as data gathering with multiple SenCars and SDMA technique (MDG-MS) problem. Besides pursuing the tradeoff between the shortest moving tour and the full utilization of SDMA technique as in the MDG-SDMA problem we have discussed in Chapter 2, the focus of the MDG-MS problem is to properly partition the selected anchor points and their associated sensors so as to balance the data gathering time among different regions.

The MDG-MS problem can be formally described as follows. Given a set of sensors $\mathcal{S} = \{1, 2, \ldots, N_s\}$, a set of anchor points $\mathcal{P} = \{1, 2, \ldots, N_p\}$, and a set of SenCars $\mathcal{K} = \{1, 2, \ldots, N_k\}$, find: (1) a set of subsets of $\mathcal{P}$, denoted by $\mathcal{P}'_1, \mathcal{P}'_2, \ldots, \mathcal{P}'_{N_k}$, which represent the selected anchor points in different regions that satisfy $\mathcal{P}'_1 \bigcap \mathcal{P}'_2 \bigcap \cdots \bigcap \mathcal{P}'_{N_k} = \Phi$ and $\mathcal{P}'_1 \bigcup \mathcal{P}'_2 \bigcup \cdots \bigcup \mathcal{P}'_{N_k} = \mathcal{P}' \subseteq \mathcal{P}$, (2) a set of subsets of $\mathcal{S}$, denoted by $\mathcal{S}'_1, \mathcal{S}'_2, \ldots, \mathcal{S}'_{N_k}$, which represent the sensors assigned in different regions that satisfy $\mathcal{S}'_1 \bigcap \mathcal{S}'_2 \bigcap \cdots \bigcap \mathcal{S}'_{N_k} = \Phi$ and $\mathcal{S}'_1 \bigcup \mathcal{S}'_2 \bigcup \cdots \bigcup \mathcal{S}'_{N_k} = \mathcal{S}$, (3) the compatible pairs among the sensors in $\mathcal{S}'_i$, $i = 1, 2, \cdots, N_k$, (4) the sequence by which each SenCar visits the selected anchor points in $\mathcal{P}'_i$, $i = 1, 2, \cdots, N_k$,

such that the maximum data gathering time among a total of N_k regions can be minimized. The MDG-MS problem can be similarly formulated into an ILP problem as the MDG-SDMA problem, since the latter is a special case of the former with N_k equal to 1. It is known that the complexity of the ILP solution is generally high, which may not be suitable for large-scale sensor networks. Thus, next we propose a heuristic region-division and tour-planning algorithm (RDTP) to solve this problem.

3.3 Region-Division and Tour-Planning (RDTP) Algorithm

The basic idea of RDTP algorithm is to first consider the sensing field as a whole, find compatible pairs and selected anchor points that would result in short data gathering time as in the single SenCar case, then assign each of the selected anchor points a weight and divide them into different regions based on their weight. Specifically, the RDTP algorithm contains four steps: (1) Find the compatible pairs among sensors that result in minimum data uploading time; (2) Determine the selected anchor points, which can achieve the compatible pairs obtained in the first step and meanwhile lead to a short moving tour; (3) Build the minimum spanning tree among selected anchor points and assign a weight for each vertex on the tree; (4) Decompose the minimum spanning tree into a set of subtrees based on the weight of each vertex and find the shortest moving tours along the selected anchor points on each subtree. We will describe the details in the following with the help of an example in Fig. 3.2. In the example, there are 20 sensors in $\mathcal{S}$ (plotted as labeled dots) and 25 anchor points in $\mathcal{P}$ (plotted as small numbered circles). The compatibility relationship among sensors is shown in Fig. 3.2(a), where any two compatible sensors are connected by a link. We have two SenCars available. Thus, the problem is to divide the sensing field into two regions and plan the moving tour for each SenCar in order to balance their data gathering time.

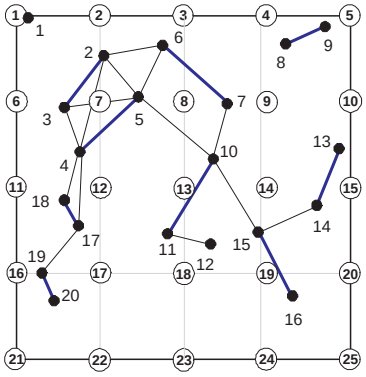
Since the data uploading time of sensors may become dominant compared to the moving time of the SenCar as the number of sensors increases, it is reasonable to consider the data uploading time as the main factor that affects the selection of the anchor points. Thus, in the first and second steps of the RDTP algorithm, we follow

the principles of MCP algorithm for the single SenCar case to determine the compatible pairs and selected anchor points. That is, regardless of the region division pattern, we would first consider the sensing field as a whole to find the maximum compatible pairs among all sensors and the minimum number of selected anchor points that can achieve such maximum compatible pairs. In this way, the sensors can achieve the full utilization of SDMA globally for the minimum data uploading time and the SenCar would also move on a short tour by visiting the minimum number of selected anchor points. The details are similar to those in MCP algorithm in Chapter 2 and we will not repeat here. In the example, there are 9 compatible pairs among the 20 sensors to the maximum extent based on their compatibility relationship, which are plotted as the bold lines in Fig. 3.2(a). Without loss of generality, we still assume that the data sink is located at the position of anchor point 1. Fig. 3.2(b) indicates that anchor points $1, 4, 7, 8, 12, 13, 15, 16$ and 19 are chosen as the selected anchor points, i.e., the elements in $\mathcal{P}'$. Fig. 3.2(e)-(f) list the original and updated neighbor sets of each anchor point and Fig. 3.2(g) shows the selected anchor points and their associated sensors, which are also shown in Fig. 3.2(b) as the sensors in the shadowed area around each selected anchor point.

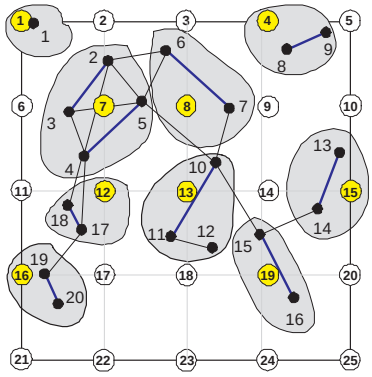
After finding compatible pairs and selected anchor points, in the third step of the RDTP algorithm, we organize the selected anchor points into a tree structure and assign each of them a weight as the metric for partition. Specifically, we find the minimum spanning tree $T(V, E)$ among the selected anchor points in $\mathcal{P}'$ rooted at the static data sink, denoted by r_T. For example, in Fig. 3.2(c) the minimum spanning tree among the nine selected anchor points is shown as the bold lines connecting them rooted at anchor point 1. Let $w(v)$ represent the weight for selected anchor point v. Next, we calculate the weight for each selected anchor point in $\mathcal{P}'$ according to the following criteria.

$$w(v) = \sum_{u \in V(\text{subT}(v))} \rho(|f_u| - |M_u|) + \sum_{e \in E(\text{subT}(v))} \lambda L_e, \quad v \in \mathcal{P}' \tag{3.1}$$

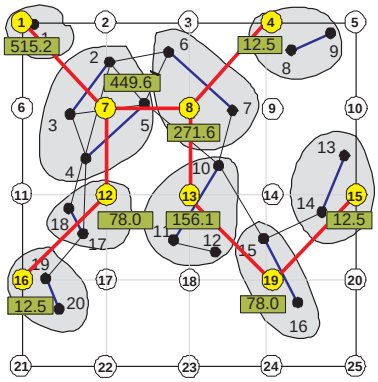
where ρ and λ are constant coefficients, which represent the time for a sensor to upload its data and for a SenCar to move a unit distance, respectively, $\text{subT}(v)$ denotes the subtree of T rooted at v, $V(\cdot)$ and $E(\cdot)$ represent the vertices and edges on the tree, f_u denotes the set of associated sensors with selected anchor point u,



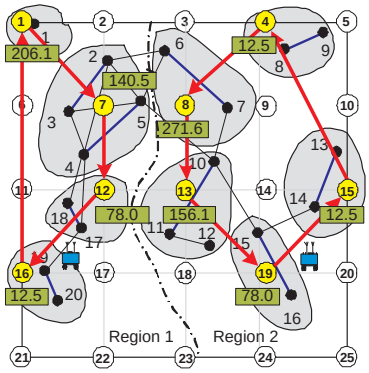
(a) Step 1: Find the maximum compatible pairs among sensors.



(b) Step 2: Determine the selected anchor points and association pattern of sensors.



(c) Step 3: Find the minimum spanning tree among the selected anchor points.



(d) Step 4: Divide the selected anchor points into two parts based on their weights.

Original neighbor set of each anchor point:				
1: {1}	6: {3,4}	11: {4,17,18}	16: {17,19,20}	21: {20}
2: {1,2,6}	7: {2,3,4,5,6}	12: {4,11,17,18}	17: {11,17,19,20}	22: {20}
3: {6}	8: {5,6,7,10}	13: {10,11,12}	18: {11,12}	23: { }
4: {8,9}	9: {7,8,10}	14: {10,12,14,15}	19: {12,14,15,16}	24: {16}
5: {8,9}	10: {9,13}	15: {13,14}	20: {14,16}	25: {16}

(e) Original neighbor set of each anchor point

Updated neighbor set of each anchor point:				
1: {1}	6: { }	11: { }	16: {19,20 }	21: { }
2: {1}	7: {2,3,4,5}	12: {17,18}	17: { }	22: { }
3: { }	8: {6,7}	13: {10,11,12}	18: {12}	23: { }
4: {8,9}	9: { }	14: { }	19: {15,16}	24: { }
5: {8,9}	10: { }	15: {13,14}	20: { }	25: { }

(f) Updated neighbor sets based on maximum compatible pairs.

Selected anchor points and their associated sensors:	
1: {1}	12: {17,18}
4: {8,9}	13: {10,11,12}
7: {2,3,4,5}	15: {13,14}
8: {6,7}	16: {19,20 }
	19: {15,16}

(g) Selected anchor points.

Figure 3.2: Illustration of the region-division and tour-planning (RDTP) algorithm.

M_u denotes the compatible pairs among the sensors associated with selected anchor point u, and L_e is the length of edge e. The first term of $w(v)$ represents the sum of the data uploading time at the selected anchor points on the subtree rooted at v, while the second term is the sum of the moving time along the edges on the subtree. Hence, the weight of vertex v implicitly indicates the expected data gathering time if a SenCar visits each of the selected anchor points on the subtree rooted at it and collects data from the associated sensors. Apparently, the root has the largest weight compared to all other vertices on T. We consider it as the total weight of T and denote it as W_T. In the example, the weight for each selected anchor point is labeled as shown in Fig. 3.2(c), where ρ and λ are set to 12.5 and 1.25, respectively and the two adjacent anchor points are $30m$ apart. Under these settings, W_T is equal to 515.2, which is also the weight of the root of T (i.e., $w(1)$).

Now, the remaining problem is how to divide the selected anchor points and their associated sensors into different regions (for different SenCars), in order to balance the data gathering time among these regions. In the fourth step of the algorithm, we focus on solving this problem. Suppose there are a total of N_k available SenCars, which means that the selected anchor points are to be partitioned into N_k parts. The basic idea is to decompose the minimum spanning tree T into N_k parts, by iteratively finding a subtree t based on the weight of each vertex on T and pruning t from T. To build a subtree in each iteration, first find the farthest leaf vertex v on T with the minimum weight. Let m denote the number of remaining SenCars at each iteration, thus, $m = N_k$ initially. If $w(v) < W_T/m$, find its parent vertex on T, denoted by $PA(v)$ and let $v = PA(v)$. Check its weight and repeat this up-tracing process until $w(v) \geq W_T/m$. Record this vertex v and consider it as the root of the subtree t. All vertices on t are removed from T, which means that the corresponding selected anchor points on t will be assigned to a region (or a SenCar). After that, update W_T, m, and $w(v)$ for each vertex on the updated T, and then repeat the procedure to find another subtree. When there is only one available SenCar left, i.e., $m = 1$, all remaining selected anchor points and their associated sensors are simply assigned to this SenCar and the procedure terminates. To better understand it, let us take a look at the example in Fig 3.2(c). In the first iteration, anchor point 15, as the farthest leaf vertex on T, has the minimum weight equal to 12.5. Thus, v is set to 15. Since $m = N_k = 2$ and $W_T = 515.2$, $w(15) < W_T/m = 257.6$. Next, check the weight of anchor point 19, which is the parent vertex of anchor

Table 3.1: Procedure of dividing selected anchor points and their associated sensors into N_k parts.

Procedure Division (T, N_k)
For all v on T do
  Calculate $w(v)$ according to Eq. (3.1);
end for
$m \leftarrow N_k$;
While $m > 1$
  $W_T \leftarrow w(r_T)$;
  $v \leftarrow$ the farthest leaf vertex on T with minimum $w(v)$;
  While $w(v) < W_T/m$
    $v \leftarrow PA(v)$;
  end while
  Build the subtree t of T rooted at v;
  Add the vertices on t to $\mathcal{P}'_m$;
  Add the corresponding associated sensors to $\mathcal{S}'_m$;
  Remove the subtree t from T;
  Update $w(v)$ for each v on the remaining T;
  $m \leftarrow m - 1$;
end while
Assign the remaining selected anchor points on T to $\mathcal{P}'_1$;
Assign the corresponding associated sensors to $\mathcal{S}'_1$;
Find approximate shortest tours that visit selected anchor points in $\mathcal{P}'_1, \mathcal{P}'_2, \ldots, \mathcal{P}'_{N_k}$, respectively.

point 15 on T. Since $w(19)$ is still less than W_T/m, the up-tracing continues until anchor point 8 is found with the weight larger than W_T/m. Thus, anchor point 8 is considered as the root of the subtree t. All vertices on t, which are anchor points $4, 8, 13, 15$ and 19, are assigned to $\mathcal{P}'_2$ and their associated sensors are assigned to $\mathcal{S}'_2$. This means that the sensors in $\mathcal{S}'_2$ are considered belonging to region 2 and the SenCar responsible for region 2 will gather the sensing data from these sensors by visiting the selected anchor points in $\mathcal{P}'_2$. After removing these selected anchor points from T, update $w(v)$ for each vertex on the remaining T, which is shown in Fig. 3.2(d). Also, W_T is recalculated with the result equal to 206.1 and m is updated to 1. Now, since only one available SenCar is left (i.e., $m = 1$), all remaining selected anchor points and their associated sensors are simply assigned to $\mathcal{P}'_1$ and

Table 3.2: Region-division and tour-planning (RDTP) algorithm.

Inputs:
Set $\mathcal{S}$ containing N_s sensors
Set $\mathcal{P}$ containing N_p anchor points
Set $\mathcal{K}$ containing N_k SenCars
Neighbor family set $\mathcal{F} = \{f_i | i \in \mathcal{P}\}$, where f_i is the neighbor set of anchor point i
Distance matrix $\mathcal{D} = \{d_{i,j}\}_{|\mathcal{P}|\times|\mathcal{P}|}$, where $d_{i,j}$ is the segment length between anchor points i and j
Compatibility relationship matrix $\mathcal{C}(\mathcal{P}) = \{c_{m,n,i}\}_{|\mathcal{S}|\times|\mathcal{S}|\times|\mathcal{P}|}$
Outputs:
A set $\mathcal{P}'$ of selected anchor points with $\mathcal{P}' = \mathcal{P}'_1 \bigcup \mathcal{P}'_2 \bigcup \ldots \bigcup \mathcal{P}'_{N_k}$
A set of subsets of $\mathcal{P}$, $\mathcal{P}'_1, \mathcal{P}'_2, \ldots, \mathcal{P}'_{N_k}$, each representing selected anchor points in a region
A set of subsets of $\mathcal{S}$, $\mathcal{S}'_1, \mathcal{S}'_2, \ldots, \mathcal{S}'_{N_k}$, each representing sensors associated in a region
Compatible pairs in each region
Moving tour of each SenCar
RDTP algorithm:
1: Find maximum compatible pairs $\mathcal{M}$ based on $\mathcal{C}(\mathcal{P})$;
2: Update $\mathcal{F}$ according to $\mathcal{M}$;
Find minimum set cover of $\mathcal{F}$ by the greedy algorithm;
Add corresponding anchor points of selected neighbor sets to $\mathcal{P}'$;
3: Find minimum spanning tree T among selected anchor points in $\mathcal{P}'$;
Calculate the weight of each vertex on T;
4: Divide $\mathcal{P}'$ into $\mathcal{P}'_1, \mathcal{P}'_2, \ldots, \mathcal{P}'_{N_k}$ and divide $\mathcal{S}$ into $\mathcal{S}'_1, \mathcal{S}'_2, \ldots, \mathcal{S}'_{N_k}$ by iteratively finding a subtree of T;
Find approximate shortest tours that visit selected anchor points in $\mathcal{P}'_1, \mathcal{P}'_2, \ldots, \mathcal{P}'_{N_k}$, respectively.

$\mathcal{S}'_1$, respectively. Finally, we determine the shortest moving tour for each SenCar to visit the selected anchor points in its region by the approximate algorithm for the TSP problem. As a result, the moving tours of the two SenCars in the example are as given in Fig. 3.2(d): SenCar 2: $8 \rightarrow 13 \rightarrow 19 \rightarrow 15 \rightarrow 4 \rightarrow 8$ and SenCar 1: $1(DS) \rightarrow 7 \rightarrow 12 \rightarrow 16 \rightarrow 1(DS)$. The details of the procedure in step 4 of the RDTP algorithm are given in Table 3.1.

Finally, we summarize the RDTP algorithm in Table 3.2. Given a total of N_s sensors and N_p anchor points, the time complexity of the RDTP algorithm can be analyzed as follows. The operations of the first two steps in the RDTP algorithm, which are identical to those in the MCP algorithm, take $O(N_s^3 + N_s^s N_p + N_s N_p \min(N_s, N_p))$ time. In the third step, a simple implementation of Prim's algorithm [88] for finding the minimum spanning tree among the selected anchor points requires $O(N_p^2)$ time. Assigning the weight for each vertex on the spanning tree costs $O(N_p)$ time. For the last step as shown in Table 3.1, given a total of m available SenCars, the outer while loop will be executed $m-1$ rounds and the inner while loop will be performed at most N_p times. Furthermore, finding the approximate shortest tours that visit the selected anchor points in different regions can be done in $O(N_p^2)$ time. Thus, in the general case where $N_s \geq N_p$, the overall time complexity of the RDTP algorithm is $O(N_s^3)$.

3.4 Performance Evaluation

In this section, we evaluate the performance of the RDTP algorithm for the MDG-MS problem and compare it with other three mobile data gathering schemes.

We still consider a $D \times D$ square sensing field, where a total of N_s sensors are randomly distributed, and N_p anchor points are located at the intersections of grids with each anchor point apart from its adjacent anchor points in horizontal and vertical directions with equal distance. A total of N_k SenCars are available for use. We assume that the radius r of the coverage area of each anchor point is $30m$, the size of the sensing data q in each sensor is $1Mb$, the effective data uploading rate of each sensor v_d is equal to $80Kbps$, and the moving velocity of each SenCar v_m is set to $0.8m/sec$, if not stated otherwise.

Fig. 3.3 plots the data gathering time of different schemes when N_s varies from 10 to 150, where N_p is equal to 36 and D is set to $100m$. We compare four

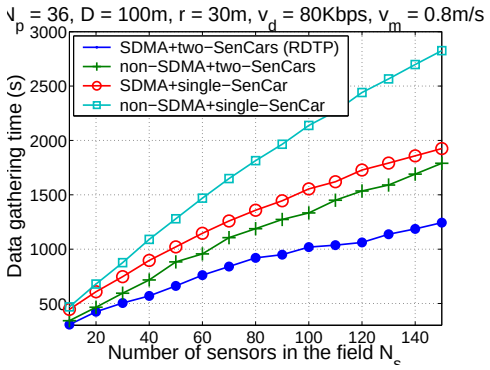


Figure 3.3: Performance of the RDTP algorithm: Data gathering time of four schemes as a function of N_s.

mobile data gathering schemes: without SDMA and with a single SenCar (non-SDMA+single-SenCar), with SDMA and with a single SenCar (SDMA+single-SenCar), without SDMA and with two SenCars (non-SDMA+two-SenCars), and with SDMA and with two SenCars (SDMA+two-SenCars, which is the RDTP scheme). When multiple SenCars are used, the data gathering time refers to the maximum time of a data gathering tour among different regions. It can be seen that the data gathering time of all schemes increases as N_s increases. However, the RDTP always outperforms other schemes due to the concurrent use of multiple SenCars and simultaneous data uploading among the sensors with the support of SDMA technique. For instance, it achieves 56% time saving compared to the non-SDMA+single-SenCar scheme when N_s is set to 100. This trend of superiority becomes even more remarkable as N_s increases. Shorter data gathering time leads to longer network lifetime since sensors can turn to power-saving mode once the data gathering in their region is done.

Fig. 3.4 shows that data gathering time of RDTP varies with N_s under different settings of v_m and v_d, where D is set to $100m$. There are 36 anchor points and two available SenCars. We consider two configurations of (v_m, v_d), which are ($v_m = 1m/sec$, $v_d = 50kbps$) and ($v_m = 0.6m/sec$, $v_d = 110kbps$), to represent two different cases. The observations in the single SenCar algorithms are still applicable here. It is noticed that when N_s is small, the moving velocity of SenCar v_m has a greater impact on the data gathering time than v_d. Higher moving velocity, such as $v_m = 1m/sec$ in Case I, results in shorter data gathering time even with a smaller

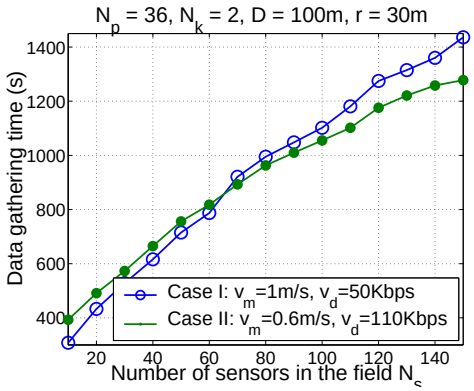


Figure 3.4: Performance of the RDTP algorithm: Data gathering time of RDTP under different settings of v_m and v_d.

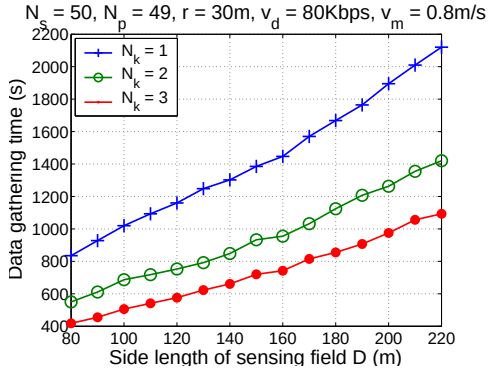


Figure 3.5: Performance of the RDTP algorithm: Data gathering time of RDTP under different settings of N_k.

v_d than the other case. It is reasonable since the moving time of each SenCar is dominant when the sensors are sparsely scattered. On the contrary, when N_s is large, the effect of v_d on the data gathering time of a SenCar overwhelms that of v_m. For example, when N_s increases to more than 60, the data gathering time for Case II, which has a higher effective data uploading rate as $v_d = 110kbps$, is shorter than that of Case I. This is because that more sensors make the data uploading time dominant in each region and they provide more opportunities to extract the benefit of SDMA technique to the maximum extent.

Fig. 3.5 plots the data gathering time of RDTP varying with D under different settings of N_k, where N_s is set to 50 and N_p is set to 49. We can see that as D increases, the data gathering time increases. The reason of the increase is two fold.

First, the maximum number of compatible pairs among sensors decreases as the sensing area becomes larger since the sparser distribution makes it less possible for any two sensors to be compatible. Second, the distance between any two selected anchor points becomes larger with the increase of D. Apparently, the moving tour of each SenCar may also become longer than that with a smaller D. It is also noticed that the data gathering time is shortened with more available SenCars as data gathering load is shared and balanced among different SenCars. For example, when D is set to $120m$, the data gathering time is equal to $1160sec$ if there is only one available SenCar (i.e., $N_k = 1$). In contrast, the data gathering time drops to $753sec$ and $576sec$ when there are two and three available SenCars, respectively, achieving 35% and 51% improvement with respect to that with a single SenCar, respectively.

3.5 Conclusions

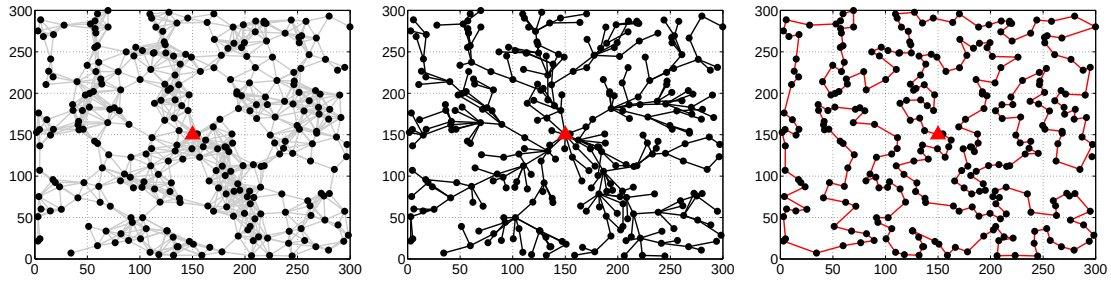
In this chapter, we have considered mobile data gathering in WSNs by applying multiple mobile collectors and SDMA technique. We formalized this problem as MDG-MS problem and proposed a region-division and tour-planning algorithm to provide a practically good solution to the problem. Simulation results demonstrate that our proposed RDTP algorithm can effectively shorten the data gathering latency for large-scale sensor networks compared to other non-SMDA or single mobile collector schemes.

Chapter 4

Bounded Relay Hop Mobile Data Gathering in WSNs

In this chapter, we study the tradeoff between energy saving and data gathering latency in mobile data gathering by exploring a balance between the relay hop count of local data aggregation and the moving tour length of the SenCar. We first propose a polling-based mobile gathering approach and then formulate it into an optimization problem, named *bounded relay hop mobile data gathering (BRH-MDG)*. Specifically, a subset of sensors will be selected as polling points that buffer locally aggregated data and upload the data to the SenCar when it arrives. In the meanwhile, when other sensors are affiliated with these polling points, it is guaranteed that any packet relay for local aggregation is bounded within a given number of hops. We give two efficient algorithms for selecting polling points among sensors. The effectiveness of our approach is validated through extensive simulations.

The rest of this chapter is organized as follows. Section 4.1 provides the background and introduction of this research work. Section 4.2 reviews the related work on some categories of mobile data gathering schemes. Section 4.3 outlines the polling-based approach and formulates the BRH-MDG problem. Sections 4.4 and 4.5 present two algorithms to solve the BRH-MDG problem, respectively. Section 4.6 evaluates the efficiency of the proposed algorithms through extensive simulations. Finally, Section 4.7 concludes the chapter.



(a) 300 sensors deployed over a 300m × 300m field.

(b) Relay routing along shortest paths with minimum hop counts.

(c) Mobile data gathering by visiting each sensor and static data sink.

Figure 4.1: An example to illustrate the tradeoff between energy saving and data gathering latency in a sensor network.

4.1 Introduction

Recent studies [22]-[36] have shown that energy consumption at sensors can be greatly reduced with mobile data gathering, since the mobility of the collector effectively dampens the relay hops of each packet. Intuitively, to pursue maximum energy saving, a SenCar should traverse the transmission range of each sensor in the field so that each packet can be transmitted to the SenCar in a single hop. However, due to the low velocity of the SenCar, it would incur long latency in data gathering, which may not meet the delay requirement of time-sensitive applications.

According to the empirical studies [36][38], the packet relay speed in a WSN is about several hundred meters per second, which is much higher than the velocity at which the SenCar moves. Hence, in general, the latency of multi-hop relay routing and its variants is much shorter than that of the mobile data gathering. Whereas, as aforementioned, mobile data gathering pursues energy saving by simply reducing the relay hops among sensors. From these observations, it is clear that there is an intrinsic tradeoff between the energy saving and the data gathering latency. To better understand this tradeoff, we illustrate it with an example in Fig. 4.1. A network with 300 sensors is configured as shown in Fig. 4.1(a) with the static data sink located at the center of a 300m by 300m field. When we adopt multi-hop routing for data gathering and each packet is forwarded along its shortest path with the minimum hop count to the data sink, the result is depicted in Fig. 4.1(b), where each packet needs 5.3 hops on average to reach the data sink. On the other hand, when a SenCar

is employed, one of the extreme cases for energy saving is that the SenCar gathers data packets by sequentially visiting each sensor, which guarantees that each sensor can directly upload data to the SenCar without any relay. In this way, the number of transmissions is greatly reduced, however, the SenCar has to travel along a tour of 4012m in length as shown in Fig. 4.1(c). Since the typical velocity of a practical mobile system is about 0.1 - 2m/s [39], it will take the SenCar about 66.9 minutes on the tour when it moves at an average speed of 1m/s.

Therefore, in order to shorten data gathering latency, it is necessary to incorporate multi-hop relay into mobile data gathering, however, the relay hop count should be constrained to a certain level to limit the energy consumption at sensors. In this chapter, we would address this issue by proposing a polling-based approach that pursues a tradeoff between the energy saving and data gathering latency, which achieves a balance between the relay hop count for local data aggregation and the moving tour length of the SenCar. Specifically, a subset of sensors will be selected as the *polling points* (PPs), each aggregating the local data from its affiliated sensors within a certain number of relay hops. These PPs will temporarily cache the data and upload them to the SenCar when it arrives.

The main contributions of this work can be summarized as follows. (1) We characterize the polling-based mobile data gathering as an optimization problem, named *bounded relay hop mobile data gathering*, or *BRH-MDG* for short. We then formulate it into an integer linear program (ILP) and prove its NP-hardness. (2) We propose two efficient algorithms to find a set of PPs among sensors. The first algorithm is a centralized algorithm that places the PPs on the shortest path trees rooted at the sensors closest to the data sink, and takes into consideration the constraints on relay hops for local aggregation while shortening the tour length of the SenCar. The second algorithm is a distributed algorithm, where sensors compete to be a PP based on their priorities in a distributed manner. (3) We evaluate the performance of the proposed algorithms by comparing them not only with the optimal solution obtained by CPLEX [94] based on our ILP formulation modeling in AMPL [95], but also with other two existing mobile data gathering schemes. Simulation results demonstrate that the proposed algorithms achieve superior performance.

4.2 Related Work

In this section, we briefly review some recent work on mobile data gathering related to the topic of this work. Based on the mobility pattern, we can divide mobile data gathering schemes into two categories.

The first category has uncontrollable mobility, in which the mobile collector either moves randomly or along a fixed track, see, for example, [22]-[39]. In [22], Shah, et al. proposed to use a special type of mobile nodes as forwarding agents to facilitate connectivity among static sensors and transport data with random mobility. Jain, et al. [12] enhanced the work in [22] by presenting an analytical model to understand the key performance metrics of the systems that exploit the mobility in data collection, such as data transfer, latency to the destination, and power consumptions. Jea, et al. [26] restricted the mobile nodes to move along straight lines to collect data in the vicinity of the lines. In [41][42], radio-tagged zebras and whales were used as mobile nodes in a wild area. Finally, Batalin, et al. [39, 40] set up a system named NIMs, where mobile collectors can only move along fixed cables between trees to ensure that they can be recharged any time during the movement. A common feature of these approaches is that they generally have high stability and reliability, and the system maintenance is simple. However, they typically lack the agility and cannot be adaptive to the sensor distribution and environmental dynamics.

The second category has controlled mobility, in which mobile collectors can freely move to any location in the field and its trajectory can be planned for specific purposes, see, for example, [30]-[36]. Within this category, the schemes can be further divided into three sub-classes. In the first subclass, the mobile collector is controlled to visit each sensor or traverse the transmission range of each sensor and gather the sensing data from them within single hop transmissions [30][28]. Somasundara, et al. [30] studied the scheduling of mobile elements to ensure no data loss due to buffer overflow. While these approaches minimize the energy cost and balance energy consumption among different sensors by completely avoiding multi-hop relays, they may result in long data gathering latency especially in a large-scale sensor network. In the second subclass, mobile collectors gather data from the sensors in the vicinity via multi-hop transmissions along its trajectory. Ma and Yang [27] gave a moving path planning algorithm by finding some turning points, which

is adaptive to the sensor distribution and can effectively avoid obstacles on the path. Along each moving line segment between the turning points, the sensors forward packets to the mobile collector in a multi-hop fashion. Kusy, et al. [43] proposed algorithm for improving routing reliability by introducing the mobility graph, which encodes the knowledge of likely mobility patterns within the network. The mobility graph can be extracted from training data and is used to predict future relay nodes for the mobile node to maintain uninterrupted data streams. Luo and Hubaux [32] proposed that data packets should be gathered with multi-hop relays while the mobile collector moves along the perimeter of the sensing field, which is considered as the optimal path for the mobile collector. Karenos and Kalogeraki [44] explored the congestion and rate allocation problems in mobile data gathering. They provided a new routing alternative that is adaptive to fast reliability fluctuations caused by sink mobility. Xu, et al. [45] studied the event collection problem by leveraging the mobility of the sink node and the spatial-temporal correlation of the event, in favor of maximizing the network lifetime at a guaranteed event collection rate. They modeled the problem as a sensor selection problem and analyzed the design of a feasible moving route for the mobile sink to minimize the velocity requirements for a practical system. Gatzianas and Georgiadis [59] optimized data gathering performance by presenting a distributed maximum lifetime routing algorithm, where a mobile collector sequentially visits a set of anchor points and each sensor transmits data to the mobile collector at some anchor points via energy-efficient multi-hop paths. These approaches can effectively shorten or constrain the moving tour of the mobile collector to a certain level, however, they do not impose any constraint on the relay hop count. As a result, network lifetime (or a certain level of energy efficiency) cannot be guaranteed. The last subclass includes the approaches that jointly consider data transmission patterns and moving tour planning. For instance, Bote, et al. [47] considered utilizing ultra-wide band (UWB) communications for data gathering in WSNs. Adopting a Voronoi diagram based approach, they proposed an algorithm to determine the minimal set of data collection points and the route taken by the mobile node. Xing, et al. [36] proposed a rendezvous design to minimize the distance of multi-hop routing paths for local data aggregation under the constraint that the tour length of the mobile collector is no longer than a threshold. Our work in this chapter falls into this subclass, which aims to minimize the tour length of the mobile collector and guarantee the local data aggregation within

bounded relay hops.

4.3 BRH-MDG Problem

In this section, we first give an overview of the proposed polling-based mobile data gathering scheme and then formulate it into an optimization problem.

4.3.1 Overview

Since the mobile collector, i.e., SenCar, has the freedom to move to any location in the sensing field, it provides an opportunity to plan an optimal tour for it. Our basic idea is to find a set of special nodes referred to as *polling points* (PPs) in the network and determine the tour of the SenCar by visiting each PP in a specific sequence. With sensors properly affiliated with these PPs, the relay routing for local data aggregation can be constrained within d hops, where d is a system parameter for the relay hop bound. Or, alternatively, we can say that a PP covers its affiliated sensors within d hops. The setting of d is based the user-application needs, which reflects how to balance the tradeoff between the energy saving and data gathering latency. For example, when the energy supply of sensors is not sufficient or the data gathering service is somewhat delay-tolerant, we typically set d to a small value. The PPs can simply be a subset of sensors in the network or some other special devices, such as storage nodes [21] with larger memory and more battery power. In the latter case, the storage nodes are not necessarily be placed at the positions of sensors, which may bring more flexibility for the tour planning. However, such special devices would incur a significant amount of extra cost. Therefore, in this chapter, we focus on selecting a subset of sensors as the PPs. Each PP temporarily buffers the data originated from its affiliated sensors. When the SenCar arrives, it polls each PP to request data uploading. Upon receiving the polling message, a PP uploads data packets to the SenCar in a single hop. The SenCar starts its tour from the static data sink, which is located either inside or outside the sensing field, collects data packets at the PPs and then returns the data to the data sink. Since the data sink is the starting and ending points of the data gathering tour, it can also be considered as a special PP. We refer to this scheme as the *polling-based mobile data gathering scheme*. It is further illustrated in Fig. 4.2, where the sensors in the

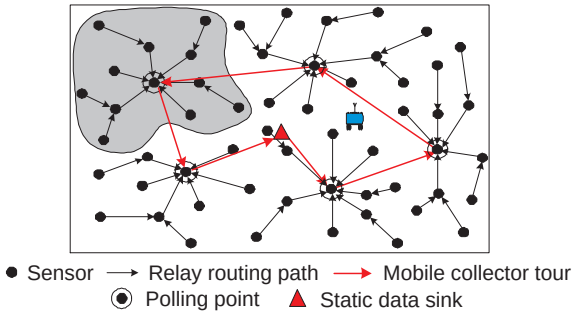


Figure 4.2: Illustration of polling-based mobile data gathering with the relay hop count bounded within two hops, i.e., $(d = 2)$, for local data aggregation.

shadowed area will locally aggregate data packets to their affiliated PP within two hops (i.e, $d = 2$). For generality, we do not make any assumption on the distribution of the sensors or node capability, such as location-awareness. Each sensor is only assumed to be able to communicate with its neighbors, that is, the nodes within its proximity.

In practice, there are several reasons that the relay hop count should be bounded. First, a sensor network may be expected to achieve a certain level of energy efficiency system wide. For instance, if each transmission costs one unit of energy and the energy efficiency of 0.33 energy_unit/packet is expected, each packet should be forwarded from its originating sensor to the data sink in no more than three hops on average, i.e., each packet should be relayed to its PP within two hops. Second, the bound is necessary due to buffer constraint on the sensors. Since the PPs need to buffer the locally aggregated data before the SenCar arrives, it is not desirable to associate too many sensors with a PP. Otherwise, the buffer of the PP may not be able to accommodate all the data packets. For example, consider a sensor network with an average node degree of 4. If a sensor is selected as a PP and the local relaying is constrained within two hops, there will be up to 17 sensors affiliated with this PP. Therefore, the buffer capacity of the PPs and the sensor density impose a limit on relay hops.

4.3.2 BRH-MDG Problem Formulation

Having described the polling-based mobile data gathering scheme, in this subsection, we formulate it into an optimization problem, named *bounded relay hop mo-*

Table 4.1: Notations used in formulation of BRH-MDG Problem.

Indices:		
$\mathcal{S} = \{1, \ldots, N\}$		A set of sensors, which is also the set of candidate PPs.
π		The static data sink.
Constants:		
$d > 0$		The relay hop bound for local data aggregation.
$f_{ij} = \{0, 1\}$	$\forall i, j \in \mathcal{S}$	If sensor i is one-hop neighbor of sensor j, $f_{ij} = 1$, otherwise, $f_{ij} = 0$.
$l_{ij} > 0$	$\forall i, j \in \mathcal{S}$	Distance between sensor i and sensor j.
Variables:		
$I_i = \{0, 1\}$	$\forall i \in \mathcal{S} \cup \{\pi\}$	If sensor i is selected as a PP, $I_i = 1$, otherwise, $I_i = 0$. The static data sink is a special PP, i.e., $I_\pi = 1$.
$a_{iu} = \{0, 1\}$	$\forall i, u \in \mathcal{S}$	If sensor i is on the routing tree rooted at PP (= sensor j), $a_{ij} = 1$, otherwise, $a_{ij} = 0$.
$x^h_{iju} = \{0, 1\}$	$\forall i, j, u \in \mathcal{S}$ $h = 1, 2, \ldots, d$	A node (i, u, h) is associated with sensor i, if the path from which to PP (= sensor u) contains h arcs. If arc $\{(i, u, h-1), (j, u, h)\}$ is contained in the optimal solution, $x^h_{iju} = 1$, otherwise, $x^h_{iju} = 0$. When $i = j$, x^h_{iiu} represents whether sensor i is on the layer h of the routing tree rooted at PP (= sensor u).
$e_{uv} = \{0, 1\}$	$\forall u, v \in \mathcal{S} \cup \{\pi\}$	If the moving tour contains the line segment between u and v, $e_{uv} = 1$, otherwise, $e_{uv} = 0$.
$y_{uv} > 0$	$\forall u, v \in \mathcal{S} \cup \{\pi\}$	The flow from sensor u to sensor v.

bile data gathering, or *BRH-MDG* for short. Our objective is to find a subset of sensors as the PPs and a set of routing paths that connect each sensor in the field to a PP within d hops, such that the tour length of the SenCar can be minimized. The problem is formally defined as follows.

Definition 1. *Bounded Relay Hop Mobile Data Gathering (BRH-MDG) Problem.* Given a set of sensors $\mathcal{S}$ and a relay hop bound d, find (1) A subset of $\mathcal{S}$, denoted by $\mathcal{P}$ ($\mathcal{P} \subseteq \mathcal{S}$), which represents the PPs; (2) A set of geometric trees $\{T_i(V_i, E_i)\}$ that are rooted at each PP in $\mathcal{P}$ and $\bigcup_i V_i = \mathcal{S}$. The depth of each geometric tree is at most d; (3) The data gathering tour U by visiting each PP in $\mathcal{P}$ and the data sink π exactly once, such that $\sum_{(u,v)\in U} |uv|$ is minimized, where $u, v \in \mathcal{P} \cup \{\pi\}$, (u, v) is a line segment on the tour and $|uv|$ is its Euclidean distance.

Apparently, the BRH-MDG problem consists of several subproblems. The first one is the affiliation pattern of the sensors, which can be stated as: by which PP a sensor is covered within the relay hop bound. The second one is how to construct a routing tree rooted at a particular PP with depth at most d that connects all its affiliated sensors. The third one is to find a shortest round trip among the PPs and the static data sink, which is exactly the Traveling Salesman Problem (TSP).

The challenge of the BRH-MDG problem is that these three subproblems should be jointly considered in order to find optimal PPs among sensors. Based on these subproblems and using the notations in Table 4.1, the BRH-MDG problem can be formulated as the following integer linear program.

$$\textbf{Minimize} \sum_{u,v \in \mathcal{S} \cup \{\pi\}, u \neq v} l_{uv} e_{uv} \tag{4.1}$$

Subject to

$$a_{iu} \leq I_u, \forall i, u \in \mathcal{S} \tag{4.2}$$

$$\sum_{u \in \mathcal{S}} a_{iu} = 1, \forall i \in \mathcal{S} \tag{4.3}$$

$$\sum_{i \in \mathcal{S}} a_{iu} \geq I_u, \forall u \in \mathcal{S} \tag{4.4}$$

$$a_{uu} = I_u, \forall u \in \mathcal{S} \tag{4.5}$$

$$x^h_{iju} <= I_u, \forall i, j, u \in \mathcal{S}, h = 1, 2, \ldots, d \tag{4.6}$$

$$x^h_{uuu} = 0, \forall u \in \mathcal{S}, h = 1, 2, \ldots, d \tag{4.7}$$

$$x^h_{iju} \leq \frac{1}{2}(a_{iu} + a_{ju}) \cdot f_{ij}, \forall i, j, u \in \mathcal{S}, i < j, h = 1, 2, \ldots, d \tag{4.8}$$

$$x^1_{iiu} = x^1_{uiu} = a_{iu} \cdot f_{iu}, \forall i, u \in \mathcal{S} \tag{4.9}$$

$$\sum_{h=1}^{d} \sum_{u \in \mathcal{S}} x^h_{iiu} = 1 - I_u, \forall i \in \mathcal{S} \tag{4.10}$$

$$x^h_{iiu} = \sum_{j \in \mathcal{S}, i \neq j} x^h_{jiu}, \forall i, u \in \mathcal{S}, i \neq u, h = 1, 2, \ldots, d \tag{4.11}$$

$$x^h_{iju} \leq 0.5 \cdot \left(x^{h-1}_{iiu} + x^h_{jju} \right), \forall i, j \in \mathcal{S}, i \neq j, h = 2, \ldots, d \tag{4.12}$$

$$\sum_{h=1}^{d} \sum_{i,j \in \mathcal{S}, i \neq j} x^h_{iju} = \sum_{i \in \mathcal{S}, i \neq u} a_{iu}, \forall u \in \mathcal{S} \tag{4.13}$$

$$\sum_{v \in \mathcal{S} \cup \{\pi\}} e_{uv} = I_u, \forall u \in \mathcal{S} \cup \{\pi\} \tag{4.14}$$

$$\sum_{u \in \mathcal{S} \cup \{\pi\}} e_{uv} = I_v, \forall v \in \mathcal{S} \cup \{\pi\} \tag{4.15}$$

$$y_{uv} \leq (|\mathcal{S}| + 1) \cdot e_{uv}, \forall u, v \in \mathcal{S} \cup \{\pi\}, u \neq v \tag{4.16}$$

$$\sum_{u \in \mathcal{S} \cup \{\pi\}, u \neq \pi} y_{u\pi} = \sum_{u \in \mathcal{S} \cup \{\pi\}} I_u \tag{4.17}$$

$$\sum_{v \in \mathcal{S} \cup \{\pi\}, u \neq v} y_{uv} - \sum_{w \in \mathcal{S} \cup \{\pi\}, w \neq u} y_{wu} = I_u, \forall u \in \mathcal{S} \cup \{\pi\} \tag{4.18}$$

In the above formulation, objective function (4.1) minimizes the tour length of the SenCar, which also implies the shortest latency for data gathering. The constraints are explained as follows.

Constraints (4.2)-(4.5) for subproblem 1: These constraints ensure that a sensor should be affiliated with (or covered by) one and only one PP such that its sensing data can be collected during the tour. A sensor selected as a PP will be affiliated with itself.

Constraints (4.6)-(4.13) for subproblem 2: Since the relay hop bound d limits the number of hops between the root PP and its affiliated sensors, each geometric tree can be considered as having at most d layers. Sensor i is associated with a triple (i, u, h) and variable x^h_{iiu} indicates whether sensor i is in layer h ($1 \leq h \leq d$) of the tree rooted at sensor u. Variable x^h_{iju} ($i \neq j$) is associated with arc $\{(i, u, h-1), (j, u, h)\}$, which indicates whether arc (i, j) is on the geometric tree rooted at sensor u with sensors i and j in layers $h-1$ and h, respectively. Constraints (4.6)-(4.7) guarantee that each sensor can only be associated with the tree rooted at a PP and exclude the sensors selected as the PPs since they are automatically the roots and will not be in any layer of a tree (or can be considered in layer 0). Constraints (4.8)-(4.9) address that only if two neighboring sensors, say, i and j, are simultaneously affiliated with the same PP u, arc $\{(i, u, h-1), (j, u, h)\}$ is qualified to be on the tree in the optimal solution. For the special case that sensor i is the neighbor of its affiliated root u, sensor i will be in layer 1 of the tree rooted at u, and arc $\{(u, u, 0), (i, u, 1)\}$ is the edge connected u and i on the tree. Constraints (4.10)-(4.12) enforce that each sensor can only be in one layer of a tree and it has only one connection with the sensors in the immediate upper layer to ensure the tree structure. Constraints (4.13) indicates that the number of edges on each tree is equivalent to the number of affiliated sensors excluding the PP itself.

Constraints (4.14)-(4.18) for subproblem 3: Constraints (4.14) -(4.15) guarantee that the SenCar enters and departs each PP as well as the data sink only once. Constraint (4.16) restricts that the network flow can take place only when the arc is on the moving tour. Constraints (4.17)-(4.18) enforce that for each PP, the units of outgoing flow are one unit more than that of the incoming flow. The flow units entering the data sink, which acts as the starting and ending points of the tour, are equal to the number of PPs [28]. It was shown in [89] that constraints (4.16)-(4.18)

can effectively exclude the solution with subtours.

We have the following theorem concerning the BRH-MDG problem.

Theorem 3. *The BRH-MDG problem is NP-hard.*

Proof. The NP-hardness of the BRH-MDG problem can be shown by giving a polynomial-time reduction from TSP problem to a special case of BRH-MDG problem. Given a complete graph $G = (V, E)$ as an instance of TSP. We construct an instance of BRH-MDG on graph $G' = (V', E')$, which is topologically identical to G. V' is the set of vertices that includes all the sensors and the data sink, and E' represents the edges between any two vertices. We assume that the sensors are located such that they are unreachable from each other via wireless transmissions, which can be achieved by reducing the transmission range below a certain level. This reduction is straightforward and can certainly be done in polynomial time. Now, in this case it is infeasible for the data packets of a sensor to be relayed by others. The SenCar has to visit each sensor to gather data packets, which implies that all the sensors and the data sink are the PPs. Hence, the tour length of the data gathering in G' corresponds to the total cost of the TSP in G. The TSP in G will have a path with minimum cost in distance if and only if the same path in G' is the tour with the minimum length for BRH-MDG. Thus, the BRH-MDG problem is NP-hard. □

4.4 Centralized Algorithm for BRH-MDG Problem

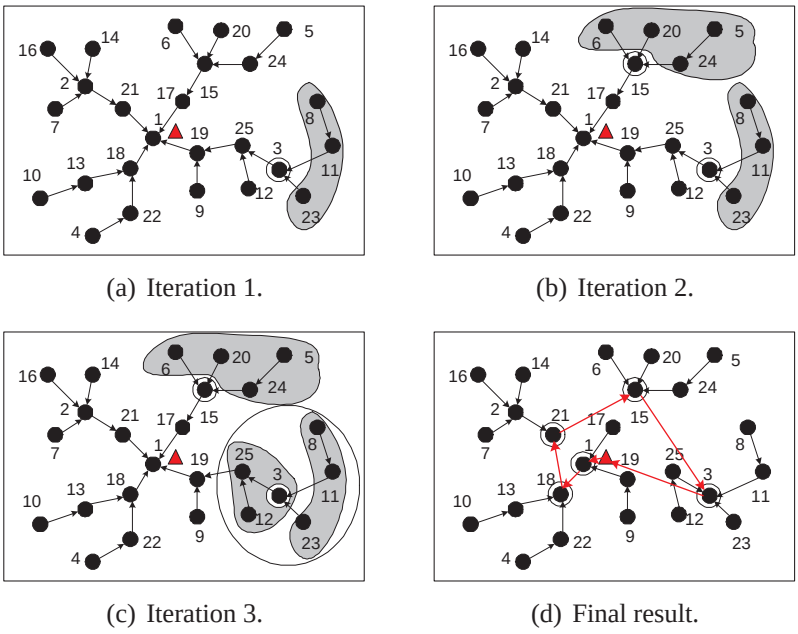
Due to the NP-hardness of the BRH-MDG problem, in this section, we first develop a centralized heuristic algorithm for the BRH-MDG problem. It will serve as a basis for the distributed algorithm in the next section. It is worth pointing out that the solution exploration procedure for the algorithms only needs to be executed when the network topology updates or the relay hop bound changes, thus does not need to be frequently repeated.

As discussed earlier, in order to find optimal PP locations among sensors, relay routing paths and the tour of the SenCar should be jointly considered. On one hand, when no SenCar is employed, for each sensor, the best way to relay data packets to the static data sink is along its shortest path with the minimum hop count, under the assumption that energy consumption is proportional to the number of transmissions. On the other hand, when a SenCar is available, the data gathering tour can be

effectively shortened in two ways: First, the sensors selected as the PPs are compactly distributed and close to the data sink. Second, the number of the PPs is the smallest under the constraint of the relay hop bound. Based on these observations, we propose an algorithm, named *shortest path tree based data gathering algorithm (SPT-DGA)* with its pseudo code listed in Algorithm 1. The basic idea of the algorithm is to iteratively find a PP among the sensors on a shortest path tree (SPT), which is the nearest sensor to the root that can connect the remote sensors on the tree. Also, each PP strives to link as many as possible sensors it can reach within the relay hop bound in order to minimize the total number of PPs.

The first task of SPT-DGA is to construct SPTs that cover all sensors in the network (see Algorithm 1, line 1). Since the network can be disconnected, there may exist more than one SPTs when the sensors are sparsely distributed. Considering this, when we find a root for the SPT to be constructed, we will choose the sensor closest to the data sink from the sensors not on the existing SPTs. We call such a sensor *centroid.* The reason why we choose the centroid rather than a random sensor as the root is that we want the PPs to converge towards the static data sink. Each SPT would link all the possible sensors under the connectivity restriction. This way, our scheme can be applied to not only connected networks, but also disconnected networks, which is one of the main advantages of the mobile data gathering over the traditional relay routing.

The next task of SPT-DGA is to iteratively find a PP on SPTs. We consider the sensor network as a graph $G(V, E)$, where $V = \mathcal{S}$ represents all the sensors in the network, and E is the set of edges connecting any two neighboring sensors. In the following discussion, for clarity and simplicity, we will focus on a single SPT. The algorithm can be described as follows. We consider a SPT denoted by $T'(V', E')$ with $V' \subseteq V$ and $E' \subseteq E$. In each step, we first find the farthest leaf vertex v on T'. There are two possible cases for v depending on whether it is already a PP or not. The first case is that v has not been selected as a PP yet (see Algorithm 1, lines 5-15). In this case, T' is traversed along the shortest path of v towards the root to find its d-hop parent vertex. Let u denote the d-hop parent of v. Since v is the vertex with the farthest depth, all other child vertices of u can reach u within d hops. Hence, we can let the corresponding sensor u be the PP found in the current iteration since it is the nearest one to the root that can connect the sensors in the periphery of the network based on the SPT structure. Then T' is updated by removing all the



(a) Iteration 1. (b) Iteration 2.

(c) Iteration 3. (d) Final result.

Figure 4.3: An example to illustrate the SPT-DGA algorithm ($N = 25, d = 2$).

child vertices of u and their pertinent edges, which implies that the corresponding sensors will be affiliated with u for local data aggregation. It is worth pointing out that we still keep u on the updated T' in order to facilitate the possible affiliations of other nearby sensors with u in future iterations. In the rare case that the root of T' was reached during the process of finding the d-hop parent vertex of v, the algorithm terminates since all the vertices on current T' are definitely within d hops to the root. Correspondingly, the root will be selected as the PP. The second case is that the farthest leaf vertex v on current T' has already been selected as a PP (see Algorithm 1, lines 16-28). In this case, we aim to affiliate more sensors with v if possible in order to reduce the number of PPs. Specifically, in order to find more sensors in the vicinity of v, we first find v's $\lfloor \frac{d}{2} \rfloor$-hop parent vertex w. As v is the farthest leaf vertex on current T', all other child vertices of w will be within $\lfloor \frac{d}{2} \rfloor$ hops away from w so that they are able to reach v within d hops along the edges on T'. Hence, besides the existing affiliated sensors of v, the sensors on the subtree rooted at w can also be affiliated with v. Thus, all the affiliated sensors of a PP will be found in these two steps. The inherited edges among these sensors from T' will be used to determine their relay paths to the affiliated PP for local data aggregation.

To better understand the algorithm, we give an example in Fig. 4.3, where 25 sensors are scattered over a field with the static data sink located in the center of the area, and d is set to 2, which means that it is required for each sensor to forward its data to the affiliated PP within two hops. The constructed SPT among the sensors rooted at sensor 1, denoted by T', is depicted in Fig. 4.3(a). In the first iteration, sensor 8 is found as the farthest leaf vertex on T' with 5 hops away from the root, i.e., $v = 8$. Its 2-hop parent vertex u on current T' is sensor 3 (i.e., $u = 3$.), which will be marked as a PP. All the child vertices of u, including sensors 8, 11 and 23, and their associated edges will be removed from T'. The result is depicted in Fig. 4.3(a), where sensor 3 is still kept on the updated SPT, and the removed vertices highlighted by the shadowed area are its affiliated sensors found in the current iteration. In the second iteration, the farthest leaf vertex on the updated T' turns to be sensor 5 with 4 hops away from the root. Similarly, its 2-hop parent (i.e., sensor 15) is selected as another PP to cover the sensors in the other shadowed area as shown in Fig. 4.3(b). In the third iteration, sensor 3 is chosen as the farthest leaf vertex on current T' and it happens to be marked as a PP already. In this case, we strive to search for more qualified sensors to affiliate with it. We find that sensor 25 is its 1-hop parent, i.e., $w = 25$. Sensor 25 and all its child vertices on current T' can reach sensor 3 within two hops along the edges on T'. Therefore, the subtree rooted at sensor 25 will be pruned from T'. All the sensors on the subtree, including sensors 25, 12 and 3, will also be affiliated with sensor 3. Fig. 4.3(c) indicates that a total of 6 sensors will be covered by sensor 3, which are found in iterations 1 and 3, respectively. In this way, T' is decomposed into a set of subtrees, each of which contains a selected PP and its affiliated sensors. Fig. 4.3(d) gives the final result, where the data gathering tour is highlighted by the line segments linking the PPs and the static data sink.

We now analyze the time complexity of SPT-DGA. Assume that there are a total of N sensors distributed in K disconnected subnetworks ($1 \leq K \leq N$). For subnetwork k ($k = 1, 2, \ldots, K$), it takes $\mathcal{O}(N_k^2)$ time to find a SPT [88], where N_k represents the number of sensors in subnetwork k. It takes $\mathcal{O}(N_k^2 + N_k d)$ time to iteratively find a PP and its affiliated sensors on the SPT in subnetwork k. Moreover, the work of finding an approximate shortest tour on the PPs and the data sink can be done in at most $\mathcal{O}(N^2)$ time. Thus, the total time of SPT-DGA is $\sum_{k=1}^{K} [\mathcal{O}(N_k^2) + \mathcal{O}(N_k^2 + N_k d)] + \mathcal{O}(N^2)$. Hence, in the worst case, the time

complexity of SPT-DGA is $\mathcal{O}(N^2 + Nd)$.

4.5 Distributed Algorithm for BRH-MDG Problem

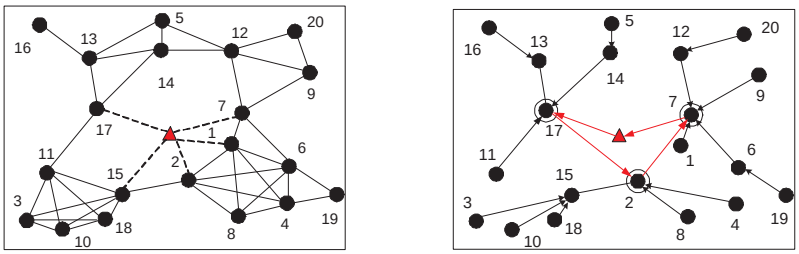
Given the complete knowledge of sensor distribution, the centralized SPT-DGA algorithm can work well in finding a good data gathering tour. However, in practice, such global information is difficult to obtain. In this section, we propose a distributed algorithm searching for suitable sensors as the PPs to achieve better scalability, which follows the same basic idea as the centralized algorithm.

As discussed in the previous section, two factors greatly affect the suitability of a sensor to be a PP. One is the number of sensors within its d-hop range and the other is its distance to the data sink. A sensor that can cover more sensors in its d-hop neighborhood and is close to the data sink will be more favorable to be a PP since it leads to a smaller total number of PPs and more compacted distribution among the PPs. Considering these factors, we propose an algorithm named *priority based PP selection algorithm,* or *PB-PSA* for short. Two parameters are used to prioritize each sensor in the network, which can be easily obtained in a distributed manner. The primary parameter is the number of d-hop neighbors, which are the sensors in its d-hop range. The secondary parameter is the minimum hop count to the data sink. The basic idea of PB-PSA is that each sensor uses the primary parameter to select an initial set of sensors as its preferred PPs, and then uses the secondary parameter to "break ties." A tie in this context means that the preferred PPs of a sensor have the same number of d-hop neighbors.

We now describe PB-PSA in more detail. The pseudo code for each sensor is given in Algorithm 2. Before a sensor makes the decision on whether it becomes a PP, d rounds of local information exchange are performed to ensure that each sensor can gather the node information in its d-hop neighborhood. In each round, each sensor locally maintains a structure, named TENTA_PP, based on the information exchange. TENTA_PP is the selected sensor temporarily considered as a preferred PP in a particular round by the sensor. TENTA_PP has three sub-domains: TENTA_PP.ID, TENTA_PP.d_Nbrs and TENTA_PP.Hop which denote the node identification, the number of its d-hop neighbors and the minimum hop count of the tentative PP to the data sink, respectively. Initially, each sensor treats itself as its TENTA_PP and labels its status as "Tentative." In a particular round, each sensor

first broadcasts the information of its TENTA_PP to its 1-hop neighbors. When it has heard from all the neighbors, the sensor will update its TENTA_PP according to the following rule: among the pool of all the received TENTA_PPs and its own TENTA_PP, choose the one with maximum TENTA_PP.d_Nbrs to set it as its updated TENTA_PP. If there are more than one such TENTA_PP, choose the one with minimum TENTA_PP.Hop. Such treatment implies that a sensor having the ability to cover more other sensors and also being close to the data sink has higher chance to be a PP than others. After d rounds of iterations are completed, each sensor is able to tell whether it is the one with the highest priority among its d-hop neighbors. If a sensor finds that its TENTA_PP is still itself after d rounds of information exchange, it will declare to be a PP instantly by sending out a declaration message and change its status accordingly. This message will then be propagated up to d hops. For other sensors still with "Tentative" status, they will be delayed for a period of time. The delay time for a sensor consists of a major part proportional to its hop count to the data sink plus a small random time duration to differentiate the sensors with the same hop count. During the delay period, a sensor keeps listening and receiving the declaration messages from others. Once its own delay timer expires, a sensor with "Tentative" status will check whether it has received any declaration message. If yes, the sensor will affiliate itself with the nearest PP among those whose declaration messages are received. Otherwise, the sensor itself will declare to be a PP since there is no PP in its d-hop neighborhood for the moment. This way, the sensors with "Tentative" status closer to the data sink will become a PP ahead of others due to the shorter delay, which effectively refrains other sensors with "Tentative" status from declaring to be the unwanted PPs.

To better understand the PB-PSA, we give an example as shown in Fig. 4.4 where there is a total of 20 sensors and the data sink is assumed to be located at the center of the area. The connectivity among the sensors and the data sink is shown by the links between neighboring nodes in Fig. 4.4(a). We set d to 2, which implies that each sensor needs to do two rounds of local data exchange. Every sensor updates its TENTA_PP based on the received information, and the result in each round is listed in Table 4.2. When the iterations are completed, sensors 2, 7 and 17 find that they are the TENTA_PPs for themselves and consequently send out the declaration messages to claim to be the PPs. During the delay period, all other sensors can receive some declaration messages. Thus, there will be no other PPs.



(a) Network configuration. (b) Moving Tour along the PPs.

Figure 4.4: An example to illustrate the PB-PSA algorithm ($N = 20, d = 2$).

In the next step, each sensor with "Tentative" status will choose to be affiliated with a PP among those it has heard from, which will not necessarily be constrained to the current TENTA_PP of the sensor. The final PPs, the sensors' affiliation pattern and the data gathering tour are depicted in Fig. 4.4(b).

Finally, we give the following two properties concerning the complexity of the PB-PSA algorithm.

Property 1. *PB-PSA has the worst-case time complexity of* $O(Nd)$ *per node, where* N *is the number of sensors.*

Proof. Each sensor first experiences d rounds of iterations. In each iteration, it takes at most $O(N)$ time for a sensor to gather the information of TENTA_PPs from its one-hop neighbors. Except the sensors that declare themselves to be the PPs once the iterations complete, each of other sensors will delay for a period time. Since the delay time is proportional to the minimum hop count of a sensor to the data sink, in the worst case, it takes $O(N \times T_s)$ time for a sensor to finally determine its status, where T_s is a pre-defined constant time slot length. Hence, the total time complexity of the PB-PSA is $O(Nd)$ per node. □

Property 2. *PB-PSA has the worst-case message exchange complexity of* $O(N+d)$ *per node.*

Proof. During the execution of iterations in the PB-PSA, each sensor generates d messages to broadcast its current TENTA_PPs. Once a sensor reaches its final status, either a PP or a regular sensor, it will generate a declaration message to claim to be a PP or a joining message for affiliation. Since each declaration message or joining message will be propagated up to d hops, in the worst case, a sensor may

forward up to $2(N-1)$ messages. Thus, the total number of messages a sensor handles is at most $d+1+2(N-1)$, i.e., the message exchange complexity of the PB-PSA is $O(N+d)$ per node. □

4.6 Performance Evaluation

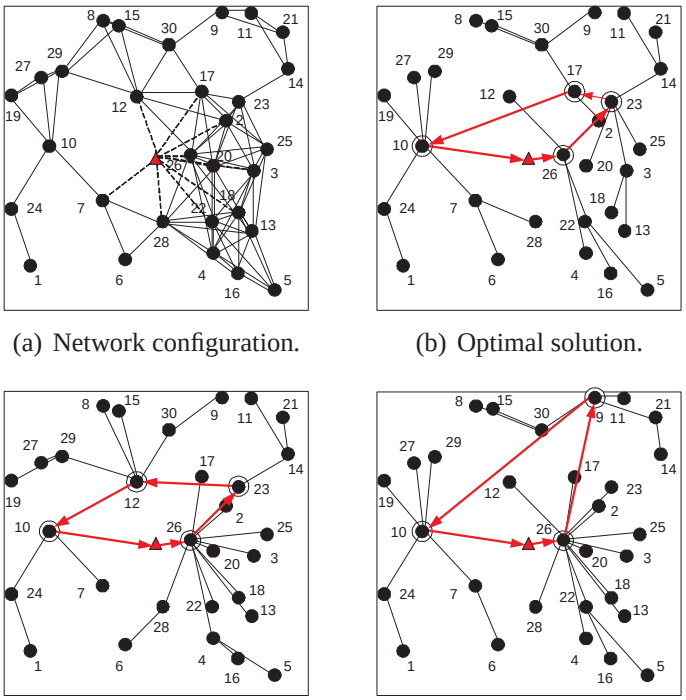
In the previous sections, we have provided two efficient algorithms for the BRH-MDG problem. To evaluate their performance, in this section, we first implement the ILP formulation given in Section 4.3 for a small network as an illustrative example and compare the optimal solution with the proposed algorithms, and then we conduct extensive simulations in large networks and compare the results of the proposed algorithms with other two existing mobile data gathering schemes.

4.6.1 Comparison with the Optimal Solution

We have solved the ILP formulation of BRH-MDG problem given in Section 4.3 for a sensor network with 30 nodes by using CPLEX [94]. We now compare this optimal solution with the results of the proposed algorithms.

As shown in Fig. 4.5(a), a network with 30 sensors scattered over a 70m $\times$ 70m square area. The connectivity is represented by solid links between neighboring sensors. The static data sink is located at the center of the area and the connectivity between the sink and the sensors is plotted by dashed links. d is set to 2. The results of different solutions, each of which contains the selected PPs, the relay routing trees for local data aggregation rooted at PPs and the moving tour of the SenCar, are shown in Fig. 4.5(b)-(d), respectively. Moreover, the performance comparison is summarized in Table 4.3.

From Fig. 4.5 and Table 4.3, we can see that the optimal solution for the example achieves the shortest tour length of 94.78m at the expense of 1.27 relay hops on average for local data aggregation. In contrast, SPT-DGA and PB-PSA result in 3% and 24% longer tour length, however, 7.8% and 11% less average relay hop count, respectively. These observations further reveal the intrinsic tradeoff between the tour length and the relay hop count. Since the distributed PB-PSA algorithm commits itself to dampen the number of PPs by prioritizing the sensors with their d-hop neighbors and overhearing the declaration messages propagated in the d-hop prox-



(a) Network configuration. (b) Optimal solution.

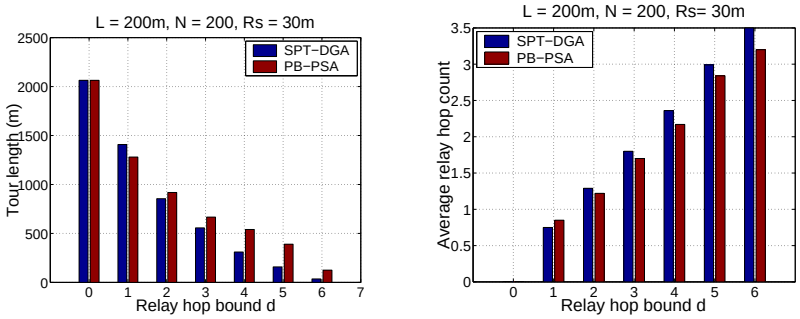
(c) SPT-DGA. (d) PB-PSA.

Figure 4.5: Different solutions for the BRH-MDG problem with d set to 2 in a 30-node network.

imity, it results in the smallest number of PPs compared to others. Apparently, this will also lead to the most average affiliated sensors for a PP for a given number of sensors. SPT-DGA achieves the same number of PPs as the optimal solution, thus, they have the same number of average affiliated sensors for a PP. However, due to the structural restriction of the shortest path tree, the affiliation pattern among sensors in SPT-DGA is not as uniform as that of the optimal solution, which results in 55% more maximum number of affiliated sensors to a PP.

4.6.2 Performance of SPT-DGA and PB-PSA

We have also conducted a suite of simulations to evaluate the performance of our proposed algorithms in large sensor networks. In this subsection, we present the simulation results and compare them with other two existing mobile data gathering schemes. The first scheme is the single-hop data gathering (SHDG) [28], in which a SenCar stops at some selected points among a set of predefined candidate positions to collect data from each sensor such that single-hop data uploading from each sensor to the SenCar can be guaranteed. Another scheme is the controlled mobile element scheme (CME) [26], where a SenCar traverses the sensing field along par-



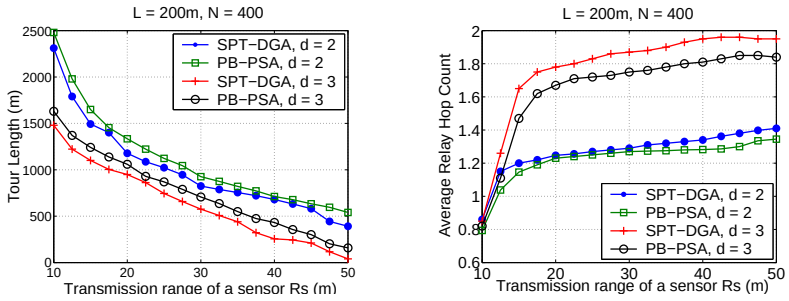
(a) Tour length. (b) Average relay hop count.

Figure 4.6: Performance of SPT-DGA and PB-PSA as a function of d.

allel straight tracks and collects data from the sensors nearby with multi-hop relays. For clarity, we list the comparisons between the compared work and our proposed polling-based approach in Table 4.4.

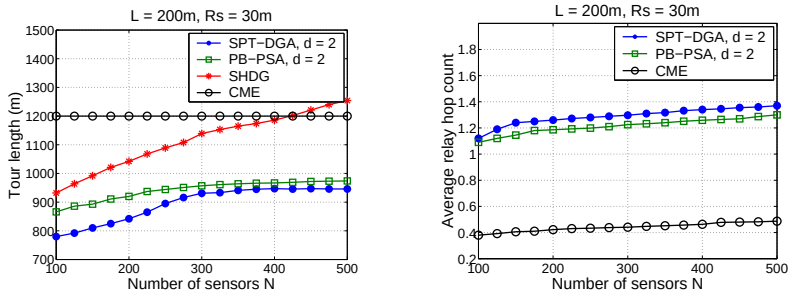
In the simulation, we consider a generic sensor network with N sensors randomly distributed over an $L \times L$ square area. The data sink is located at the center of the area. The transmission range of a sensor is R_s. Each packet is locally aggregated to a PP within the relay hop bound d before the SenCar arrives. If not specified otherwise, d is set to 2. We adopt the nearest neighbor (NN) algorithm [88] in our simulation for the TSP problem to determine the moving tour, which lets the SenCar start from the data sink and choose the nearest unvisited PP for the next visit, and finally return to the data sink. Considering the randomness of the network topology, each performance point in the figures is the average of the results in 500 simulation experiments.

Fig. 4.6 plots the performance of SPT-DGA and PB-PSA as a function of d, in terms of tour length and average relay hop count for local data aggregation. When d is set to zero, it means that the SenCar will visit the sensors one by one for data gathering. N and L are 200 and 200m, respectively. R_s is equal to 30m. From the figure, we can see that as d becomes larger, the tour length is evidently shortened and the average relay hop count gradually increases in both algorithms. Furthermore, in most cases in Fig. 4.6(a), SPT-DGA always outperforms PB-PSA with about 39% shorter tour length on average, and such superiority becomes even more noticeable as d increases. There are two reasons for this. First, since sensors are densely deployed in the scenario under consideration, there are quite a few



(a) Tour length. (b) Average relay hop count.

Figure 4.7: Performance of SPT-DGA and PB-PSA as a function of R_s for the cases of $d = 2$ and $d = 3$.

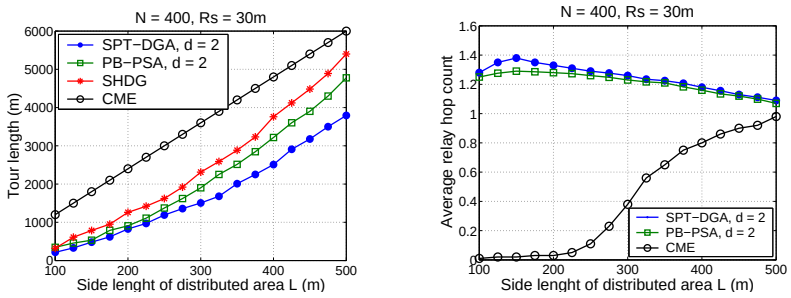


(a) Tour length. (b) Average relay hop count.

Figure 4.8: Performance comparison for SPT-DGA, PB-PSA, SHDG and CME as a function of N.

sensors scattered around the sink. This provides a good opportunity to build a SPT rooted at a sensor close to the sink in the centralized SPT-DGA algorithm. As a result, with the increase of d, the selected PPs become convergent towards the data sink and also get closer to each other. Another reason is that in the distributed PB-PSA algorithm, though the number of total PPs drops as d increases, some sensors that are still in "Tentative" status after d rounds of iterations tend to claim themselves to be the PPs with higher probability. Some of them may be located far from other PPs such that the tour length in PB-PSA is somewhat longer than that of SPT-DGA. We can also observe in Fig. 4.6(b) that PB-PSA results in a smaller average relay hop count as compared to SPT-DGA since the PPs in PB-PSA are distributed in a more relaxed pattern.

Fig. 4.7 shows the performance of SPT-DGA and PB-PSA as a function of R_s



(a) Tour length. (b) Average relay hop count.

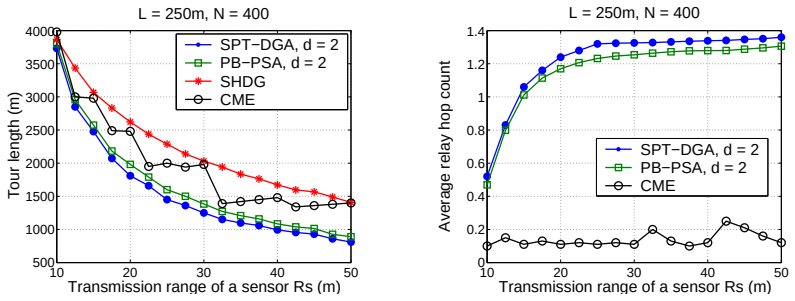
Figure 4.9: Performance comparison for SPT-DGA, PB-PSA, SHDG and CME as a function of L.

for the cases of $d = 2$ and $d = 3$. R_s varies from 10m to 50m to stand for different transmission ranges of sensors. Clearly, more sensors will become neighbors to each other as R_s increases. As a result, the situation that most of the PPs are far away from the sink can be avoided and the number of PPs can also be effectively reduced as each PP is able to link up more sensors. Therefore, the tour length will be greatly shortened as R_s becomes larger. For instance, as shown in Fig. 4.7(a), for the case of $d = 2$, when R_s is 20m, the tour length of SPT-DGA and PB-PSA is 1178m and 1334m, respectively. In contrast, when R_s increases to 45m, their tour length drops to 591m and 634m, respectively. It is also noticed that the average relay hop counts for both algorithms slightly increase with R_s since more sensors will be affiliated with the same PP with a larger hop count under the constraint of the relay hop bound. Moreover, it is evident that no matter what R_s is, the case of $d = 3$, which is with relatively looser relay hop bound, would result in much shorter tour length compared to the case of $d = 2$ at the expense of more average relay hop counts.

Fig. 4.8 depicts the performance of the proposed algorithms as a function of N, and compares it with SHDG and CME. L is set to 200m and N varies from 100 to 500 to represent different node density. R_s is still fixed to 30m. The relay hop bound d is set to 2 for the proposed algorithms. For SHDG, we assume that each predefined position where the SenCar could stop for data gathering is on a grid, which is apart from its adjacent positions in horizontal and vertical directions with the same distance of 20m. Also, in the CME scheme, we assume that the parallel straight tracks traversing the field are 100m apart from each other. The track in the

middle goes through the center of the field. The SenCar can go along the area border to change onto other tracks. Both SHDG and CME schemes are implemented in a centralized fashion. We can observe in Fig. 4.8(a) that as N increases, the tour length of SPT-DGA and PB-PSA first gradually increases and then stabilizes when N becomes sufficiently large. This is because that when sensors become more densely scattered, they will have higher probability to be affiliated with a PP close to the sink. Hence, any further increase on the number of sensors will have little impact on the selection of the preferred PPs. In contrast, the tour length of SHDG continuously increases with N, which is around 33% longer than that of SPT-DGA. Since the SenCar travels along the fixed tracks in a given area, the tour length will stay constant for CME. In addition, Fig. 4.8(b) shows the average relay hop counts for SPT-DGA, PB-PSA and CME. Since a sensor always directly uploads data to the SenCar in SHDG, there is no local relay required. Thus, we did not include it in the figure. We can see that the average relay hop count slightly increases for each scheme as N becomes large. SPT-DGA and PB-PSA result in more relay hops for local data aggregation compared to CME, which is the cost to achieve a shorter data gathering tour.

Fig. 4.9 further plots the tour length and the average relay hop counts obtained for different schemes when L varies from 100m to 500m. N is set to 400 and R_s is 30m. We fix 5 parallel straight tracks with the same interval distance in CME scheme, which traverse the field with the outermost two tracks on the border of the area. All other settings are kept unchanged as in the previous set of simulations. From Fig. 4.9(a), we can see that as L increases, the tour length of all the schemes becomes longer. This is reasonable since sensors become more sparsely distributed as L becomes larger. The SenCar needs to go further away from the sink and visit more positions to collect data from all the sensors. Also, with the increase of the field area, the fix track traversing the field in CME scheme also becomes longer than the case with a smaller L. However, as can be seen, our proposed algorithms always outperform others, with up to 38% and 80% shorter tour length compared with SHDG and CME, respectively. This attributes to the effort in SPT-DGA and PB-PSA algorithms on minimizing the tour length by fully utilizing the available relays for local data aggregation. Consequently, as the expense, in Fig. 4.9(b) the average relay hop counts for SPT-DGA and PB-PSA algorithms are higher than that of CME. However, such a gap in the relay hop count quickly shrinks as L increases.



(a) Tour length. (b) Average relay hop count.

Figure 4.10: Performance comparison for SPT-DGA, PB-PSA, SHDG and CME as a function of R_s.

Fig. 4.10 plots the performance of different schemes when R_s varies from 10m to 50m. N is set to 400 and L is fixed to 250m. The relay hop bound d is set to 2 for the proposed algorithms. For the SHDG scheme, we still assume that the candidate positions that the SenCar can stop for data gathering are on the grids and each one is apart from its neighbors in the vertical and horizontal directions with the same distance of $1.4R_s$. For the CME scheme, we set the distance between any two neighboring parallel straight tracks to $2R_s$ and there is a total of $\left\lceil \frac{L}{2R_s} \right\rceil$ parallel tracks in the field. Fig. 4.10(a) shows that in all cases of R_s investigated, SPT-DGA always achieves the shortest tour length with up to 45% improvement compared to SHDG and CME. It is also noticed that while the tour length of SPT-DGA, PB-PSA and SHDG gradually decreases with the increase of R_s, CME results in a stair-case decrease. This is because that the tour length of CME is mainly determined by the number of parallel tracks. Fig. 4.10(b) plots the average relay hop count for SPT-DGA, PB-PSA and CME. As the parallel tracks in CME are $2R_s$ apart from each other, regardless of the value of R_s, most sensors can directly upload data to the SenCar when it moves along the nearest tracks and comes close enough to the sensors. Thus, CME always results in small relay hops for local data aggregation. In contrast, SPT-DGA and PB-PSA allow more local data aggregation with bounded relay hops in order to shorten the tour length as much as possible. It is shown in the figure that the average relay hop counts achieved in SPT-DGA and PB-PSA increase with R_s initially. This is because that when the transmission range of sensors is small, the degree of sensors would evidently increase as R_s increases. This provides us much opportunity to shorten the moving tour length by selecting

a fewer number of PPs. As there are limited PPs, more remote sensors are likely to be linked by the PPs with a larger number of hops under the constraint of relay hop bound. Then when R_s becomes large, the average relay hop counts would not increase further with R_s and tend to be relatively stable. This is reasonable because when R_s becomes large, it is much easier for sensors to reach each other within a fewer number of relay hops, which greatly counteracts the impact of the decrease of the number of PPs.

4.7 Conclusion

In this chapter, we have studied improving the efficiency of mobile data gathering by exploring the tradeoff between the relay hop count of sensors for local data aggregation and the tour length of the SenCar. We have proposed a polling-based scheme and formulated it into the BRH-MDG problem. We then presented two efficient algorithms for solving the BRH-MDG problem. Extensive simulations have been carried out to validate the efficiency of the scheme. The results demonstrate that the proposed algorithms can greatly shorten the data gathering tour length with a small relay hop count, and achieve 38% and 80% improvement on the tour length compared to SHDG and CME schemes, respectively.

Algorithm 1: Centralized algorithm: SPT-DGA

Input: A sensor network $G(V, E)$, the relay hop bound d, and the static data sink π.
Output: A set of PPs $\mathcal{P}$, a set of geometric trees $\{t_u | u \in \mathcal{P}\}$, and the tour U visiting the PPs and the data sink.

Construct SPTs for G that cover all the vertices in V;
for *each SPT* $T'(V', E')$ **do**
  while T' *is not empty* **do**
    Find the farthest leaf vertex v on T';
    if v *is not a PP* **then**
      // Find v's d-hop parent vertex u on T'.
      for $i = 1$ *to* d **do**
        $u \longleftarrow$parent(v); $v \longleftarrow u$;
        if *u is the root of* T' **then** Break;
      Consider u as a PP and add corresponding sensor into $\mathcal{P}$;
      if u *is not the root of* T' **then**
        Update T' by removing all the child vertices of u and the pertinent edges.
        Corresponding sensors of these removed vertices are affiliated with u on the geometric tree t_u.
      else
        All the sensors on T' are affiliated with t_u;
        T' is set to empty;
    else
      if $d = 1$ **then**
        Remove v from current T' and it belongs to t_v;
      else
        // Find v's $\lfloor \frac{d}{2} \rfloor$-hop parent vertex w on T'.
        for $i = 1$ *to* $\lfloor \frac{d}{2} \rfloor$ **do**
          $w \longleftarrow$parent(v); $v \longleftarrow w$;
          if *w is the root of* T' **then** Break;
        if *w is not the root of* T' **then**
          Remove the subtree rooted at w from T';
          Corresponding sensors on the removed subtree are affiliated with v on the geometric tree t_v.
        else
          All the sensors on T' that are not selected as PPs are affiliated with v on the geometric tree t_v;
          T' is set to empty;
Find an approximate shortest tour U visiting π and all the PPs in $\mathcal{P}$;

Algorithm 2: Distributed Algorithm: PB-PSA

My.TENTA_PP $\longleftarrow$ My; My.status $\longleftarrow$ Tentative;
for $i = 1$ *to* d **do**
    Send_Msg (My.TENTA_PP);
    if *received packets from all my* 1*-hop neighbors* **then**
        $A \longleftarrow \{a|$ is a received or own TENTA_PP with maximum a.d_Nbrs$\}$;
        My.TENTA_PP $\longleftarrow \arg\min_{a \in A}$a.Hop;
if *My.TENTA_PP.ID = My.NodeID* **then**
    My.status$\longleftarrow$ PP;
    Send_Declar_Msg (My.NodeID, My.status, 0);
else
    Set my delay timer $t =$My.Hop$\times T_s \pm$rand($\triangle$);
    while *My delay timer is not expired* **do**
        Record source node IDs of the received declaration messages;
        Forward the received declaration messages if it has been propagated for less than d hops;
    if *ever received a declaration message* **then**
        My.PP $\longleftarrow$ the nearest PP among all the PPs in the received declaration messages;
        My.status $\longleftarrow$ non_PP;
        Send_Join_Msg (NodeID, My.PP);
    else
        My.status $\longleftarrow$ PP;
        Send_Declar_Msg (My.NodeID, My.status, 0);

Table 4.2: Two round update of TENTA_PP by each sensor in the example.

		1	2	3	4	5	6	7	8	9	10
Initial	TENTA_PP.ID	1	2	3	4	5	6	7	8	9	10
	TENTA_PP.d_Nbrs	10	12	7	8	9	10	12	8	8	7
	TENTA_PP.Hop	1	1	2	2	3	2	1	2	2	2
Round 1	TENTA_PP.ID	2	2	15	2	12	2	7	2	7	15
	TENTA_PP.d_Nbrs	12	12	11	12	10	12	12	12	12	11
	TENTA_PP.Hop	1	1	1	1	2	1	1	1	1	1
Round 2	TENTA_PP.ID	2	2	2	2	7	2	7	2	7	2
	TENTA_PP.d_Nbrs	12	12	12	12	12	12	12	12	12	12
	TENTA_PP.Hop	1	1	1	1	1	1	1	1	1	1
		11	12	13	14	15	16	17	18	19	20
Initial	TENTA_PP.ID	11	12	13	14	15	16	17	18	19	20
	TENTA_PP.d_Nbrs	9	10	7	10	11	5	11	7	7	6
	TENTA_PP.Hop	2	2	2	2	1	3	1	2	3	3
Round 1	TENTA_PP.ID	15	7	17	17	2	13	17	15	6	12
	TENTA_PP.d_Nbrs	11	12	11	11	12	7	11	11	10	10
	TENTA_PP.Hop	1	1	1	1	1	2	1	1	2	2
Round 2	TENTA_PP.ID	2	7	17	7	2	17	17	2	2	7
	TENTA_PP.d_Nbrs	12	12	11	12	12	11	11	12	12	12
	TENTA_PP.Hop	1	1	1	1	1	1	1	1	1	1

Table 4.3: Performance comparison with optimal solution.

	Optimum	SPT-DGA	PB-PSA
Sensors Selected as PPs	$10, 17, 23, 26$	$10, 12, 23, 26$	$9, 10, 26$
Tour Length (m)	94.78	97.56	117.86
Ave. Relay Hop Count	1.27	1.17	1.13
Max Num of Affiliated Sensors to a PP	9	14	15
Ave. Num of Affiliated Sensors to a PP	7.5	7.5	10

Table 4.4: Comparisons among three mobile data gathering schemes.

	Polling-Based Approach (SPT-DGA and PB-PSA)	Single Hop Data Gathering (SHDG)	Controlled Mobile Element Scheme (CME)
Motion Pattern	Controllable Free to go anywhere	Controllable Free to go anywhere	Uncontrollable With fixed moving tracks
Pausing Locations of SenCar for Data Gathering	SenCar pauses at locations of selected sensors (i.e., PPs) and gather buffered data from the PPs	SenCar pauses at selected polling points, which are the locations chosen from a set of candidate pausing locations, and gather data from each nearby sensor in a single hop	Exact pausing locations are not explicitly specified. It is assumed that the SenCar can always gather data while moving along the tracks
Moving Trajectory	Start from the data sink, visit each PP once and go back to data sink	Start from the data sink, visit some locations that cover the transmission range of all the sensors, and finally go back to the data sink	Start from the data sink, go along the parallel straight tracks back and forth, and finally go back to the data sink
Relay for Local Data Aggregation	Multi-hop relays in bounded hops	No local relay	Multi-hop relays without hop bound
Data Uploading	The PPs buffer the local aggregated packets and upload them to mobile collector when it arrives at PPs	Each sensor directly uploads data to the SenCar in a single hop when it arrives within its transmission range	Some sensors close to the tracks upload aggregated packets to the SenCar when it comes

Chapter 5

Pricing-based Network Cost Minimization Algorithm for Mobile Data Gathering in WSNs

In the previous chapters, we mainly focus on proposing a variety of mobile data gathering schemes. In the following two chapters, we shift our effort from the scheme design to the performance optimization on some typical schemes, with the purpose of obtaining in-depth investigation on the impact of some critical system parameters on the entire data gathering performance.

In this chapter, we study the performance optimization of the anchor-based range traversing data gathering [28][34], where a set of locations in the sensing field is chosen as *anchor points* and the SenCar periodically carries out a data gathering tour by visiting each anchor point such that it can traverse the transmission range of all the sensors in the network. We characterize the performance optimization as a cost minimization problem constrained by the channel capacity, the minimum amount of data gathered from each sensor and the bound of total sojourn time at all anchor points. We assume that the cost of a sensor for a particular anchor point is a function of the data amount a sensor uploads to the SenCar during its sojourn time at this anchor point. The network cost is the aggregate of all sensors for all anchor points, which is a direct metric to evaluate the efficiency of the data gathering strategies. In order to provide an efficient and distributed algorithm, we decompose this global optimization problem into two subproblems to be solved by each

sensor and the SenCar, respectively. We show that such decomposition can be characterized as a pricing mechanism, in which each sensor independently adjusts its payment for the data uploading opportunity based on the shadow prices of different anchor points set by the SenCar. Correspondingly, we give an efficient algorithm to jointly solve the two subproblems. Our theoretical analysis demonstrates that the proposed algorithm can achieve the optimal data control for each sensor and the optimal sojourn time allocation for the SenCar, which minimizes the overall network cost. Finally, extensive simulation results further validate that our algorithm achieves lower cost than the compared data gathering strategy.

The remainder of this chapter is organized as follows. Section 5.1 provides the introduction of research issue. Section 5.2 reviews the related work. Section 5.3 introduces the system model and provides the formulation of network cost minimization problem. Section 5.4 decomposes the problem into two subproblems and presents a pricing-based algorithm that jointly solves the two subproblems. Section 5.5 addresses how to solve the subproblem at each sensor. Finally, Section 5.6 presents the simulation results of the proposed algorithm and Section 5.7 concludes the chapter.

5.1 Introduction

In this chapter, we focus on anchor-based range traversing data gathering [28]-[34] and study how to achieve its optimal performance. In such a scheme, the SenCar directly collects data from each sensor in a single hop by visiting the anchor points in the field to traverse the transmission range of the sensors. We characterize data gathering performance by introducing *network cost*, which is a function quantifying the aggregated cost on gathering data from sensors at different anchor points. The "cost" here physically implies the energy consumption or monetary expense on gathering a certain amount of data from a sensor at a particular anchor point. In this way, optimizing data gathering performance is equivalent to solving the corresponding cost minimization problem. To find the optimal solution to this problem, we consider regulating two tunable parameters under given constraints. One parameter is the amount of data a sensor uploads to the mobile collector at a particular anchor point. Since it is expected to collect a sufficient amount of data during a data gathering tour, we require that the aggregated data uploaded from a sensor to

the mobile collector at all anchor points should be no less than a specified amount. Another parameter is the sojourn time of the mobile collector at each anchor point. We require that the total sojourn time at all anchor points should be constrained within a limit so that the latency of a data gathering tour is bounded.

Since the cost minimization problem essentially answers the questions that where and how sensors communicate with the SenCar, we can characterize it as a pricing mechanism, where sensors independently adjust their payments competing for the data uploading opportunity to the SenCar based on the shadow prices of different anchor points set by the SenCar. Using this feature, we decompose the cost minimization problem into two simpler subproblems that describe the behaviors of sensors and the SenCar, respectively. In this way, instead of directly resolving the original global problem, alternatively we can jointly solve these two subproblems. By iteratively adjusting the payment and the shadow price between each sensor and the mobile collector, an equilibrium [62][104] that reconciliates the two subproblems can be reached, where the overall network cost is minimized.

The contributions of this work can be summarized as follows: (1) We characterize data gathering performance by network cost and formulate the problem of optimizing data gathering performance as a convex problem. (2) We show that this problem can be described as a pricing mechanism so that it can be correspondingly decomposed into two subproblems. (3) We provide a pricing-based algorithm to jointly solve these subproblems in a distributed manner. (4) We present a theoretical analysis and extensive simulation results to validate the convergence of the proposed algorithm and demonstrate that our algorithm can achieve lower network cost than the compared data gathering strategy.

5.2 Related Work

There has been some work in the literature on optimizing data gathering performance in WSNs. Most of the work studied static data gathering and focused on optimal routing for maximum lifetime. For example, Madan and Lall [50] proposed distributed algorithms using a dual decomposition approach to computing an optimal routing that maximizes the time the first node in the network depletes its energy. In [105], Madan, et al. modeled the circuit energy consumption and the traditional physical, MAC and routing layers. They considered the optimization

of individual layers as well as cross-layer optimization by computing a strategy that maximizes network lifetime. In [53], Hua and Yum jointly considered optimal data aggregation based on the correlation of sensors and maximum lifetime routing, which aims to reduce the traffic across the network and balance traffic to avoid overwhelming bottleneck nodes.

There has also been limited work in the literature on the optimization of mobile data gathering performance, where mobile sinks or mobile collectors are employed. Gatzianas and Georgiadis [59] studied network lifetime maximization problem by formulating it into a linear problem. It assumed static data rates of sensors and focused on finding optimal routing from sensors to the mobile sink at different anchor points. Yun and Xia [46] proposed a framework for improving network lifetime by developing several models under delay bound constraints, node energy constraints, and flow conservation constraints. In contrast, our work is significantly different from these work in the sense that we consider data control on each sensor and sojourn time allocation for the mobile collector, instead of the routing problem, to minimize network cost. Moreover, in our model, we impose constraints on data amount gathered from each sensor and total sojourn time at all anchor points in a data gathering tour, which aims to obtain sufficient sensing data within a bounded data gathering latency. These considerations address important practical issues in mobile data gathering applications, which have not been considered in the existing work.

In the meanwhile, pricing mechanisms [60]-[63] have received much attention in recent years. The mechanisms were especially proposed for resource allocation problems, where a resource provider establishes resource prices to charge users, in order to regulate the behavior of selfish users and achieve social welfare maximization [104]. In particular, Kelly et al. [60][61] proposed a scheme in a wired network for elastic traffic that a network provider charges the users based on the traffic load on individual links and the users choose their transmission rates as a function of prices. Qiu and Marbach [63] extended Kelly's work to the bandwidth allocation problem in ad hoc networks, where users can charge other users a price for relaying their data packets. In a more recent work, Hou and Kumar [62] studied the utility maximization problem with delay-based Quality-of-Service (QoS) requirements in a wireless local area network. They characterized the problem as a bidding game, where clients bid for service time from the access point, and the

access point assigns delivery ratios to the clients according to their bids. In contrast, our work demands inelastic data traffic uploaded from sensors to the mobile collector at different anchor points at the lowest possible cost. We attempt to use pricing as a means to regulate the communication between sensors and the mobile collector, where the mobile collector sets the shadow prices for anchor points and each sensor learns link prices from itself to the neighboring anchor points based on these shadow prices, then determines its payment for the data uploading opportunity to the mobile collector at different anchor points.

5.3 System Model and Problem Formulation

Consider a sensor network which consists of a set of static sensors, denoted by $\mathcal{N}$, and a set of anchor points, denoted by $\mathcal{A}$. We study the anchor-based range traversing data gathering scheme, where the mobile collector, i.e., SenCar, gathers data directly from sensors by visiting each anchor point in a periodic data gathering tour. There are several ways to decide the locations of anchor points. One way is to consider the sensing field as a grid and anchor points can be uniformly distributed on grid intersections [110]. An alternative way is to use the positions of a subset of sensors as the locations of anchor points [36][28]. In this work, we would follow the latter option, which not only simplifies the setting of anchor points but also can facilitate the distributed implementation of our algorithm that will be presented in subsequent sections. An example of this data gathering scheme is illustrated in Fig. 5.1, where the locations of seven sensors are chosen as anchor points and the SenCar starts its tour from the static data sink and sequentially visits each anchor point for data gathering.

Since the SenCar moves over different anchor points, we now define two sets that depict the relationship between the SenCar and sensors in the movement. One set is $\mathcal{N}^a$ ($a \in \mathcal{A}$), which represents the sensors in the coverage area of anchor point a. These sensors can directly upload data to the SenCar when it arrives at anchor point a. Another set is $\mathcal{A}_i$ ($i \in \mathcal{N}$), which contains the anchor points sensor i can reach in a single hop. To ensure that each sensor has the opportunity to upload data to the SenCar, we assume that $\mathcal{A}_i$ is always non-empty. This can be guaranteed by choosing the anchor points through finding a set of neighbor sets of sensors such that the selected sets contain all the sensors in the neighbor sets.

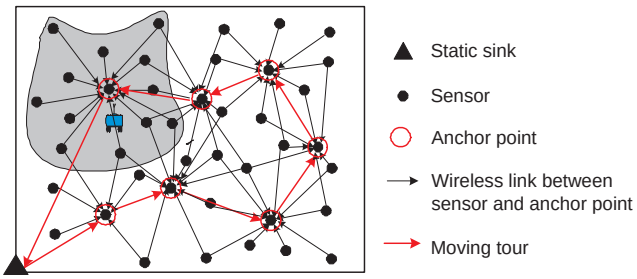


Figure 5.1: An example of anchor-based range traversing data gathering scheme in a WSN, where the positions of a subset of sensors are used as anchor points.

We assume that each sensor has enough buffered sensing data and sensor i would upload an amount of data x_i^a to the SenCar when it stops at anchor point a. In order to ensure that the SenCar can obtain a sufficient amount of data from each sensor in a data gathering tour, we impose a minimum data amount for each sensor, M_i, which indicates the minimum aggregated data uploaded from sensor i to the SenCar at all anchor points in a data gathering tour.

The SenCar would stay at anchor point a for a period of sojourn time t^a to gather data from nearby sensors. In some time-sensitive applications, the data gathering task is expected to be completed in a bounded period, which is equivalent to constraining the total sojourn time at all anchor points within a limit. We denote such a limit by T and call it the bound of total sojourn time. Moreover, considering the prevalence of unreliable channels in WSNs [15], we assume that the data transmission between sensor i and the SenCar at anchor point a experiences a lossy link with a successful delivery ratio of p_i^a. Thus, in order to ensure the SenCar receives x_i^a amount of data, sensor i needs to send out $\frac{x_i^a}{p_i^a}$ amount of packets.

In order to characterize the impact of data uploading from a sensor to the SenCar at a particular anchor point on the overall data gathering performance, we introduce a *cost function*, $C_i^a(\cdot)$, as a strictly convex, increasing and twice-differentiable function with respect to the amount of data uploaded from sensor i to the SenCar at anchor point a (i.e., x_i^a). In practice, the "cost" can be evaluate in terms of energy consumption, monetary cost or other metrics modeling user application needs. Cost function $C_i^a(\cdot)$ implicitly quantifies the suitability for sensor i to upload data towards the SenCar at anchor point a. Correspondingly, the *network cost* is defined as the sum of data gathering costs of all sensors at all anchor points. Our work in

Table 5.1: List of notations used in problem formulation.

Notation	Definition
$\mathcal{N}$	Set of sensors
$\mathcal{A}$	Set of anchor points
$\mathcal{N}^a$	Set of sensors in the coverage of anchor point a, $\mathcal{N}^a \subseteq \mathcal{N}$
$\mathcal{A}_i$	Set of neighboring anchor points of sensor i, $\mathcal{A}_i \subseteq \mathcal{A}$
t^a	Sojourn time of SenCar at anchor pint a
T	Bound of total sojourn time at all anchor points in a data gathering tour
x_i^a	Data amount sensor i uploads to the SenCar at anchor point a in a data gathering tour
p_i^a	Stochastic successful delivery ratio of a link from sensor i to the SenCar at anchor point a
B	Channel bandwidth of the system
M_i	Minimum amount of data sensor i needs to upload to the SenCar in a data gathering tour

this chapter is to minimize the network cost by means of properly scheduling communications between sensors and the SenCar and dynamically adjusting the sojourn time at different anchor points.

The network cost minimization problem can be formalized as follows:

Definition 2. *NCM: Network Cost Minimization Problem for Mobile Data Gathering in WSNs.* Given a set of sensors, $\mathcal{N}$, a set of anchor points, $\mathcal{A}$, the minimum data amount of sensor i ($i \in \mathcal{N}$), M_i, and the bound of total sojourn time at all anchor points, T, find: (1) the data amount x_i^a uploaded from sensor i to the SenCar at anchor point a; (2) the sojourn time t^a of the SenCar at anchor point a, such that network cost is minimized.

Using the notations listed in Table 5.1, the NCM problem can be formulated into the following convex optimization problem.

NCM:

$$\text{Minimize} \quad \sum_{a\in\mathcal{A}}\sum_{i\in\mathcal{N}^a} C_i^a(x_i^a) \tag{5.1}$$

$$\text{Subject to} \quad \sum_{a\in\mathcal{A}_i} x_i^a \geq M_i, \quad \forall i \in \mathcal{N} \tag{5.2}$$

$$\sum_{i\in\mathcal{N}^a} \frac{x_i^a}{p_i^a} \leq B \cdot t^a, \quad \forall a \in \mathcal{A} \tag{5.3}$$

$$\sum_{a\in\mathcal{A}} t^a \leq T, \tag{5.4}$$

$$\text{Over} \quad x_i^a, t^a \geq 0, \quad \forall i \in \mathcal{N}^a, \forall a \in \mathcal{A} \tag{5.5}$$

The constraints in the NCM problem can be explained as follows.

- Data constraint (5.2) shows that for each sensor, its aggregated uploaded data at all anchor points should be no less than the specified minimum amount.
- Link capacity constraint (5.3) enforces that when the SenCar is located at anchor point a, the total transmitted data amount from the sensors in the neighborhood is restricted by the product of channel bandwidth B and sojourn time t^a.
- Total sojourn time constraint (5.4) ensures that the total sojourn time of the SenCar at all anchor points is bounded by T.

5.4 Problem Decomposition and Pricing-based Algorithm

In the previous section, we provided the formulation of the NCM problem. Since the problem has a strictly convex function with respect to x_i^a ($a \in \mathcal{A}, i \in \mathcal{N}^a$) and is over a convex feasible region, the NCM problem is mathematically tractable. However, there exist some difficulties to directly solve it: (1) Cost functions, $C_i^a(\cdot)$, for all $a \in \mathcal{A}, i \in \mathcal{N}^a$, are typically the knowledge of sensors and are unlikely to be known by the network provider or a central controller; (2) Due to the asymmetry of wireless channels, the successful delivery ratio p_i^a that indicates the uplink channel quality from a sensor to the SenCar at an anchor point may not be easily obtained by sensors, which are the senders in the transmissions. This information

can be available at the SenCar by performing a receiving estimation; (3) The adjustable variables x_i^a and t^a in the formulation in fact reveal the behaviors of different entities, where x_i^a reflects the schedule on data uploading at each sensor, and in contrast, t^a characterizes the movement of the SenCar; (4) Finally, it would be difficult to implement a solution in any centralized way in a WSN. To circumvent these difficulties, next we decompose the NCM problem into two simpler subproblems [60][63].

Suppose sensor i chooses to pay q_i^a for the data uploading opportunity when the SenCar stops at anchor point a in a data gathering tour, and in return is permitted to upload x_i^a amount of data proportional to q_i^a, i.e., $q_i^a = \lambda_i^a x_i^a$, where λ_i^a can be considered as the price for uploading a unit amount of data over the link from sensor i to the SenCar at anchor point a. In the following, we simply call λ_i^a *link price*. Then, the local cost minimization problem for sensor i can be expressed as follows.

SENSOR_i:

$$\begin{aligned} \text{Minimize} \quad & \sum_{a\in\mathcal{A}_i} C_i^a\left(\frac{q_i^a}{\lambda_i^a}\right) + \sum_{a\in\mathcal{A}_i} q_i^a \\ \text{Subject to} \quad & \sum_{a\in\mathcal{A}_i} \frac{q_i^a}{\lambda_i^a} \geq M_i, \\ \text{Over} \quad & q_i^a \geq 0, \qquad \forall a \in \mathcal{A}_i \end{aligned} \tag{5.6}$$

In the above we consider two parts of costs for sensor i: $\sum_{a\in\mathcal{A}_i} C_i^a\left(\frac{q_i^a}{\lambda_i^a}\right)$ represents the sum of data uploading cost to all the neighboring anchor points of sensor i, and $\sum_{a\in\mathcal{A}_i} q_i^a$ is the total payment used in competing for the data uploading opportunity. In SENSOR_i problem, given link prices λ_i^a's, sensor i independently minimizes its overall cost under the constraint that its aggregated uploaded data is no less than M_i. Note that to solve this problem, there is no need for sensor i to have the knowledge of link condition p_i^a for all $a \in \mathcal{A}_i$.

On the other hand, given the payments from all sensors, the SenCar tries to maximize function $\sum_{a\in\mathcal{A}}\sum_{i\in\mathcal{N}^a} q_i^a \log(x_i^a)$ under the constraints of channel capacity and total sojourn time bound. In other words, the SenCar needs to solve the following optimization problem.

SENCAR:

$$\begin{aligned} \text{Maximize} \quad & \sum_{a\in\mathcal{A}}\sum_{i\in\mathcal{N}^a} q_i^a \log(x_i^a) \\ \text{Subject to} \quad & \sum_{i\in\mathcal{N}^a} \frac{x_i^a}{p_i^a} \leq B\cdot t^a, \forall a\in\mathcal{A} \\ & \sum_{a\in\mathcal{A}} t^a \leq T, \\ \text{Over} \quad & x_i^a, t^a \geq 0, \qquad \forall i\in\mathcal{N}^a, \forall a\in\mathcal{A} \end{aligned} \tag{5.7}$$

Clearly, the above maximization problem does not require the SenCar to know the cost functions $C_i^a(\cdot)$ for all $a\in\mathcal{A}$ and $i\in\mathcal{N}^a$.

The following theorem shows that by solving SENSOR_i and SENCAR problems, optimal data control and sojourn time allocation can be achieved as the global cost minimization (i.e. NCM) problem.

Theorem 4. *There exist non-negative matrices $x=\{x_i^a|a\in\mathcal{A}, i\in\mathcal{N}^a\}$, $q=\{q_i^a|a\in\mathcal{A}, i\in\mathcal{N}^a\}$ and $\lambda=\{\lambda_i^a|a\in\mathcal{A}, i\in\mathcal{N}^a\}$, and non-negative vector $t=\{t^a|a\in\mathcal{A}\}$ with $q_i^a=\lambda_i^a x_i^a, \forall i\in\mathcal{N}, a\in\mathcal{A}_i$ such that*
(a) For $i\in\mathcal{N}$, with $\lambda_i^a>0$ for all $a\in\mathcal{A}_i$, $q_i=\{q_i^a|a\in\mathcal{A}_i\}$ is the solution to the SENSOR_i problem;
(b) Given that each sensor is charged q_i^a for uploading data to the SenCar when it is located at anchor point a, (x,t) is the solution to the SENCAR problem;
In addition, given such $x,\lambda,t\succ 0$, matrix x and vector t solve the NCM problem.

Proof. We first show the existence of x, q and λ that satisfy (a) and (b), and then prove that the corresponding (x,t) is the solution to the NCM problem.

We assume that with proper settings of parameters M_i and T, there always exist feasible variable matrices x and q, and variable vector t that satisfy the constraints in NCM, SENSOR_i and SENCAR problems with strict inequality, which means that they are interior points in the feasible region of the respective problem. Thus, the Slater's condition for constraint qualification is satisfied [98][50]. Since SENSOR_i, SENCAR and NCM problems are all convex problems, the solution to each problem that satisfies the corresponding Karush-Kuhn-Tucker (KKT) conditions is sufficient to be optimal for the respective problem [98].

For the global cost minimization problem (i.e., NCM), we introduce non-negative Lagrangian multipliers σ^a, μ_i and γ for the constraints in (5.2)-(5.4), respectively.

Then, the Lagrangian of NCM can be obtained as

$$\begin{aligned} & L_{\text{sys}}(x,t,\sigma,\mu,\gamma) \\ = & \sum_{a\in\mathcal{A}}\sum_{i\in\mathcal{N}^a} C_i^a(x_i^a) + \sum_{a\in\mathcal{A}}\sigma^a(\sum_{i\in\mathcal{N}^a}\frac{x_i^a}{p_i^a} - Bt^a) \\ & -\sum_{i\in\mathcal{N}}\mu_i(\sum_{a\in\mathcal{A}} x_i^a - M_i) + \gamma(\sum_{a\in\mathcal{A}} t^a - T). \end{aligned}$$

Assuming $x^* = \{x_i^{a*} | a \in \mathcal{A}, i \in \mathcal{N}^a\}$ and $t^* = \{t^{a*} | a \in \mathcal{A}\}$ are the optimal solution to the NCM problem, we obtain the following KKT conditions.

$$\frac{\partial L_{\text{sys}}}{\partial x_i^a} = C_i^{a'}(x_i^{a*}) + \frac{\sigma^{a*}}{p_i^a} - \mu_i^* = 0, \quad \forall a \in \mathcal{A}, i \in \mathcal{N}^a, \tag{5.8}$$

$$\frac{\partial L_{\text{sys}}}{\partial t^a} = -\sigma^{a*}B + \gamma^* = 0, \forall a \in \mathcal{A}, \tag{5.9}$$

$$\sigma^{a*}(\sum_{i\in\mathcal{N}^a}\frac{x_i^{a*}}{p_i^a} - Bt^{a*}) = 0, \forall a \in \mathcal{A}, \tag{5.10}$$

$$\mu_i^*(\sum_{a\in\mathcal{A}} x_i^{a*} - M_i) = 0, \forall i \in \mathcal{N}, \tag{5.11}$$

$$\gamma^*(\sum_{a\in\mathcal{A}} t^{a*} - T) = 0, \tag{5.12}$$

$$x_i^{a*} \geq 0, t^{a*} \geq 0, \forall a \in \mathcal{A}, i \in \mathcal{N}^a, \tag{5.13}$$

$$\sigma^{a*}, {\mu_i}^*, \gamma^* \geq 0, \forall a \in \mathcal{A}, i \in \mathcal{N}. \tag{5.14}$$

Introducing ν_i as the Lagrangian multipliers for the data constraint of SENSOR$_i$, its Lagrangian is given by

$$\begin{aligned} & L_{\text{sen}\_i}(q,\nu,\varepsilon) \\ = & \sum_{a\in\mathcal{A}_i} C_i^a\left(\frac{q_i^a}{\lambda_i^a}\right) + \sum_{a\in\mathcal{A}_i} q_i^a - \nu_i(\sum_{a\in\mathcal{A}}\frac{q_i^a}{\lambda_i^a} - M_i). \end{aligned}$$

By the KKT conditions, $q_i^* = \{q_i^{a*} | a \in \mathcal{A}_i\}$ is the optimal solution to SENSOR$_i$

problem if and only if there exists ν_i^* that satisfies

$$\frac{\partial L_{\text{sen}\_i}}{\partial q_i^a} = \frac{1}{\lambda_i^a} \cdot C_i^{a\prime}\left(\frac{q_i^{a*}}{\lambda_i^a}\right) + 1 - \frac{\nu_i^*}{\lambda_i^a} = 0, \forall a \in \mathcal{A}_i, \tag{5.15}$$

$$\nu_i^*(\sum_{a\in\mathcal{A}} \frac{q_i^{a*}}{\lambda_i^a} - M_i) = 0, \tag{5.16}$$

$$q_i^{a*} \geq 0, \forall a \in \mathcal{A}_i, \tag{5.17}$$

$$\nu_i^* \geq 0. \tag{5.18}$$

Similarly, introducing multipliers α^a and β for the constraints of SENCAR problem, the Lagrangian of SENCAR can be expressed as follows.

$$\begin{aligned} & L_{\text{car}}(x, t, \alpha, \beta, \rho, \eta) \\ = & -\sum_{a\in\mathcal{A}}\sum_{i\in\mathcal{N}^a} q_i^a \log x_i^a + \sum_{a\in\mathcal{A}} \alpha^a (\sum_{i\in\mathcal{N}^a} \frac{x_i^a}{p_i^a} - Bt^a) \\ & +\beta(\sum_{a\in\mathcal{A}} t^a - T) \end{aligned}$$

For a given q, by the KKT conditions, we have that matrix x^* and vector t^* are the optimal solutions to SENCAR problem if and only if there exist $\alpha^* = \{\alpha^{a*} | a \in \mathcal{A}\}$ and β^* such that

$$\frac{\partial L_{\text{car}}}{\partial x_i^a} = -\frac{q_i^a}{x_i^{a*}} + \frac{\alpha^{a*}}{p_i^a} = 0, \forall a \in \mathcal{A}, i \in \mathcal{N}^a, \tag{5.19}$$

$$\frac{\partial L_{\text{car}}}{\partial t^a} = -\alpha^{a*} B + \beta^* = 0, \forall a \in \mathcal{A}, \tag{5.20}$$

$$\alpha^{a*}(\sum_{i\in\mathcal{N}^a} \frac{x_i^{a*}}{p_i^a} - Bt^{a*}) = 0, \forall a \in \mathcal{A}, \tag{5.21}$$

$$\beta^*(\sum_{a\in\mathcal{A}} t^{a*} - T) = 0, \tag{5.22}$$

$$x_i^{a*} \geq 0, t^{a*} \geq 0, \forall a \in \mathcal{A}, i \in \mathcal{N}^a, \tag{5.23}$$

$$\alpha^{a*} \geq 0, \beta^* \geq 0, \forall a \in \mathcal{A}. \tag{5.24}$$

Let (x^*, t^*) be the optimal solution to the NCM problem and σ^*, μ^* and γ^* be the corresponding multipliers that satisfy the KKT conditions in (5.8)-(5.14). Let $x_i^a = x_i^{a*}$, $t^a = t^{a*}$, $\lambda_i^a = \frac{\sigma^{a*}}{p_i^a}$ and $q_i^a = \frac{\sigma^{a*}}{p_i^a} x_i^{a*}$. It is clear that x_i^a, t^a, λ_i^a and q_i^a are all non-negative. By defining $\alpha^a = \sigma^{a*}$ and $\beta = \gamma^*$, we find that x, t, α and β satisfy

the KKT conditions for SENCAR problem in (5.19)-(5.24). Thus, (x,t) solves SENCAR, which implies that the solution satisfying KKT conditions of NCM also identifies a solution to SENCAR. Defining $\nu_i = {\mu_i}^*$ together with $\lambda_i^a = \frac{\sigma^{a*}}{p_i^a}$, the KKT conditions of SENSOR_i problem are satisfied such that $q_i^a = \frac{\sigma^{a*}}{p_i^a} x_i^{a*}$ is the solution to SENSOR_i. This analysis establishes the existence of x, λ and t.

On the other hand, suppose we are given x, t and λ satisfying conditions (a) and (b) of the theorem. We show that (x,t) is the solution to NCM. It is clear that by (5.19) and the definition of λ_i^a, we have $\lambda_i^a = \frac{\alpha^a}{p_i^a}$. Letting $\sigma^a = \alpha^a$ and $\mu_i = \nu_i$, we have that $\lambda_i^a = \frac{\sigma^a}{p_i^a}$ and condition (5.8) of NCM holds by (5.15). Furthermore, letting $\gamma = \beta$, conditions in (5.9)-(5.14) are equivalent to (5.16)-(5.18) and (5.20)-(5.24). Thus, x, t, σ, μ and γ satisfy the KKT conditions in (5.8)-(5.14). Therefore, we conclude that (x,t) solves NCM. □

Theorem 4 implies that instead of directly solving the NCM problem, alternatively we can jointly solve the SENSOR_i and SENCAR subproblems, which have less complexity and facilitate a distributed implementation for the solution. System optimum can be achieved when sensors' payment q and SenCar's data control x and link price λ reach equilibrium, i.e., $q_i^a = \lambda_i^a x_i^a$ for all $a \in \mathcal{A}, i \in \mathcal{N}^a$.

The SENCAR problem requires the payment information from all the sensors. It may incur high communication overhead if the SENCAR problem is solved in a centralized way [61]. Thus, we consider its dual problem to decompose it into a set of subproblems with respect to each anchor point [101][103]. By taking advantage of the fact that there is a sensor at each anchor point, the subproblems can be solved with the aid of these sensors. For clarity, we call them *help nodes* in the following. This way, to announce payment q_i^a, sensor i only needs to locally inform the help node at anchor point a.

We form the dual problem of SENCAR by introducing Lagrangian multipliers α^a's ($a \in \mathcal{A}$) for channel capacity constraints. This results in the partial Lagrangian as

$$\begin{aligned} L'_{car}(x,t,\alpha) &= -\sum_{a\in\mathcal{A}}\sum_{i\in\mathcal{N}^a} q_i^a \log x_i^a + \sum_{a\in\mathcal{A}} \alpha^a \Big(\sum_{i\in\mathcal{N}^a} \frac{x_i^a}{p_i^a} - Bt^a\Big) \\ &= \sum_{a\in\mathcal{A}}\sum_{i\in\mathcal{N}^a} \Big(-q_i^a \log x_i^a + \alpha^a \frac{x_i^a}{p_i^a}\Big) - \sum_{a\in\mathcal{A}} \alpha^a B t^a, \end{aligned}$$

where α^a is also referred to as *shadow price* of anchor point a.

Given price λ, the minimum of L'_{car} occurs when $x_i^a = q_i^a/\lambda_i^a$. Thus, the dual

function is defined as

$$g(\alpha) = \inf_{t} \left\{ L'_{\text{car}}\left(\tfrac{q}{\lambda}, t, \alpha\right) \middle| \sum_{a \in \mathcal{A}} t^a \leq T \right\}.$$

Correspondingly, the dual problem is to find a shadow price vector α^* that maximizes dual function $g(\alpha)$.

Moreover, based on Lagrangian L'_{car}, we have

$$\frac{\partial L'_{\text{car}}}{\partial x_i^a} = -\frac{q_i^a}{x_i^a} + \frac{\alpha^a}{p_i^a} = -\lambda_i^a + \frac{\alpha^a}{p_i^a}.$$

For the optimum solution to the SENCAR problem, we have $\frac{\partial L'_{\text{car}}}{\partial x_i^a} = 0$, i.e., $\lambda_i^a = \frac{\alpha^a}{p_i^a}$. This can be interpreted as that link price λ_i^a from sensor i to anchor point a is actually determined by the shadow price of this anchor point and the quality of the link between them.

From the above analysis, we can see that data matrix x^* is the optimal solution to the NCM problem if and only if there exists shadow price vector α^* solving the dual problem of SENCAR such that for each $a \in \mathcal{A}$, $i \in \mathcal{N}^a$, we have that $x_i^{a*} = \frac{q_i^a}{\lambda_i^a}$, where $\lambda_i^a = \frac{\alpha^{a*}}{p_i^a}$ and q_i^a is the solution to the SENSOR_i problem for a given λ_i^a. Based on this result, to find the optimal solution, we will gradually vary shadow price α of anchor points, derive link price λ accordingly and give data amount x as a function of link price λ. When shadow price vector α iteratively converges to its optimum α^*, the optimal solution to the NCM problem can be achieved. Note that the shadow price is associated with an anchor point. Therefore, the task of finding optimal vector α^* can be done by the help node at each anchor point in a distributed manner. Next, we propose a pricing-based algorithm to jointly solve the dual problem of SENCAR and SENSOR_i.

Pricing-based Algorithm:

For all $a \in \mathcal{A}$, the help node independently initializes the shadow price α^a for anchor point a to a positive value.

Repeat the following iteration until the shadow price vector α converges to α^*.

At iteration n,

- For all $a \in \mathcal{A}$, the help node at anchor point a determines link price $\lambda_i^a(n)$ for all $i \in \mathcal{N}^a$ by setting

$$\lambda_i^a(n) = \frac{\alpha^a(n)}{p_i^a},$$

and then sends this information to sensors in its neighborhood.

- For all $i \in \mathcal{N}$, after learning link price $\lambda_i^a(n)$ for all $a \in \mathcal{A}_i$, sensor i decides its payments $q_i^a(n)$'s for its neighboring anchor points by solving SENSOR_i problem to minimize the local cost, i.e.,

$$q_i^a(n) = \arg\min_{q_i^a \geq 0} \left\{ \sum_{a \in \mathcal{A}_i} C_i^a \left(\frac{q_i^a}{\lambda_i^a(n)} \right) + \sum_{a \in \mathcal{A}_i} q_i^a \right. \\ \left. \left| \sum_{a \in \mathcal{A}_i} \frac{q_i^a}{\lambda_i^a(n)} \geq M_i \right. \right\}, \lambda_i^a(n) > 0,$$

  and then announces these payments to the corresponding help nodes at neighboring anchor points.

- Help nodes exchange the information of shadow prices. In the cases that the help nodes are not connected, we assume that they can use slightly higher transmission power to ensure a minimum degree of connectivity among themselves. Once obtaining the payment information from nearby sensors, the help node at each anchor point a would derive the possible data amount that each nearby sensor can upload to the SenCar when arriving at this anchor point based on the respective link cost, i.e., for each $a \in \mathcal{A}$ and $i \in \mathcal{N}^a$, the help nodes set

$$x_i^a(n) = q_i^a(n)/\lambda_i^a(n). \tag{5.25}$$

  In order to minimize L'_{car}, each help node sets the sojourn time for its located anchor point by following rule

$$t^a(n) = \begin{cases} T, & \text{If } a = \arg\max_{a \in \mathcal{A}} \alpha^a(n) \\ 0, & \text{Otherwise.} \end{cases} \tag{5.26}$$

- Upon receiving the payment information from all sensors in its neighborhood and having identified the sojourn time, each help node updates the shadow price for its located anchor point according to

$$\alpha^a(n+1) = \left[\alpha^a(n) + \theta(n) \left(\sum_{i \in \mathcal{N}^a} \frac{x_i^a(n)}{p_i^a} - B t^a(n) \right) \right]^+, \tag{5.27}$$

where $[\cdot]^+$ denotes the projection onto the positive orthant and $\theta(n)$ is a properly chosen scalar stepsize for iteration n. In our algorithm, we choose the diminishing stepsize, i.e., $\theta(n) = d/(b+cn)$, $\forall n, c, d > 0, b \geq 0$, where b, c and d are adjustable parameters that regulate the convergence speed. The diminishing stepsize can guarantee the convergence regardless of the initial value of α^a [98].

- Note that SENCAR problem is not strictly concave with respect to sojourn time t^a, which implies that the values of t^a in the optimal solution to the Lagrangian dual cannot be directly applied to the primal SENCAR problem. In view of this, we recover the solutions by applying the method introduced in [100]. For iteration n, we compose a primal feasible $\hat{t}^a(n)$ as follows.

$$\begin{aligned} \hat{t}^a(n) &= \tfrac{1}{n}\sum_{h=1}^{n} t^a(h) \\ &= \begin{cases} t^a(1) & n = 1 \\ \frac{n-1}{n}\hat{t}^a(n-1) + \frac{1}{n}t^a(n) & n > 1 \end{cases} \end{aligned} \tag{5.28}$$

  It was proved in [100] that when the diminishing stepsize is used, any accumulation point of sequence $\{\hat{t}^a(n)\}$ generated by (5.28) is feasible to the primal problem and $\{\hat{t}^a(n)\}$ can converge to a primal optimal solution.

Finally, the converged value of matrix $x(n) = \{x_i^a(n) | i \in \mathcal{N}, a \in \mathcal{A}\}$ and vector $\hat{t}(n) = \{\hat{t}^a(n) | a \in \mathcal{A}\}$ indicates the optimal data control for sensors and the optimal sojourn time allocation for the SenCar, i.e., $(x, \hat{t})$ is the optimal solution to the NCM problem.

5.5 Local Cost Minimization at Sensors

In this section, we consider the second step of the pricing-based algorithm: how to solve SENSOR_i problem by each sensor under given link price vector $\lambda_i = \{\lambda_i^a | a \in \mathcal{A}_i\}$. As aforementioned, $C_i^a(\cdot)$ is a monotonic increasing function. Thus, the minimum of the objective function in (5.6) should be achieved when $\sum_{a \in \mathcal{A}_i} \frac{q_i^a}{\lambda_i^a} = M_i$. Considering this fact, the SENSOR_i problem can be rewritten as follows.

SENSOR_i:

$$\begin{aligned} \text{Minimize} \quad & \sum_{a\in\mathcal{A}_i} C_i^a\left(\frac{q_i^a}{\lambda_i^a}\right) + \sum_{a\in\mathcal{A}_i} q_i^a \\ \text{Subject to} \quad & \sum_{a\in\mathcal{A}_i} \frac{q_i^a}{\lambda_i^a} = M_i, \\ \text{Over} \quad & q_i^a \geq 0, \qquad \forall a \in \mathcal{A}_i \end{aligned} \tag{5.29}$$

Let f_i denote the objective function of SENSOR_i. Since $f_i = \sum_{a\in\mathcal{A}_i} C_i^a(\frac{q_i^a}{\lambda_i^a}) + \sum_{a\in\mathcal{A}_i} q_i^a = \sum_{a\in\mathcal{A}_i} C_i^a(\frac{q_i^a}{\lambda_i^a}) + \sum_{a\in\mathcal{A}_i} \lambda_i^a(\frac{q_i^a}{\lambda_i^a})$, f_i is a function with respect to variable vector $x_i = \{x_i^a = \frac{q_i^a}{\lambda_i^a} | a \in \mathcal{A}_i\}$, where x_i can be considered as the demand of uploading data vector at sensor i.

For each sensor i, let $\hat{a}_i$ be the index of the minimum-marginal-cost anchor point for sensor i. That is,

$$\hat{a}_i = \arg\min_{a\in\mathcal{A}_i}\left\{\frac{\partial f_i(x_i)}{\partial x_i^a}\right\} = \arg\min_{a\in\mathcal{A}_i}\left\{C_i^{a'}\left(\frac{q_i^a}{\lambda_i^a}\right) + \lambda_i^a\right\}.$$

If there are multiple minimum-marginal-cost anchor points, we can randomly choose one. Since SENSOR_i is a convex problem, we can characterize solution q_i^* by the following optimality condition [97][105].

$$\begin{aligned} & \sum_{a\in\mathcal{A}_i} \frac{\partial f_i(x_i^*)}{\partial x_i^a}\left(x_i^a - x_i^{a*}\right) \\ = & \sum_{a\in\mathcal{A}_i}\left(C_i^{a'}\left(\frac{q_i^{a*}}{\lambda_i^a}\right) + \lambda_i^a\right)\left(\frac{q_i^a - q_i^{a*}}{\lambda_i^a}\right) \geq 0. \end{aligned}$$

This optimality condition can be equivalently expressed as

$$q_i^{a*} > 0, \quad \text{only if } \left[\frac{\partial f_i({x_i}^*)}{\partial x_i^{a'}} \geq \frac{\partial f_i({x_i}^*)}{\partial x_i^a}, \forall a' \in \mathcal{A}_i\right].$$

That is, for each anchor point $a \in \mathcal{A}_i$, sensor i only pays for the data uploading opportunity to the SenCar at those anchor points that incur the minimum marginal cost. This intuitively suggests that sensor i should gradually shift the payment to the minimum-marginal-cost anchor point from other neighboring anchor points and finally reach an equilibrium, where the aggregated marginal cost of anchor points selected for data uploading is less than or equal to that of unselected anchor points [106]. In the following, we present an adaptation algorithm that strikes for such

equilibrium.

Adaptation algorithm:

1. Case I: If $|\mathcal{A}_i| = 1$, then $q_i^a = \lambda_i^a M_i$;

2. Case II: If $|\mathcal{A}_i| > 1$, sensor i first initializes its payment vector $q_i(0) = \{q_i^a(0) \geq 0 | a \in \mathcal{A}_i\}$ that satisfies $\sum_{a\in\mathcal{A}_i} \frac{q_i^a(0)}{\lambda_i^a} = M_i$. For example, we can let $q_i^a(0) = \frac{M_i \lambda_i^a}{|\mathcal{A}_i|}$, where $|\mathcal{A}_i|$ represents the cardinality of set $\mathcal{A}_i$. Then, it iteratively updates vector $q_i(k)$ according to

$$q_i^a(k+1) = \varphi(k)\bar{q}_i^a(k) + [1-\varphi(k)]q_i^a(k), \forall a \in \mathcal{A}_i \tag{5.30}$$

$$\bar{q}_i^a(k) = \begin{cases} \left[q_i^a(k) - \delta(k)\lambda_i^a \left(\frac{\partial f_i(x_i(k))}{\partial x_i^a} - \frac{\partial f_i(x_i(k))}{\partial x_i^{\hat{a}_i}}\right)\right]^+ \\ \qquad \text{if } a \in \mathcal{A}_i, a \neq \hat{a}_i \text{ and } q_i^a(k) \geq 0, \\ \lambda_i^{\hat{a}_i} \cdot \left(M_i - \sum\limits_{a\in\mathcal{A}_i, a\neq\hat{a}_i} \frac{\bar{q}_i^a(k)}{\lambda_i^a}\right), \text{if } a = \hat{a}_i, \end{cases} \tag{5.31}$$

where $[\cdot]^+$ denotes the projection onto the non-negative orthant, k stands for the iteration index, $\delta(k)$ is a small positive scalar stepsize, and $\varphi(k)$ is a scalar on $[a, 1]$ with $0 < a \leq 1$. In other words, the new payment for each neighboring anchor point is a weighted average of the amount in the previous iteration and currently derived optimal value.

The adaptation algorithm can be explained as follows. If anchor point a is not chosen as the minimum-marginal-cost anchor point by sensor i (i.e., $a \neq \hat{a}_i$) and there still exists positive payment for it, this payment should be reduced. On the contrary, if a is chosen as the minimum-marginal-cost anchor point (i.e., $a = \hat{a}_i$), the payment for it should be increased and the increased amount is proportional to the linear combination of the aggregated payment shifted from all other neighboring anchor points of sensor i in order to ensure $\sum_{a\in\mathcal{A}_i} \frac{q_i^a}{\lambda_i^a} = M_i$.

We have the following theorem regarding the convergence of the adaptation algorithm.

Theorem 5. *When stepsize $\delta(k)$ is small enough, the adaptation algorithm converges to a unique optimal solution q_i^* to the SENSOR_i problem.*

Proof. We first show that when $\delta(k)$ is no more than a certain value, adjusting payment vector q_i by the adaptation algorithm in (5.30)-(5.31) always results in the

decrease of local cost at sensor i, i.e., $f_i(x_i(k+1)) \leq f(x_i(k))$. Then we show that such adaptation would finally reach an equilibrium to achieve the unique optimal solution $q_i^* = \{q_i^{a*} | a \in \mathcal{A}_i\}$.

From the adaptation algorithm, it is straightforward to verify

$$\sum_{a \in \mathcal{A}_i} \frac{\bar{q}_i^a(k) - q_i^a}{\lambda_i^a} = 0, \text{ and} \tag{5.32}$$

$$\begin{aligned} \left(\frac{\bar{q}_i^{\hat{a}_i} - q_i^{\hat{a}_i}}{\lambda_i^{\hat{a}_i}} \right)^2 &= \left(\sum_{a \in \mathcal{A}_i, a \neq \hat{a}_i} \frac{q_i^a - \bar{q}_i^a}{\lambda_i^a} \right)^2 \\ &\leq (|\mathcal{A}_i| - 1) \cdot \sum_{a \in \mathcal{A}_i, a \neq \hat{a}_i} \left(\frac{q_i^a - \bar{q}_i^a}{\lambda_i^a} \right)^2. \end{aligned} \tag{5.33}$$

It is clear that $f_i(x_i)$ is defined on the compact set $\chi = \{x_i^a \in x_i | \sum_{a \in \mathcal{A}_i} x_i^a = M_i, x_i^a \geq 0\}$. As $\nabla^2 f_i$ is continuous on χ, we assume that its norm is bounded by some scalar $L > 0$ [107]. Denoting the cost difference between two consecutive iterations as Δ_{x_i} and applying the mean value theorem [97][107], we have

$$\begin{aligned} \Delta_{x_i} &= f_i(x_i(k+1)) - f_i(x_i(k)) \\ &\leq < \nabla f_i(x_i(k), x_i(k+1) - x_i(k) > \\ &\quad + \frac{L}{2} |x_i(k+1) - x_i(k)|^2 \end{aligned} \tag{5.34}$$

Based on (5.34) and $q_i^a(k+1) - q_i^a(k) = \varphi(k)(\bar{q}_i^a(k) - q_i^a(k))$ by (5.30), Δ_{x_i} can

be rewritten as

$$
\begin{aligned}
\Delta_{x_i} &\leq \sum_{a\in\mathcal{A}_i} \frac{\partial f_i(x_i(k))}{\partial x_i^a}\varphi(k)\left(\frac{\bar{q}_i^a(k)-q_i^a(k)}{\lambda_i^a}\right) \\
&\quad + \frac{L}{2}\varphi^2(k)\sum_{a\in\mathcal{A}_i}\left(\frac{\bar{q}_i^a(k)-q_i^a(k)}{\lambda_i^a}\right)^2 \\
&= \sum_{a\in\mathcal{A}_i}\left(\frac{\partial f_i(x_i(k))}{\partial x_i^a} - \frac{\partial f_i(x_i(k))}{\partial x_i^{\hat{a}_i}}\right)\varphi(k)\left(\frac{\bar{q}_i^a(k)-q_i^a(k)}{\lambda_i^a}\right) \\
&\quad + \frac{\partial f_i(x_i(k))}{\partial x_i^{\hat{a}_i}}\varphi(k)\sum_{a\in\mathcal{A}_i}\left(\frac{\bar{q}_i^a(k)-q_i^a(k)}{\lambda_i^a}\right) \\
&\quad + \frac{L}{2}\varphi^2(k)\sum_{a\in\mathcal{A}_i}\left(\frac{\bar{q}_i^a(k)-q_i^a(k)}{\lambda_i^a}\right)^2 \\
&= \sum_{a\in\mathcal{A}_i, a\neq\hat{a}_i}\left(\frac{\partial f_i(x_i(k))}{\partial x_i^a} - \frac{\partial f_i(x_i(k))}{\partial x_i^{\hat{a}_i}}\right)\varphi(k)\left(\frac{\bar{q}_i^a(k)-q_i^a(k)}{\lambda_i^a}\right) \\
&\quad + \frac{L}{2}\varphi^2(k)\sum_{a\in\mathcal{A}_i}\left(\frac{\bar{q}_i^a(k)-q_i^a(k)}{\lambda_i^a}\right)^2 \\
&\leq -\sum_{a\in\mathcal{A}_i, a\neq\hat{a}_i}\frac{a}{\delta(k)}\left(\frac{\bar{q}_i^a(k)-q_i^a(k)}{\lambda_i^a}\right)^2 + \frac{L}{2}\cdot \\
&\quad \left[\left(\frac{\bar{q}_i^{\hat{a}_i}(k)-q_i^{\hat{a}_i}(k)}{\lambda_i^{\hat{a}_i}}\right)^2 + \sum_{a\in\mathcal{A}_i, a\neq\hat{a}_i}\left(\frac{\bar{q}_i^a(k)-q_i^a(k)}{\lambda_i^a}\right)^2\right] \\
&\leq \sum_{a\in\mathcal{A}_i, a\neq\hat{a}_i} -\left(\frac{a}{\delta(k)} - \frac{L}{2}|\mathcal{A}_i|\right)\left(\frac{\bar{q}_i^a-q_i^a}{\lambda_i^a}\right)^2,
\end{aligned}
\tag{5.35}
$$

where the first equality follows from adding and subtracting the same term of $\frac{\partial f_i(x_i(k))}{\partial x_i^{\hat{a}_i}}\varphi(k)\sum_{a\in\mathcal{A}_i}\left(\frac{\bar{q}_i^a(k)-q_i^a(k)}{\lambda_i^a}\right)$, the second equality holds by (5.32), the second inequality follows from the fact that $a \leq \varphi \leq 1$ and the observation by (5.31) that for all $a \neq \hat{a}_i$, $\frac{\partial f_i(x_i(k))}{\partial x_i^a} - \frac{\partial f_i(x_i(k))}{\partial x_i^{\hat{a}_i}} \geq \frac{q_i^a(k)-\bar{q}_i^a(k)}{\delta(k)\lambda_i^a}$, and the third inequality holds by (5.33). Therefore, when $\delta(k) \leq \frac{2a}{L|\mathcal{A}_i|}$, the right side of (5.35) is non-positive so that $f_i(x_i(k+1)) \leq f_i(x_i(k))$ always holds. This implies that the updating on $\{q_i^a(k)\}$ by the adaptation algorithm always reduces the local cost at sensor i.

From the KKT conditions of the SENSOR_i problem listed in (5.15)-(5.18), we can obtain $C_i^{a'}(\frac{q_i^{a*}}{\lambda_i^a}) + \lambda_i^a = \nu_i^*$. Since $C_i^a(\cdot)$ is strictly convex, increasing and twice differentiable, the inverse function of $C_i^{a'}(\cdot)$, i.e., $C_i^{a'-1}(\cdot)$, exists and is continuous. Thus, over the orthant $q_i^{a*} \geq 0$ for all $a \in \mathcal{A}_i$, we have

$$
q_i^{a*} = \begin{cases} 0, & \text{if } \nu_i^* < C_i^{a'}(0) + \lambda_i^a \\ \lambda_i^a \cdot C_i^{a'-1}(\nu_i^* - \lambda_i^a), & \text{if } \nu_i^* \geq C_i^{a'}(0) + \lambda_i^a. \end{cases}
\tag{5.36}
$$

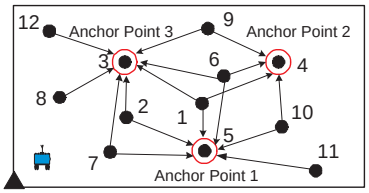


Figure 5.2: An example network with 12 sensors and 3 anchor points.

In the adaptation algorithm, in order to obtain the optimal solution, we always increase the minimum marginal cost, i.e., $C_i^{\hat{a}_i\prime}\left(\frac{q_i^{\hat{a}_i}}{\lambda_i^{\hat{a}_i}}\right)+\lambda_i^{\hat{a}_i}$, by increasing the corresponding payment $q_i^{\hat{a}_i}$ for anchor point $\hat{a}_i$, and decrease other marginal costs $C_i^{a\prime}\left(\frac{q_i^a}{\lambda_i^a}\right)+\lambda_i^a$ for all $a \in \mathcal{A}_i$ and $a \neq \hat{a}_i$ by reducing the payments for them. In this way, we stipulate the marginal costs towards to the same value ν_i^* for all anchor points with positive q_i^{a*}'s ($a \in \mathcal{A}_i$). Therefore, at the equilibrium, the unique optimal solution can be achieved by (5.36). □

5.6 Simulation Results

In this section, we provide simulation results to demonstrate the usage and efficiency of the proposed algorithm and compare its performance with another data gathering strategy.

5.6.1 Convergence

In this subsection, we illustrate the convergence of the pricing-based algorithm via a numerical case study. We consider a WSN with a total of 12 sensors as shown in Fig. 5.2. The locations of sensors 3, 4 and 5 are chosen as anchor points and each of these sensors would act as the helping node in computing for the respective anchor point. In the figure, there is a link between an anchor point and each of its neighboring sensors. We define the cost function as $C_i^a(x_i^a) = \omega_i^a x_i^{a2}$, where ω_i^a is the weight of cost for sensor i to upload data to the SenCar at anchor point a. Clearly, a larger weight ω_i^a would have more impact on the entire network cost. For clarity, we list all the parameter settings in Table 5.2.

Fig. 5.3 shows the evolution of network cost, shadow price α^a, recovered sojourn time variable $\hat{t}^a$, and data variable x_i^a versus the number of iterations in the

Table 5.2: Parameter settings.

Notation	Value	Notation	Value
ω_i^a	ranging from 0.01 to 0.08	T	42 seconds
B	250kbps	$\theta(n)$	$\frac{1}{1+20n}$
p_i^a	ranging from 0.7 to 1	$\delta(k)$	0.03
M_i	800Kb	$\varphi(k)$	0.8

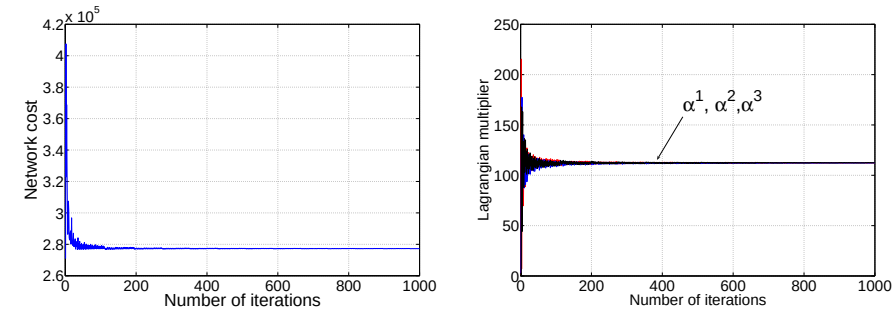
pricing-based algorithm. It can be seen from Fig. 5.3(a) that network cost first drops sharply in the first few iterations and then slightly decreases until it reaches optimum. It falls within 2% of its optimum after only 40 iterations. Fig. 5.3(b) shows that the shadow prices of three anchor points converge very fast and they finally reach almost the same value in the equilibrium. Since all shadow prices are much larger than zero, it indicates that the communication opportunity between sensors and the SenCar at all anchor points is fully utilized. By the adjustment policy on shadow prices in the pricing-based algorithm, when T is large enough to satisfy all the data uploading demands from sensors to each anchor point, the corresponding shadow prices can be reduced to almost zero. Fig. 5.3(c) shows the convergence of recovered sojourn time at different anchor points. It further validates that at any iteration step, the recovered sojourn time is feasible to the primal SENCAR problem, i.e, satisfying the total sojourn time constraint, and when diminishing stepsize is used, the recovery process guarantees its convergence to optimum. In Fig. 5.3(d)-(f), we investigate the evolution of the data amount uploaded from selected sensors 1, 6 and 10 to their neighboring anchor points. We can see that they all approach the stable state after 200 iterations. For a particular sensor, say, sensor 1, as its weight of the cost for anchor point 1 is smaller than those of other two anchor points, more data would destine to the SenCar at anchor point 1 so as to minimize the cost. In Fig. 5.4, we plot two instances to demonstrate the convergence of the adaptation algorithm for solving the SENSOR_i problem with stepsize $\delta(k) = 0.03$. We focus on sensor 1 in two cases where link price vectors are $\lambda_1 = \{1.11, 2, 3\}$ and $\lambda_1 = \{124.6, 112.3, 111.97\}$, respectively. In both cases, we find that the payment for each neighboring anchor point can be determined in about 1000 iterations. It is clear that the smaller the stepsize is, the slower the convergence is, however, the smoother the adaptation towards optimum. In practice, besides using the constant stepsize for the adaptation algorithm as in our simulations, each sensor can dynam-

ically set its stepsize by first choosing a larger value to ensure faster convergence, and subsequently reducing the stepsize once there is an oscillating around some values.

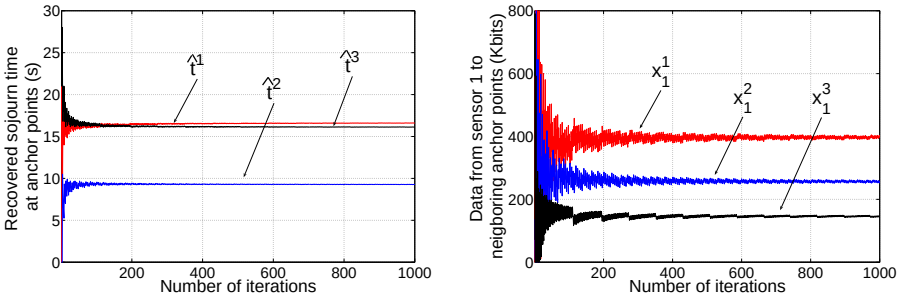
5.6.2 Network Cost

In this subsection, we conduct a suite of simulations to evaluate the network cost achieved by the pricing-based algorithm and compare the results with another data gathering strategy called cluster-based algorithm, where sensors are virtually clustered, i.e., each sensor is randomly associated with a neighboring anchor point and uploads all its data to the SenCar only when it arrives at this anchor point. This algorithm is commonly considered as a simple and effective strategy for the anchor-based range traversing data gathering scheme in the existing literature [28][33]. We consider a generic sensor network with $|\mathcal{N}|$ sensors randomly distributed over the sensing field and $|\mathcal{A}|$ anchor points that can cover all sensors. The cost functions are defined as $C_i^a(x_i^a) = \omega_i^a x_i^{a2}$ for all $a \in \mathcal{A}$, $i \in \mathcal{N}^a$ and the weights of the cost ω_i^a's are generated as discrete uniform random numbers ranging from 0.01 to 0.10. We assume the minimum data amount M_i is equally set for all sensors and its value equals 800Kb if not specified otherwise. The channel bandwidth B is set to 250Kbps. Moreover, we introduce $\bar{p}$ to denote the average successful delivery ratio of all links and use it to characterize the physical condition of the network. The successful delivery ratio p_i^a of each link between a sensor and an anchor point would ranges from $2\bar{p} - 1$ to 1. Considering the randomness of the network topology, each performance point in the figures below is the average of the results in 100 simulation experiments.

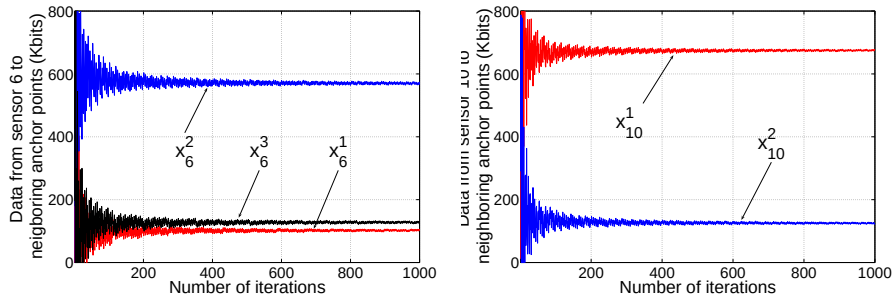
Fig. 5.5 plots the network cost of the pricing-based algorithm when the bound of total sojourn time T is varied from 175 seconds to 225 seconds. The number of sensors $|\mathcal{N}|$ is set to 50 and the number of anchor points $|\mathcal{A}|$ is set to 3. We investigate network cost of four cases, where $\bar{p}$ equals 0.8, 0.85, 0.9, 0.95 and 1, respectively. It can be seen from the figure that in most cases, network cost decreases as T increases. This result is reasonable and can be explained as follows. Since cost function $C_i^a(\cdot)$ is convex, it is expected that each sensor sends parts of its data to the SenCar at different anchor points so as to minimize the aggregated cost. When the restriction on the total sojourn time becomes loose, each sensor can send the



(a) Network cost vs. n. (b) α^a vs. n.

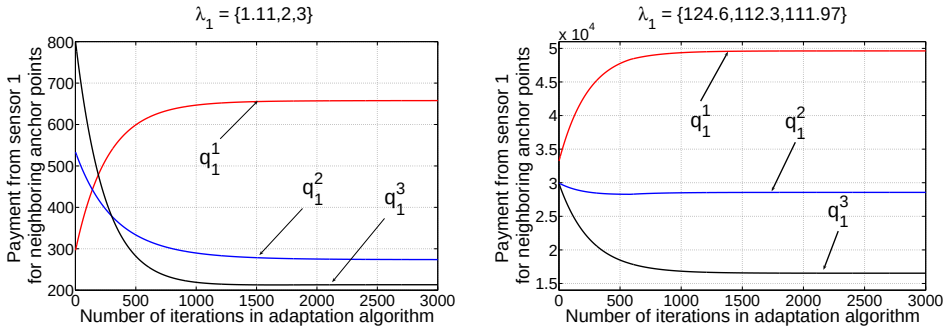


(c) $\hat{t}^a$ vs. n. (d) x_1^a vs. n.



(e) x_6^a vs. n. (f) x_{10}^a vs. n.

Figure 5.3: The evolution of network cost, shadow prices of different anchor points, recovered sojourn time for SenCar stopping at different anchor points, and uploading data from sensors 1, 6 and 10 versus the number of iterations in the pricing-based algorithm.



(a) q_1^a vs. k. (b) q_1^a vs. k.

Figure 5.4: The evolution of the payment from sensor 1 for different anchor points versus the number of iterations in the adaptation algorithm.

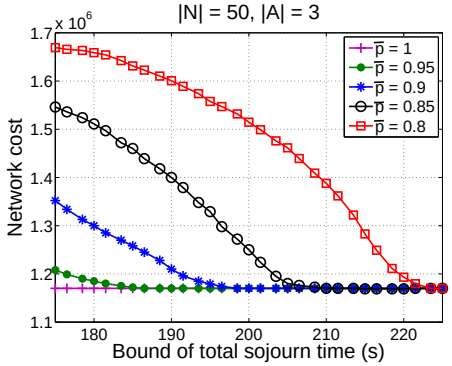


Figure 5.5: Network cost of the pricing-based algorithm as a function of the bound of total sojourn time T.

preferred amount of data to the SenCar more freely at different anchor points, otherwise, in order to ensure the bound of total sojourn time, sensors are restricted to send more data to the SenCar at some particular anchor points in order to complete the data uploading in a shorter time. We also notice that for a given T, the cases with a larger $\bar{p}$ always achieve lower network cost than the cases with a smaller $\bar{p}$. For instance, when $T = 180$s, the case of $\bar{p} = 0.95$ results in around 30% improvement on the network cost with respect to the case of $\bar{p} = 0.8$. When T becomes large enough, such as $T > 220$s, all the cases reach the same minimum network cost, which implies that T no longer affects the network performance and the benefit of data control that smartly schedules the communication between sensors and the SenCar at different anchor points can be fully extracted by all cases.

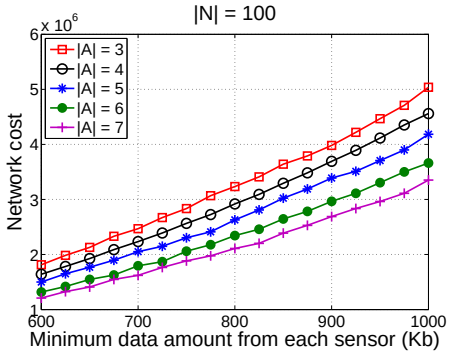


Figure 5.6: Network cost of the pricing-based algorithm as a function of minimum data amount from each sensor M_i.

Fig. 5.6 plots the network cost of the pricing-based algorithm when the minimum amount of each sensor M_i is varied from 600Kb to 800Kb. We consider the network with 100 sensors and all sensors holding the same minimum constraint M_i for gathered data amount. We assume that $\bar{p}$ is equal to 0.8 and the total sojourn time bound T is set as $\sum_i M_i/(\bar{p} \cdot B)$. We consider five cases, where the number of anchor points, i.e., $|\mathcal{A}|$, is varied from 3 to 7. It is shown in the figure that the network cost increases with M_i. This is intuitive as the network cost is the aggregation of the data cost and each data cost function is monotonically increased with the data mount transmitted from a sensor to an anchor point. When the total data mount M_i becomes large, the data of a sensor destined to a particular anchor point may correspondingly increased, though the extend of increase may be different from one anchor point to another. We also notice that for a given M_i, the network cost would decrease with the increase of the number of anchor points. For example, when $M_i = 800$Kb, the network cost of the case with $|\mathcal{A}| = 7$ is 30% lower than that of the case with $|\mathcal{A}| = 3$. This is because that larger $|\mathcal{A}|$ implies that each sensor would have more neighboring anchor points. This provides more opportunities for each sensor to preferentially balance its data to different anchor points, which would result in less network cost.

Fig. 5.7 shows the network cost comparison between the pricing-based algorithm and the cluster-based algorithm when the number of sensors is varied from 10 to 200 under different settings of $|\mathcal{A}|$ and M_i. We use PA and CA to denote the pricing-based algorithm and the cluster-based algorithm in the figure, respectively. The total sojourn time bound T is set to $4.5|\mathcal{N}|$ seconds, which is sufficient

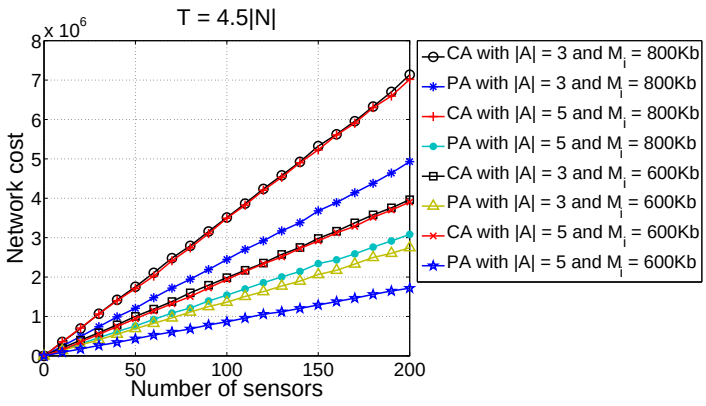


Figure 5.7: Network cost comparison between the pricing-based algorithm and the cluster-based algorithm.

to accommodate the data uploading from all sensors. From the figure, we can draw some observations. First, network cost increases in all cases of both algorithms as the number of sensors increases. This is intuitive. As each sensor needs to upload certain amount of data to the SenCar, more cost is incurred by the increase of sensors. Second, given $|\mathcal{A}|$ and M_i, the pricing-based algorithm always achieves lower network cost. For example, when $|\mathcal{N}| = 100$, $|\mathcal{A}| = 3$, and $M_i = 800$Kb, the pricing-based algorithm results in 32% less network cost with respect to the cluster-based algorithm. The underlying reason for such superiority of the pricing-based algorithm is that each sensor can adaptively split its data and send the data to the SenCar at different neighboring anchor points such that the aggregated cost is minimized. Third, the increase of $|\mathcal{A}|$ can effectively reduce the network cost for pricing-based algorithm, however, it has little impact on the performance of cluster-based algorithm. This is because that no matter how many anchor points there are, in the cluster-based algorithm, each sensor is associated with only one anchor point. That means that there is no much chance to greatly alleviate the data cost at each sensor since each sensor can not send its data dispersedly to the SenCar at multiple neighboring anchor points.

5.7 Conclusions

In this chapter, we have studied performance optimization of anchor-based range traversing data gathering in WSNs. We formalized the problem as a cost minimization problem constrained by channel capacity, the minimum amount of data uploaded from each sensor and the bound of total sojourn time at all anchor points. We characterized this problem as a pricing mechanism and decomposed it into two simpler subproblems, i.e., SENSOR_i and SENCAR subproblems. We have proved that network cost can be minimized by jointly solving the two subproblems. Correspondingly, we described a pricing-based algorithm that iteratively solves SENSOR_i and the dual problem of SENCAR. In each iteration, the help node sets the shadow price for its local anchor point and derives link prices between neighboring sensors and the anchor point. Each neighboring sensor then determines the payments to minimize its local cost. The minimum network cost can be achieved when reaching the equilibrium that reconciliates the two subproblems. We also proposed an efficient adaptation algorithm for solving the SENSOR_i subproblem at each sensor. Finally, we gave extensive simulation results to validate the efficiency of the proposed algorithm and compare its performance with another data gathering strategy.

Chapter 6

Distributed Network Utility Maximization Algorithms for Mobile Data Gathering in WSNs

In this chapter, we study the performance optimization for anchor-based mobile data gathering, where the SenCar roams over the sensing field by visiting the anchor points and gathers data from nearby sensors via multi-hop transmissions. As routing issue is also taken into consideration, the problem becomes even more complicated than the problem in the previous chapter. We characterize the performance optimization as network utility maximization problems under the constraints of guaranteed network lifetime and data gathering latency. We assume the data utility at each sensor is a function with respect to the total amount of data gathered from this sensor in a data gathering tour. The *network utility* is defined as the aggregation of the data utility of all sensors. We use network utility as a direct metric to evaluate the effectiveness of data gathering strategies. We consider the network utility maximization problem for two cases depending on whether the SenCar has fixed or variable sojourn time at each anchor point. To efficiently solve these problems, we decompose each of them into several subproblems and solve them in a distributed manner, which facilitates the scalable implementation of the optimization algorithms. Finally, we provide extensive numerical results to demonstrate the usage and efficiency of the proposed algorithms and complement our theoretical analysis.

The remainder of this chapter is organized as follows. Section 6.1 provides the introduction of research issue. Section 6.2 discusses the related work. Section 6.3 introduces our system model and formulates the two cases into convex optimization problems. Sections 6.4 and 6.5 present the optimization based distributed algorithms to solve these two problems, respectively. Finally, Section 6.6 gives the performance evaluation results and Section 6.7 concludes the chapter.

6.1 Introduction

In this work, we consider anchor based mobile data gathering as shown in Fig. 6.1, where a SenCar periodically starts a data gathering tour, and in each tour it visits some pre-defined positions called *anchor points* in the field and stays at each anchor point for a period of *sojourn time* to collect data from nearby sensors via multi-hop transmissions. To characterize the data gathering performance, we introduce *network utility*, which is a function quantifying the aggregated "value" of the gathered data from different sensors in a data gathering tour. In practice, the "value" measure can be in terms of information entropy or revenue, which provides the flexibility of modeling user application needs, or a level of "satisfaction" on a certain amount of data from each sensor. In general, good data gathering strategies should ensure an expected network lifetime and have a bounded data gathering latency as well. Therefore, our overall objective is to maximize the network utility under the constraints of guaranteed network lifetime and data gathering latency. To achieve this objective, we will address following three issues that critically affect the data gathering performance. First, from a sensor's point of view, since the SenCar may stay at different anchor points to collect data, how much data should be sent from the sensor to the SenCar at a particular anchor point? Second, in terms of communication efficiency, how to route the data to each anchor point taking into account of energy and link capacity constraints? Third, from the SenCar's point of view, to bound the data gathering latency is actually to constrain the total sojourn time at all anchor points under a threshold. Under these circumstances, what is the optimal sojourn time at each anchor point?

Based on these considerations, in this chapter, we develop optimization based distributed algorithms to find optimal solutions to the above problems. The main contribution of our work can be summarized as follows: (1) Formalize the prob-

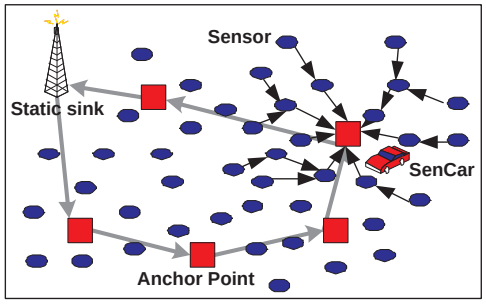


Figure 6.1: Illustration of anchor-based mobile data gathering.

lem of finding optimal mobile data gathering strategies as a network utility maximization problem with guaranteed network lifetime and data gathering latency. (2) Propose solutions to this problem for two cases, where the SenCar spends fixed and variable sojourn time at each anchor point, respectively. The former case is essentially a joint problem of rate control and optimal routing, and the latter case involves the integration of data control, routing and sojourn time allocation. (3) Present distributed algorithms for the two cases to facilitate scalable implementations, in which each sensor only needs to exchange limited information with its direct neighbors and the SenCar. (4) Provide extensive numerical results to demonstrate the usage and efficiency of the proposed distributed algorithms.

6.2 Related Work

There has been some work in the literature on optimization based data gathering algorithms for both relay routing and mobile data gathering.

For relay routing, Madan and Lall [50] proposed distributed algorithms based on a dual decomposition approach to computing an optimal routing that maximizes network lifetime. Hou, et. al [52] studied the lexicographical max-min (LMM) rate allocation problem and developed a polynomial-time algorithm by exploiting the parametric analysis. They also showed the existence of an elegant duality between the LMM rate allocation problem and the LMM node lifetime problem so that it is sufficient to solve only one of the two problems. In [53], Hua and Yum considered optimizing data aggregation and maximum lifetime routing together, which aims to reduce the traffic across the network and balance the traffic to avoid overwhelm-

ing bottleneck nodes. Zhang, et. al [54] studied the joint problem of sensing rate control, data routing and energy allocation to maximize the system utility. They first mapped the combined sensing/routing problem into a unified routing problem and then used a penalty function approach to solving the remaining joint routing and energy allocation problem. Wu, et. al [55] studied the construction of a data gathering tree to maximize the network lifetime. They showed the problem is NP-complete and designed a provably near-optimal algorithm for it, which first starts from building an arbitrary tree and then iteratively reduces the load on bottleneck nodes. Sadagopan and Krishnamachari [56] examined the problem of maximizing data extraction from energy-limited sensor networks, which requires attention to "data-awareness" in addition to "energy-awareness." They formulated this problem as a linear program and presented an iterative approximation algorithm for it. Chen, et. al [57] studied the max-min optimal rate assignment problem in a sensor network, where all possible forwarding paths are considered. They provided an iterative linear programming solution, which finds the optimal rate assignment and a forwarding schedule that implements the assignment in a low-rate sensor network. Liu, et. al [58] addressed the maximal lifetime scheduling problem in sensor surveillance systems. They proposed an optimal solution to find the target-watching schedule for sensors, in which the workload matrix obtained by using the linear programming technique is first decomposed into a sequence of schedule matrices, and then the surveillance trees are determined based on these schedule matrices.

For mobile data gathering, Xing, et. al [36] proposed approximate algorithms to minimize the distance of local multi-hop routing under the constraint that the tour length of the mobile collector is no more than a threshold. Zhao, et. al [110] studied the data gathering latency minimization problem with the joint design of mobility control and space-division multiple access (SDMA) technique. They proposed algorithms for different scenarios by balancing the tradeoff between the shortest moving tour of the mobile collector and the full utilization of SDMA for data transmissions. Gatzianas and Georgiadis [59] presented a distributed algorithm for maximum lifetime routing in a sensor network with a mobile sink by assuming a constant data rate for each sensor and defining the network lifetime as the total sojourn time of the mobile collector.

Our work in this chapter focuses on the performance optimization of anchor based mobile data gathering and differs from the earlier work in the following as-

pects: (1) In contrast to the fixed data rate assumed in most existing work [50][53][59], each sensor in our scheme may adopt a variable data rate to avoid congestion. (2) Different from the optimal routing with a static data sink [50]-[58], in our setting, each sensor can use any set of possible routes to reach the SenCar. This implies that on one hand, each sensor has options to send data to the SenCar at different anchor points; on the other hand, the routing path for each specific sensor-anchor point pair should be energy efficient. (3) In addition to the energy constraint that ensures network lifetime [50][52][53][36][59], we also impose the latency constraint on mobile data gathering and jointly optimize the sojourn time at each anchor point with the rate control and routing problems. To the best of our knowledge, this is the first work that explores such optimization and systematically provides solutions to these problems.

6.3 System Model and Problem Formulations

6.3.1 System Model

Consider a network with a set of static sensors, denoted by $\mathcal{N}$, and a set of anchor points, denoted by $\mathcal{A}$. In a periodic data gathering tour, the SenCar, denoted by s, roams over the field and collects data by visiting each anchor point in a specified sequence.

To capture the characteristics of the SenCar movements over different anchor points, we model the sensor network with the SenCar located at an anchor point a ($a \in \mathcal{A}$) by a directed graph $G^a(V^a, E^a)$. $V^a = \mathcal{N} \bigcup \{s^a\}$ and it represents the set of nodes, including all the sensors and the SenCar at anchor point a (denoted by s^a). $E^a = \{(i,j)|i,j \in V^a\}$, which is the set of directed links among the sensors and the SenCar. Sensor i ($i \in \mathcal{N}$) generates information for the SenCar at a data rate of q_i^a when the SenCar moves to anchor point a. The SenCar will stay at anchor point a for a period of sojourn time t^a to gather the data routed to it. To ensure that all the sensors can reach the SenCar in a data gathering tour in a multi-hop fashion, we assume that there exist routing paths from each sensor to at least one anchor point in the network. This can be achieved by properly placing the anchor points, such as putting them within the transmission ranges of the sensors that dominate the network connectivity [36] or evenly distributed over the sensing

Table 6.1: List of notations used in problem formulations.

Notation	Definition
$\mathcal{N}$	Set of sensors
$\mathcal{A}$	Set of anchor points
s^a	SenCar located at anchor point a
t^a	Sojourn time of SenCar at anchor pint a
ΔT	Bound of data gathering latency, i.e., maximum total sojourn time at all anchor points in a data gathering tour
E'_i	Energy budget sensor i can use in a data gathering tour, which guarantees specified network lifetime T
q_i^a	Data rate of sensor i when SenCar is located at anchor point a
Q_i	Upper bound of the data rate of sensor i
f_{ij}^a	Flow rate over link (i, j) when SenCar is at anchor point a
F_{ij}	Capacity of link (i, j)
e_{ij}	Energy consumed for transmitting a unit flow over link (i, j)
x_{ij}^a	Flow over link (i, j) when SenCar is located at anchor point a
y_i	Total data gathered from sensor i in a data gathering tour
Y_i	Upper bound of total data from sensor i in a data gathering tour
ϕ_i^a	Data split variable, i.e., fraction of the data of sensor i destined to SenCar at anchor point a

field with sufficiently small intervals [110].

Assume that sensor i has non-renewable battery energy E_i, which would be gradually depleted as the sensor transmits its own data and relays data for others. To guarantee a specific network lifetime T, we impose an energy expenditure budget E'_i for sensor i, which is the maximum energy the sensor can consume in a data gathering tour. If K data gathering tours would be performed within the network lifetime, this budget can be approximated as $E'_i = (E_i - P_{s_i}T)/K$, where P_{s_i} is the sensing power of sensor i. Moreover, in some applications, it is expected that the time cost of data gathering should be bounded. It is equivalent to constraining the total sojourn time at all anchor points to no more than a threshold, i.e., $\sum_{a\in\mathcal{A}} t^a \leq \Delta T$, where ΔT is called the bound of data gathering latency.

To facilitate our studies, we use *data utility* to characterize the impact of the data from a sensor on the overall data gathering performance. We define the data utility function of sensor i, $U_i(\cdot)$, as a strictly concave, increasing and twice-differentiable function with respect to the total amount of data gathered from sensor i in a data gathering tour (i.e., $\sum_{a\in\mathcal{A}} q_i^a t^a$). There are several typical forms that can be used for $U_i(\cdot)$, such as $U_i = w_i \log(1 + \sum_{a\in\mathcal{A}} q_i^a t^a)$ [102] or $U_i = -w_i(\sum_{a\in\mathcal{A}} q_i^a t^a)^{-0.5}$

[54], where w_i is the utility weight of at sensor i, reflecting the importance of its data. In our discussions and formulations, we will simply use $U_i(\cdot)$ as the general form for the data utility function and adopt $U_i = w_i \log(1 + \sum_{a \in \mathcal{A}} q_i^a t^a)$ in the simulations later. Accordingly, the *network utility* is defined as the aggregation of the data utility of all sensors.

In this work, we are interested in maximizing the network utility while guaranteeing a given network lifetime and data gathering latency. In the following, we first consider the case where the SenCar spends fixed sojourn time at each anchor point and then extend our study to the case where the sojourn time at each anchor point can vary. Each of these cases corresponds to an optimization problem. The notations used in the problem formulations are given in Table 6.1.

6.3.2 Formulation of Network Utility Maximization Problem with Fixed Sojourn Time at Each Anchor Point (NUM-FT)

In this subsection, we consider the case that the SenCar spends fixed sojourn time at each anchor point, i.e., $\forall a \in \mathcal{A}$, t^a is given. Our objective aims to find the proper data rate for each sensor and the flow rate for each link when the SenCar moves over different anchor points such that the network utility can be maximized. Clearly, this problem is essentially a joint rate control and routing problem, which can be formally defined as follows. Since a longer sojourn time generally corresponds to more gathered data, without loss of generality, we assume that the total sojourn time at all anchor points reaches the data gathering latency bound.

Definition 3. *Network Utility Maximization Problem with Fixed Sojourn Time at Each Anchor point (NUM-FT).* Given a set of sensors $\mathcal{N}$, a set of anchor points $\mathcal{A}$, and the sojourn time at each anchor point t^a ($a \in \mathcal{A}$ and $\sum_{a \in \mathcal{A}} t^a = \Delta T$), find: (1) data rate q_i^a of sensor i when the SenCar is at anchor point a; (2) flow rate f_{ij}^a over link $(i, j) \in E^a$ destined to the SenCar at anchor point a, such that network utility $\sum_{i \in \mathcal{N}} U_i(\sum_{a \in \mathcal{A}} q_i^a t^a)$ is maximized.

The NUM-FT problem can be formulated as follows.

$$\textbf{NUM-FT:} \quad \max \sum_{i \in \mathcal{N}} U_i\Big(\sum_{a \in \mathcal{A}} q_i^a t^a\Big) \tag{6.1}$$

$$\text{s.t.} \quad \sum_{j:(i,j)\in E^a} f_{ij}^a = q_i^a + \sum_{j:(j,i)\in E^a} f_{ji}^a, \ \forall i \in \mathcal{N}, \forall a \in \mathcal{A} \tag{6.2}$$

$$\sum_{a\in\mathcal{A}} \sum_{j:(i,j)\in E^a} f_{ij}^a e_{ij} t^a \leq E_i', \qquad \forall i \in \mathcal{N} \tag{6.3}$$

$$0 \leq f_{ij}^a \leq F_{ij}, \quad \forall i \in \mathcal{N}, \forall j : (i,j) \in E^a, \forall a \in \mathcal{A} \tag{6.4}$$

The constraints in the NUM-FT problem can be explained as follows.

- Flow conservation constraint: Eq. (6.2) shows that at each sensor node for each anchor point, the aggregated outgoing link flow rates equal the local data rate plus the incoming link flow rates.
- Energy constraint: Eq. (6.3) enforces that the total energy consumed by sensor i in a data gathering tour would not exceed energy budget E_i'.
- Link capacity constraint: Eq. (6.4) shows that link flow rate f_{ij}^a is restricted by link capacity F_{ij}.

Since constraints (6.2)-(6.4) define a convex set and the objective function is concave with respect to q_i^a, NUM-FT is a convex optimization problem. We assume that the Slater's condition [98] for constraint qualification is satisfied, i.e., there exist feasible solutions of q and f such that the constraints hold with strict inequality. Under this assumption, the strong duality holds, which implies that the optimal values of the primal and dual problems are equal. Hence, the distributed algorithm for the NUM-FT problem can be obtained by formulating and solving the corresponding Lagrange dual problem, as will be seen in Section 6.4.

6.3.3 Formulation of Network Utility Maximization Problem with Variable Sojourn Time at Each Anchor Point (NUM-VT)

We now consider the case of maximizing network utility with variable sojourn time at each anchor point, i.e., $\forall a \in \mathcal{A}$, t^a is a variable, and refer to it as the NUM-VT problem. It can be similarly formulated to the NUM-FT problem except that the data gathering latency constraint is added on variable t^a, i.e., $\sum_{a\in\mathcal{A}} t^a \leq \Delta T$. However, since t^a, q_i^a and f_{ij}^a are all variables, the problem now contains coupling variables in both objective function and constraints. Also, the objective function

is no longer concave with respect to q_i^a and t^a, since its Hessian is not negative semidefinite [96]. In order to make the NUM-VT problem solvable, we introduce auxiliary variables x_{ij}^a, y_i and ϕ_i^a and define them as follows.

$$x_{ij}^a = f_{ij}^a t^a, \quad y_i \phi_i^a = q_i^a t^a, \quad \phi_i^a \geq 0, \quad \sum_a \phi_i^a = 1$$

where x_{ij}^a represents the flow amount over link (i, j) destined to the SenCar at anchor point a, y_i is the total amount of data generated by sensor i in a data gathering tour, and ϕ_i^a is a data split variable with $\phi_i^a \geq 0$ and $\sum_a \phi_i^a = 1$, which controls the fraction of the data of sensor i that routes to the SenCar at anchor point a. As will be described later, by multiplying the flow conservation and link capacity constraints by t^a and using these auxiliary variables, we can reformulate the NUM-VT problem into a convex optimization problem with respect to x, y, ϕ and t. Thus, according to this formulation, the NUM-VT problem is essentially a joint data control, routing and sojourn time allocation problem as follows.

Definition 4. *Network Utility Maximization Problem with Variable Sojourn Time at Each Anchor Point (NUM-VT).* Given a set of sensors $\mathcal{N}$, a set of anchor points $\mathcal{A}$, and the bound of data gathering latency ΔT, find: (1) sojourn time t^a at each anchor point; (2) total amount of data y_i generated by sensor i in a data gathering tour; (3) data split variable ϕ_i^a; (4) flow amount x_{ij}^a over link $(i, j) \in E^a$ destined to the SenCar at anchor point a, such that network utility $\sum_{i \in \mathcal{N}} U_i(y_i)$ is maximized.

The NUM-VT problem now can be expressed as

$$\textbf{NUM-VT:} \quad \max \sum_{i \in \mathcal{N}} U_i(y_i) \tag{6.5}$$

$$\text{s.t.} \sum_{j:(i,j)\in E^a} x_{ij}^a = y_i\phi_i^a + \sum_{j:(j,i)\in E^a} x_{ji}^a,\ \forall i \in \mathcal{N}, \forall a \in \mathcal{A} \tag{6.6}$$

$$\sum_{a\in\mathcal{A}}\sum_{j:(i,j)\in E^a} x_{ij}^a e_{ij} \le E_i', \qquad \forall i \in \mathcal{N} \tag{6.7}$$

$$0 \le x_{ij}^a \le F_{ij}t^a, \forall i \in \mathcal{N}, \forall j : (i,j) \in E^a, \forall a \in \mathcal{A} \tag{6.8}$$

$$\sum_{a\in\mathcal{A}} \phi_i^a = 1, \qquad \forall i \in \mathcal{N} \tag{6.9}$$

$$t^a \ge 0, \qquad \forall a \in \mathcal{A} \tag{6.10}$$

$$\sum_{a\in\mathcal{A}} t^a \le \Delta T. \tag{6.11}$$

Clearly, since the objective function $\sum_i U_i(y_i)$ is strictly concave with respect to y_i, the NUM-VT problem now is a strictly convex optimization problem, however, with non-linear constraints (see Eq. (6.6)). To decompose the coupling variables y_i and ϕ_i^a in the flow conservation constraints, we will take a hierarchical decomposition approach to separating the NUM-VT problem into a repeated two-level optimization problem [103][106], which first maximizes the network utility over x, y and t while keeping ϕ fixed, then maximizes the network utility by updating ϕ. The details of this approach will be discussed in Section 6.5.

6.4 Distributed Algorithm for NUM-FT Problem

Having formulated the NUM-FT problem as in (6.1)-(6.4), in this section, we give a fully distributed algorithm to solve it. We utilize the subgradient algorithm based on the dual-decomposition method [98], which is an efficient technique for convex program and can naturally achieve the distributed implementation.

We form the dual problem by introducing Lagrangian multipliers $\lambda \in \mathrm{R}^{|\mathcal{N}|\times|\mathcal{A}|}$ for the flow conservation constraints. This results in the partial Lagrangian as

$$\begin{aligned} L(q,f,\lambda) &= \sum_i U_i(\sum_a q_i^a t^a) - \sum_a\sum_i \lambda_i^a(q_i^a + \sum_j f_{ji}^a - \sum_j f_{ij}^a) \\ &= [\sum_{i\in\mathcal{N}} U_i(\sum_{a\in\mathcal{A}} q_i^a t^a) - \sum_{a\in\mathcal{A}}\sum_{i\in\mathcal{N}} \lambda_i^a q_i^a] + \\ &\quad [\sum_{a\in\mathcal{A}}\sum_{i\in\mathcal{N}}\sum_{j:(i,j)\in E^a} (\lambda_i^a - \lambda_j^a) f_{ij}^a] \end{aligned}$$

Here, the Lagrangian multiplier λ_i^a can be interpreted as the "congestion price"

at sensor i for the SenCar at anchor point a. Defining the dual function $D(\lambda) = \max_{q,f} L(q, f, \lambda)$ with constraints (6.3)-(6.4), we obtain the dual problem as follows.

$$\min_{\lambda \succeq 0} D(\lambda) = \min_{\lambda \succeq 0} \max_{q,f} L(q, f, \lambda)$$

One immediate observation is that the dual function can be decomposed into two sets of subproblems (see the two terms in the Lagrangian). One set is the rate control subproblems in terms of rate variables q while another set contains the routing subproblems to find optimal flow variables f. Each of them is independently solvable by a sensor. We solve each set of the subproblems in the dual function by subgradient algorithms and finally obtain the joint rate control and routing algorithm for the primal NUM-FT problem.

We start with a set of initial non-negative Lagrangian multipliers $\lambda_i^a(0)$ for all $i \in \mathcal{N}$ and $a \in \mathcal{A}$. During each iteration k of the subgradient algorithm, given the current Lagrangian multipliers $\lambda_i^a(k)$, we solve the subproblems as follows.

Rate control subproblem: The dual function contains $|\mathcal{N}|$ rate control subproblems, one for each sensor. In the following, we describe our algorithm to solve the subproblem in the context of sensor i. Since $0 \leq q_i^a \leq \sum_j f_{ij}^a \leq \sum_j F_{ij}$, each q_i^a is within a closed domain. Thus, we define a loose upper bound Q_i for each q_i^a. Accordingly, for sensor i, the rate control subproblem is given by

$$\max_q U_i(\sum_{a \in \mathcal{A}} q_i^a t^a) - \sum_{a \in \mathcal{A}} \lambda_i^a q_i^a \quad \text{s.t. } 0 \leq q_i^a \leq Q_i, \forall a \in \mathcal{A} \tag{6.12}$$

This subproblem is a convex program. However, the objective is not strictly concave with respect to q_i^a such that the solution may not be unique. This difficulty is due to the linearity of $\sum_a q_i^a t^a$, with respect to which, function $U_i(\cdot)$ is in the primal NUM-FT problem so that the dual function is not differentiable at every point [102]. One method to circumvent such difficulty of lack of strict concavity is to subtract a small convex quadratic regularization term, e.g., $\epsilon \sum_i \sum_a (q_i^a)^2$, from the primal objective function [101]. However, this more or less changes the original problem and typically results in significant oscillation due to the small value of ϵ. More sophisticated approaches, including proximal optimization algorithm and augmented Lagrangian methods [97], provide effective ways to conquer such a problem. However, they have much higher complexity. Thus, instead of using these approaches,

we propose an efficient search algorithm based on the Karush-Kuhn-Tucker (KKT) conditions [96] to find an optimal solution. Since the rate control subproblem in (6.12) is concave, the solutions that satisfy the KKT conditions are sufficient to be optimal for both itself and its dual problem [96][98].

For clarity, we use $g(q_i)$ to denote the objective function in (6.12) and let $q_i^{a*} = \{q_i^{a*} | a \in \mathcal{A}\}$ be the optimal solution. Introducing Lagrangian multiplier σ_a for constraint $q_i^a \leq Q_i$ and σ_a' for constraint $q_i^a \geq 0$, we obtain the KKT conditions as follows for all $a \in \mathcal{A}$,

$$\begin{aligned} &U_i'\left(\textstyle\sum_{a\in\mathcal{A}} q_i^{a*}t^a\right)t^a - \lambda_i^{a*} - \sigma_a^* + \sigma_a'^* = 0, \\ &\sigma_a^*\left(q_i^{a*} - Q_i\right) = 0, \\ &\sigma_a'^* q_i^{a*} = 0, \\ &\sigma_a^* \geq 0, \sigma_a'^* \geq 0. \end{aligned} \tag{6.13}$$

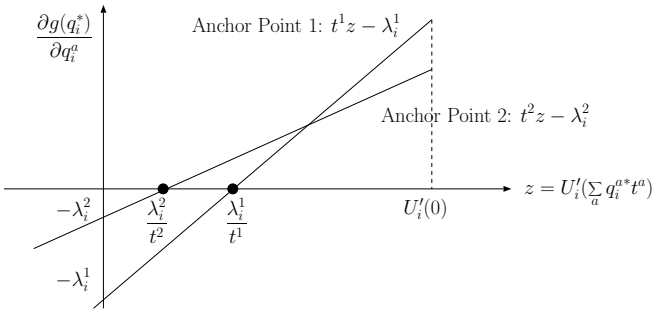
For $0 \leq q_i^{a*} \leq Q_i$, we have the following three cases according to the KKT conditions in (6.13).

1. If $\sigma_a^* > 0$, then $q_i^{a*} = Q_i$, $\sigma_a'^* = 0$, and $\sigma_a^* = U_i'(\sum_a q_i^{a*}t^a)t^a - \lambda_i^a = \frac{\partial g(q_i^*)}{\partial q_i^a} > 0$;

2. If $\sigma_a^* = 0$ and $\sigma_a'^* = 0$, then $q_i^{a*} \in [0, Q_i]$, and $U_i'(\sum_a q_i^{a*}t^a)t^a - \lambda_i^a = \frac{\partial g(q_i^*)}{\partial q_i^a} = 0$;

3. If $\sigma_a'^* > 0$, then $q_i^{a*} = 0$, $\sigma_a^* = 0$, and $-\sigma_a'^* = U_i'(\sum_a q_i^{a*}t^a)t^a - \lambda_i^a = \frac{\partial g(q_i^*)}{\partial q_i^a} < 0$.

From the above cases, we find that the exact value or the range of q_i^{a*} corresponds to different values of $\frac{\partial g(q_i^*)}{\partial q_i^a}$. For brevity, we use variable z to denote $U_i'(\sum_a q_i^{a*}t^a)$. Since t^a and λ_i^a are considered as constants here, $\frac{\partial g(q_i^*)}{\partial q_i^a}$ is a linear function of z. As U_i is a strictly concave function, $z = U_i'(\sum_a q_i^{a*}t^a)$ decreases with the increase of $\sum_a q_i^{a*}t^a$. Since $q_i^{a*}, t^a \geq 0$, $z \leq U_i'(0)$. For a given value $\tilde{z}$ of z, each $\frac{\partial g(q_i^*)}{\partial q_i^a}$ can be determined such that we can correspondingly find the exact value or the range of each q_i^{a*} by ascribing it to one of the above three cases. Moreover, q_i^{a*}'s also satisfy that $\sum_a q_i^{a*}t^a = U_i'^{-1}(\tilde{z})$, which we call the linear combination (LC) condition. Thus, finding the solutions for the q_i^{a*}'s is actually equivalent to searching for a suitable value $\tilde{z}$ of z, by which the derived q_i^{a*}'s satisfy both the KKT and LC conditions.



(a) $\frac{\partial g(q_i^*)}{\partial q_i^1}$ and $\frac{g(\partial q_i^*)}{\partial q_i^2}$ as the linear functions of z.

Cases		q_i^{1*} by KKT	q_i^{2*} by KKT	$\sum_{a=1,2} q_i^{a*} t^a$ by LC
I	$z < \lambda_i^2/t^2$	0	0	$U_i'^{-1}(\tilde{z})$
II	$z = \lambda_i^2/t^2$	0	$[0, Q_i]$	
III	$\lambda_i^2/t^2 < z < \lambda_i^1/t^1$	0	Q_i	
IV	$z = \lambda_i^1/t^1$	$[0, Q_i]$	Q_i	
V	$\lambda_i^1/t^1 < z \leq U'(0)$	Q_i	Q_i	

(b) Five cases for z.

Figure 6.2: An example to illustrate the search algorithm at sensor i for the rate control subproblem in the scenario of two anchor points.

Next, we describe the basic idea of our search algorithm using the example in Fig. 6.2, where there are two anchor points, denoted by 1 and 2, respectively. Without loss of generality, we assume that $\lambda_i^1 > \lambda_i^2$, $t^1 > t^2$ and $U_i'(0) > \lambda_i^1/t^1 > \lambda_i^2/t^2$. Accordingly, $\frac{\partial g(q_i^*)}{\partial q_i^1}$ and $\frac{\partial g(q_i^*)}{\partial q_i^2}$ as linear functions of z can be plotted as shown in Fig. 6.2(a). Based on their z-intercepts, i.e., λ_i^1/t^1 and λ_i^2/t^2, we can divide the domain of z into three intervals and two intersection points, which are listed as five cases in Fig. 6.2(b). To find $\tilde{z}$, we investigate each of these cases. For the cases where z is in an interval (e.g., Cases I, III, or V), $\frac{\partial g(q_i^*)}{\partial q_i^1}$ and $\frac{\partial g(q_i^*)}{\partial q_i^2}$ are both non-zeros. The exact values of q_i^{1*} and q_i^{2*} can be determined by the KKT conditions. Thus, we only need to examine whether there is a value $\tilde{z}$ in the current interval that satisfies the LC condition. For instance, in Case I where $z < \lambda_i^2/t^2$, since $\frac{\partial g(q_i^*)}{\partial q_i^1} < 0$ and $\frac{\partial g(q_i^*)}{\partial q_i^2} < 0$, by the KKT conditions, $q_i^{1*} = 0$ and $q_i^{2*} = 0$. Thus, $\sum_{a=1,2} q_i^{a*} t^a = 0$. However, since $z < \lambda_i^2/t^2 < U_i'(0)$, $U_i'^{-1}(z) > U_i'^{-1}(0) = 0$ always holds. This implies that there is no such $\tilde{z}$ in this interval meeting the requirement that $\sum_{a=1,2} q_i^{a*} t^a = U_i'^{-1}(\tilde{z})$. In other cases where z is at an intersection point (e.g., Cases II or IV), by the KKT conditions, one of q_i^{1*} and q_i^{2*} is a determined value,

the other is in the domain of $[0, Q_i]$. In such cases, since z has a unique value, the possible solution for the undetermined q_i^{a*} ($a = 1$ or 2) can be derived by the LC condition. Based on the value of z, we can estimate whether this solution would fall into the valid domain of $[0, Q_i]$. Let us take Case II where $z = \lambda_i^2/t^2$ as an example. In this case, since $\frac{\partial g(q_i^*)}{\partial q_i^1} < 0$ and $\frac{\partial g(q_i^*)}{\partial q_i^2} = 0$, by the KKT conditions, $q_i^{1*} = 0$ and $q_i^{2*} \in [0, Q_i]$. If $0 \leq U_i'^{-1}(z) \leq Q_i t^2$, the solution of q_i^{2*} that is set to $1/t^2 \cdot U_i'^{-1}(z)$ by the LC condition would fall into the valid domain of $[0, Q_i]$. This means that $\tilde{z}$ is found, i.e., $\tilde{z} = \lambda_i^2/t^2$. And $q_i^{1*} = 0$ and $q_i^{2*} = 1/t^2 \cdot U_i'^{-1}(\lambda_i^2/t^2)$ would be the optimal solutions to the rate control subproblem in (6.12). Otherwise, there is no valid solution for q_i^{2*} and the subsequent cases on z should be further examined in a similar way until $\tilde{z}$ is found. For the rare case where $z = \lambda_i^1/t^1 = \lambda_i^2/t^2$, by the KKT conditions, $q_i^{1*}, q_i^{2*} \in [0, Q_i]$. If $0 \leq U_i'^{-1}(z) \leq Q_i(t^1 + t^2) = Q_i \Delta T$, there would exist valid values for q_i^{1*} and q_i^{2*}. However, the possible values of q_i^{1*} and q_i^{2*} that are able to ensure the LC condition may not be unique. We can randomly choose one of them. For clarity, we summarize the details of the search algorithm at sensor i in Table 6.2. It is apparent that to find the solutions for q_i^{a*}'s, sensor i only needs to examine at most $|\mathcal{A}| + 1$ intervals and $|\mathcal{A}|$ intersection points on z. Thus, the search algorithm is a linear algorithm with $O(|\mathcal{A}|)$ time complexity.

Routing subproblem: The dual function also contains $|\mathcal{N}|$ routing subproblems, each for a sensor. Again, we consider the routing subproblem at sensor i as follows.

$$\begin{aligned} \max_f \quad & \sum_{a \in \mathcal{A}} \sum_{j:(i,j) \in E^a} (\lambda_i^a - \lambda_j^a) f_{ij}^a \\ \text{s.t.} \quad & \sum_{a \in \mathcal{A}} \sum_{j:(i,j) \in E^a} f_{ij}^a e_{ij} t^a \leq E_i', \\ & 0 \leq f_{ij}^a \leq F_{ij}, \forall j : (i,j) \in E^a, \forall a \in \mathcal{A} \end{aligned} \tag{6.14}$$

Clearly, this subproblem is a linear program. If we consider $(\lambda_i^a - \lambda_j^a)$ as the gain of link $(i, j) \in E^a$ (when $j = s^a$, we can consider $\lambda_j^a = 0, \forall a \in \mathcal{A}$), this subproblem can be easily solved by a greedy algorithm as shown in Table 6.3. The approach can be intuitively interpreted as that each sensor always allocates the maximum possible rate under the energy and link capacity constraints to a link that has the largest link gain among all its outgoing links to different anchor points. Apparently, as each sensor needs to investigate the gains of all its outgoing links for different anchor points, the time complexity of the greedy algorithm at sensor i for the routing

Table 6.2: Search algorithm for rate control subproblem.

Divide the valid domain of z, i.e., $(-\infty, U_i'(0)]$, into cases based on z-intercepts of $\frac{\partial g(q_i^*)}{\partial q_i^a}$'s (i.e., $\lambda_i^a / t^a, \forall a \in \mathcal{A}$);
For each case of z **do**
  Examine the value of $\frac{\partial g(q_i^*)}{\partial q_i^a}$ for all $a \in \mathcal{A}$;
  If $\frac{\partial g(q_i^*)}{\partial q_i^a} \neq 0$, for all $a \in \mathcal{A}$
    // *The case that z is in an interval*
    The exact value of each q_i^{a*} can be determined by the KKT conditions;
    If $\tilde{z} = U_i'(\sum_a q_i^{a*} t^a)$ that falls in the current interval
      Current value of each q_i^{a*} is optimal to (6.12);
      Break;
    end If
  else
    // *The case that z is at an intersection point*
    Find subset $\mathcal{B} \subseteq \mathcal{A}$ that $\mathcal{B} = \left\{ b \,\middle|\, \frac{\partial g(q_i^*)}{\partial q_i^b} = 0, b \in \mathcal{A} \right\}$;
    By the KKT conditions, the value of q_i^{a*} ($a \in \mathcal{A} \backslash \mathcal{B}$) can be determined and $q_i^{b*} \in [0, Q_i], \forall b \in \mathcal{B}$;
    Calculate $\sum_{b \in \mathcal{B}} q_i^{b*} t^b = U_i'^{-1}(z) - \sum_{a \in \mathcal{A} \backslash \mathcal{B}} q_i^{a*} t^a$;
    If $0 \leq \sum_{b \in \mathcal{B}} q_i^{b*} t^b \leq Q_i \sum_{b \in \mathcal{B}} t^b$
      // *There exist valid values for* $q_i^{b*}, \forall b \in \mathcal{B}$
      Set $\tilde{z}$ by the current value of z;
      If $|\mathcal{B}| = 1$, for $b \in \mathcal{B}$,
        $q_i^{b*} = \frac{1}{t^b}(U_i'^{-1}(\tilde{z}) - \sum_{a \in \mathcal{A} \backslash \mathcal{B}} q_i^{a*} t^a)$,
      else
        q_i^{b*}'s ($b \in \mathcal{B}$) are randomly set to ensure LC condition;
      end If
      Current value for each q_i^{a*} is optimal to (6.12);
      Break;
    end If
  end If
end For

Table 6.3: Greedy algorithm for routing subproblem.

Set f_{ij}^a to zero, $\forall j : (i,j) \in E^a, \forall a \in \mathcal{A}$;
Find $X_i = \left\{ (j,a) \middle| \lambda_i^a - \lambda_j^a > 0, \forall j : (i,j) \in E^a, \forall a \in \mathcal{A} \right\}$;
Initialize the remaining available energy: $E_r = E_i'$;
While ($X_i \neq \Phi$ & $E_r > 0$)
  $(\tilde{j}, \tilde{a})_i = \arg \max_{(j,a) \in X_i} (\lambda_i^a - \lambda_j^a)$;
  $f_{i,\tilde{j}}^{\tilde{a}} = \min \left\{ \frac{E_r}{e_{i\tilde{j}} t^{\tilde{a}}}, F_{ij} \right\}$;
  Update X_i by removing $(\tilde{j}, \tilde{a})_i$ from it;
  Update E_r by setting $E_r = E_r - f_{i\tilde{j}}^{\tilde{a}} e_{i\tilde{j}} t^{\tilde{a}}$;
end While

subproblem is $O(\sum_{a \in \mathcal{A}} \deg_a^+(i))$, where $\deg_a^+(i)$ is the outdegree of sensor i in the directed graph $G^a(V^a, E^a)$.

Lagrangian multiplier update: In each iteration of the subgradient algorithm, sensor i solves the subproblems in (6.12) and (6.14) with the current Lagrangian multiplier $\lambda_i^a(k)$. Then, sensor i updates the Lagrangian multipliers as follows and sends them to its direct neighbors to facilitate the computing of q and f in the next iteration.

$$\lambda_i^a(k+1) = [\lambda_i^a(k) + \theta(k)(q_i^a(k) + \sum_j f_{ji}^a(k) - \sum_j f_{ij}^a(k))]^+ \tag{6.15}$$

where $[\cdot]^+$ denotes the projection onto the non-negative orthant and $\theta(k)$ is a properly chosen scalar stepsize for subgradient iteration k. In our algorithm, we choose the diminishing stepsizes, i.e., $\theta(k) = d/(b+ck), \forall k, c, d > 0, b \geq 0$, where b, c and d are adjustable parameters that regulate the convergence speed. The diminishing stepsize can guarantee the convergence regardless of the initial value of λ [98].

Recovery of primal solutions: Note that the subproblems in (6.12) and (6.14) are not strictly concave, which implies that the values in the optimal solution of the Lagrangian dual cannot be directly applied to the primal NUM-FT problem. In view of this, we apply the method introduced in [100] to recover the primal solutions. For the k_{th} subgradient iteration, we compose a primal feasible $\hat{f}_{ij}^a(k)$ as follows.

$$\hat{f}_{ij}^a(k) = \frac{1}{k} \sum_{h=1}^{k} f_{ij}^a(h) = \begin{cases} f_{ij}^a(1) & k = 1 \\ \frac{k-1}{k} \hat{f}_{ij}^a(k-1) + \frac{1}{k} f_{ij}^a(k) & k > 1 \end{cases} \tag{6.16}$$

Table 6.4: Distributed algorithm for the NUM-FT problem.

For each sensor $i \in \mathcal{N}$ **do**
  Initialize Lagrangian multipliers $\lambda_i^a(0)$ for all $a \in \mathcal{A}$ to non-negative values;
  Repeat: for all $j : (i,j) \in E^a$ and $a \in \mathcal{A}$
    Determine $q_i^a(k)$ by search algorithm in Table 6.2;
    Determine $f_{ij}^a(k)$ by greedy algorithm in Table 6.3;
    Update Lagrangian multipliers $\lambda_i^a(k+1)$ by Eq. (6.15);
    Compute primal feasible $\hat{f}_{ij}^a(k)$ by Eq. (6.16);
    Send updated Lagrangian multipliers to its neighbors;
  Until sequence $\{\lambda(k)\}$ converges to λ^* and sequence $\{\hat{f}(k)\}$ converges to $\hat{f}^*$;
  Exchange $\hat{f}_{ij}^{a\,*}$'s with its neighbors and compute the optimal data rates by $q_i^{a*} = \sum_j \hat{f}_{ij}^{a\,*} - \sum_j \hat{f}_{ji}^{a\,*}$ for all $a \in \mathcal{A}$;
end For

It was proved in [100] that when the diminishing stepsize is used, any accumulation point of the sequence $\{\hat{f}_{ij}^a\}$ generated by (6.16) is feasible to the primal problem and $\{\hat{f}_{ij}^a\}$ can converge to a primal optimal solution. Therefore, the optimal flow rate of each outgoing link of sensor i can be obtained when $\{\hat{f}_{ij}^a\}$ converges to $\hat{f}_{ij}^{a\,*}$. Finally, sensor i can recover the optimal data rates by plugging each $\hat{f}_{ij}^{a\,*}$ back to the flow conservation constraint.

We summarize the distributed algorithm for the NUM-FT problem in Table 6.4, from which we can see that each subproblem can be efficiently solved in a distributed manner, which only requires limited computation at each sensor and local exchange of Lagrangian multipliers among the direct neighbors.

6.5 Distributed Algorithm for NUM-VT Problem

In this section, we consider the NUM-VT problem. As aforementioned, we take a hierarchical decomposition approach to separating the NUM-VT problem into two levels of optimization [103][106] to decompose the coupling variables y_i and ϕ_i^a in the flow conservation constraints. At the lower level, we consider the following problem, denoted by NUM-VT(a), that maximizes the network utility over variables

x, y and t with a fixed ϕ.

$$\textbf{NUM-VT(a):}\ \max \sum_{i\in\mathcal{N}} U_i(y_i) \tag{6.17}$$

$$\text{s.t.} \sum_j x_{ij}^a = y_i\phi_i^a + \sum_j x_{ji}^a, \quad \forall i \in \mathcal{N}, \forall a \in \mathcal{A} \tag{6.18}$$

$$\sum_{a\in\mathcal{A}} \sum_{j:(i,j)\in E^a} x_{ij}^a e_{ij} \le E_i', \quad \forall i \in \mathcal{N} \tag{6.19}$$

$$0 \le x_{ij}^a \le F_{ij}t^a, \forall i \in \mathcal{N}, \forall j : (i,j) \in E^a, \forall a \in \mathcal{A} \tag{6.20}$$

$$t^a \ge 0, \qquad \forall a \in \mathcal{A} \tag{6.21}$$

$$\sum_{a\in\mathcal{A}} t^a \le \Delta T. \tag{6.22}$$

At the higher level, we consider the problem of updating data split variables, denoted by NUM-VT(b), by solving

$$\begin{array}{lll} \textbf{NUM-VT(b):} & \max_{\phi\succeq 0} & U(\phi) \\ \text{s.t.} & \sum_{a\in\mathcal{A}} \phi_i^a = 1, & \forall i \in \mathcal{N}, \\ & \phi_i^a \ge 0, & \forall i \in \mathcal{N}, \forall a \in \mathcal{A}, \end{array} \tag{6.23}$$

where $U(\phi)$ is the optimal objective value of problem NUM-VT(a) over x, y and t.

6.5.1 Lower-Level Optimization

Since the NUM-VT(a) problem is a strictly concave optimization, we can use the dual decomposition and subgradient algorithm to solve it. Consider its partial Lagrangian with respect to constraints (6.18) and (6.20).

$$\begin{aligned} L(\phi, x, y, t, \lambda, \mu) &= \sum_i U_i(y_i) - \sum_a \sum_i \lambda_i^a (y_i\phi_i^a + \sum_j x_{ji}^a \\ &\qquad - \sum_j x_{ij}^a) - \sum_a \sum_i \sum_j \mu_{ij}^a (x_{ij}^a - F_{ij}t^a) \\ &= \left[\sum_{i\in\mathcal{N}} U_i(y_i) - \sum_{a\in\mathcal{A}} \sum_{i\in\mathcal{N}} \lambda_i^a \phi_i^a y_i\right] + \sum_{a\in\mathcal{A}} \sum_{i\in\mathcal{N}} \sum_{j:(i,j)\in E^a} \mu_{ij}^a F_{ij} t^a \\ &\quad + \left[\sum_{a\in\mathcal{A}} \sum_{i\in\mathcal{N}} \sum_{j:(i,j)\in E^a} (\lambda_i^a - \lambda_j^a - \mu_{ij}^a) x_{ij}^a\right] \end{aligned}$$

Define $D(\phi, \lambda, \mu) = \max_{x,y,t} L(\phi, x, y, t, \lambda, \mu)$ with constraints (6.19), (6.21) and (6.22). By duality, we have the dual problem as

$$U(\phi) = \min_{\lambda \succeq 0, \mu \succeq 0} D(\phi, \lambda, \mu) = \min_{\lambda \succeq 0, \mu \succeq 0} \max_{x,y,t} L(\phi, x, y, t, \lambda, \mu).$$

We observe that dual function $D(\phi, \lambda, \mu)$ can be evaluated separately in terms of data variable y, flow variable x and time variable t. We decompose it into three sets of subproblems, i.e., data control, routing and sojourn time allocation subproblems.

Data control subproblem: The dual function contains $|\mathcal{N}|$ data control subproblems, each for a sensor to determine the amount of data generated for a data gathering tour. Since $y_i = \sum_a y_i \phi_i^a \leq \sum_a \sum_j x_{ij}^a \leq \sum_a \sum_j F_{ij} t^a = \Delta T \sum_j F_{ij}$, y_i has a closed domain. Thus, we set a loose upper bound Y_i for y_i. Given Lagrangian multiplier λ_i^a for the current subgradient iteration, sensor i adjusts y_i to achieve the following optimization goal

$$\max_{0 \leq y_i \leq Y_i} U_i(y_i) - \sum_{a \in \mathcal{A}} \lambda_i^a \phi_i^a y_i$$

Since it is strictly concave, sensor i can obtain the unique optimal y_i as

$$y_i = \begin{cases} 0 & \text{if } U_i'(0) < \sum_a \lambda_i^a \phi_i^a \\ (U_i')^{-1}(\sum_a \lambda_i^a \phi_i^a) & \text{else if } U_i'(Y_i) \leq \sum_a \lambda_i^a \phi_i^a \\ Y_i & \text{otherwise.} \end{cases} \tag{6.24}$$

Routing subproblem: The dual function contains $|\mathcal{N}|$ routing subproblems, each for a sensor to adjust the flow amount over its outgoing links destined to each anchor point. The routing subproblem at sensor i is

$$\begin{aligned} \max_{x \succeq 0} \quad & \sum_{a \in \mathcal{A}} \sum_{j:(i,j) \in E^a} (\lambda_i^a - \lambda_j^a - \mu_{ij}^a) x_{ij}^a \\ \text{s.t.} \quad & \sum_{a \in \mathcal{A}} \sum_{j:(i,j) \in E^a} x_{ij}^a e_{ij} \leq E_i', \end{aligned} \tag{6.25}$$

where $(\lambda_i^a - \lambda_j^a - \mu_{ij}^a)$ can be considered as the link gain for each anchor point a (when $j = s^a$, $\lambda_j^a = 0$). It is clear that in the optimal solution to (6.25), sensor i should spend all its energy budget E_i' on the traffic flows over the link with the largest positive link gain. If we use $\tilde{j}$ and $\tilde{a}$ to represent such a preferred outgoing

neighbor and a destined anchor point, respectively, $(\tilde{j}, \tilde{a})_i = \arg\max_{j,a}[(\lambda_i^a - \lambda_j^a - \mu_{ij}^a)]^+$. Mathematically speaking, the reason why we employ such an assignment strategy is because that the optimization of sensor i over $\{x_{ij}^a\}$ is a linear program and we can always choose an extreme point solution such that

$$x_{ij}^a = \begin{cases} \frac{E_i'}{e_{ij}} & \text{if } (j,a) = (\tilde{j}, \tilde{a})_i \& (\lambda_i^a - \lambda_j^a - \mu_{ij}^a) > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{6.26}$$

Sojourn time allocation subproblem: The SenCar is responsible for allocating the sojourn time for each anchor point to satisfy the optimization goal

$$\max_{t \succeq 0} \sum_{a \in \mathcal{A}} \sum_{i \in \mathcal{N}} \sum_{j:(i,j) \in E^a} \mu_{ij}^a F_{ij} t^a \quad \text{s.t.} \sum_{a \in \mathcal{A}} t^a \leq \Delta T.$$

Similarly, since this is a linear program and each t^a has a non-negative coefficient in the objective function, we can simply assign ΔT to the t^a with the maximum coefficient.

$$t^a = \begin{cases} \Delta T & \text{if } a = \arg\max_a \sum_i \sum_j \mu_{ij}^a F_{ij} \\ 0 & \text{otherwise.} \end{cases} \tag{6.27}$$

Note that for the SenCar to determine the sojourn time, the values of Lagrangian multipliers μ_{ij}^a need to be routed to the SenCar in each subgradient iteration. To dampen the communication overhead, we can alternatively let each sensor determine the sojourn time at each anchor point. In particular, since each t^a is a global variable for sensors, we need to introduce local variable t_i^a for each sensor and impose additional constraints $t_i^a = t_j^a, \forall i \in \mathcal{N}, \forall j : (i,j) \in E^a, \forall a \in \mathcal{A}$ to the NUM-VT problem to enforce all the t_i^a's to be equal for each anchor point a. Then, we can relax these constraints in the Lagrangian for dual decomposition and solve the subproblem on t_i^a by sensor i. Such an approach only requires each sensor to communicate with direct neighbors on exchanging the corresponding Lagrangian multipliers. However, in our practice, we find that this approach results in slower convergence than the approach with $\{t^a\}$ determined by the SenCar. Thus, this is a tradeoff between the communication overhead and convergence speed [50]. In practice, an application can choose to let each sensor or the Sencar determine the values.

Lagrangian multiplier update: Sensor i updates its Lagrangian multipliers λ_i^a

and μ_{ij}^a according to

$$\begin{aligned}\lambda_i^a(k+1) &= [\lambda_i^a(k) + \theta(k)(y_i(k)\phi_i^a + \sum_j x_{ji}^a(k) - \sum_j x_{ij}^a(k))]^+, \\ \mu_{ij}^a(k+1) &= [\mu_{ij}^a(k) + \theta(k)(x_{ij}^a(k) - F_{ij}t^a(k))]^+,\end{aligned} \tag{6.28}$$

where k represents the index of subgradient iterations, and $\theta(k)$ is the diminishing stepsize discussed in the NUM-FT problem. Sensor i will send updated Lagrangian multipliers to its direct neighbors and route $\mu_{ij}^a(k+1)$ to the SenCar if necessary.

Recovery of primal solutions: Since the subproblems of routing and sojourn time allocation are linear, we need to recover the optimal primal values for variables x_{ij}^a and t^a. When data split variables ϕ_i^a reach their optimum $(\phi_i^a)^*$ in the higher-level optimization, during the subgradient iterations in the lower level, we construct the primal feasible sequences $\{\hat{x}_{ij}^a(k)\}$ and $\{\hat{t}_a(k)\}$ by using the method in [100], i.e., $\hat{x}_{ij}^a(k) = \frac{1}{k}\sum_{h=1}^k x_{ij}^a(h)$ and $\hat{t}^a(k) = \frac{1}{k}\sum_{h=1}^k t^a(h)$. In this way, the final optimal $x_{ij}^{a\,*}$ and t^{a*} can be obtained from the values sequences $\{\hat{x}_{ij}^a(k)\}$ and $\{\hat{t}^a(k)\}$ converge to.

6.5.2 Higher-Level Optimization

The above algorithm for the lower-level optimization works under the assumption that each ϕ_i^a is a constant. At the higher level, we now show how sensor i adjusts ϕ_i^a to achieve the optimum of the NUM-VT problem.

Note that $U(\phi)$ is the optimal objective value of the lower-level optimization. By using λ^* and μ^* to represent the Lagrangian multiplier values that minimize $D(\phi, \lambda, \mu)$ for a given ϕ, $U(\phi)$ is given by

$$\begin{aligned}&U(\phi) = \min_{\lambda\succeq 0, \mu\succeq 0} D(\phi, \lambda, \mu) \\ &= \sum_i U_i(y_i(\lambda^*)) - \sum_a\sum_i \lambda_i^{a*}(y_i(\lambda^*)\phi_i^a + \sum_j x_{ji}^a(\lambda^*, \mu^*) \\ &\quad -\sum_j x_{ij}^a(\lambda^*, \mu^*)) - \sum_a\sum_i\sum_j \mu_{ij}^{a\,*}(x_{ij}^a(\lambda^*, \mu^*) - F_{ij}t^a(\mu^*)).\end{aligned} \tag{6.29}$$

Then, the marginal utility for ϕ_i^a is $\frac{\partial U(\phi)}{\partial \phi_i^a} = -\lambda_i^{a*} y_i(\lambda^*)$, which reflects the gain of the data sent from sensor i to anchor point a. Intuitively, to maximize network utility $U(\phi)$, sensor i should always shift some of its data destined to other anchor points to the one with the highest marginal utility until ϕ_i reaches an equilibrium.

Table 6.5: Distributed algorithm for the NUM-VT problem.

For each sensor $i \in \mathcal{N}$ **do**
  Initialize data split variables $\phi_i^a(0)$ for all $a \in \mathcal{A}$ satisfying $\sum_a \phi_i^a(0) = 1$;
  Repeat
    Initialize Lagrangian multipliers $\lambda_i^a(0)$ and $\mu_{ij}^a(0)$, for all $j : (i,j) \in E^a$ and $a \in \mathcal{A}$, to non-negative values;
    Repeat: for all $j : (i,j) \in E^a$ and $a \in \mathcal{A}$
      Compute $y_i(k)$ by Eq. (6.24);
      Compute $x_{ij}^a(k)$ by Eq. (6.26);
      SenCar simultaneously computes $t^a(k)$ by Eq. (6.27);
      Update Lagrangian multipliers $\lambda_i^a(k+1)$ and $\mu_{ij}^a(k+1)$ according to Eq. (6.28);
      Send updated Lagrangian multipliers to its neighbors and route $\mu_{ij}^a(k+1)$ to the SenCar;
      If $\{\phi(n)\}$ reaches ϕ^* in the outer-loop iterations
        Compute primal feasible $\hat{x}_{ij}^a(k)$ by $\hat{x}_{ij}^a(k) = \frac{1}{k}\sum_{h=1}^k x_{ij}^a(h)$, and SenCar simultaneously computes primal feasible $\hat{t}^a(k)$ by $\hat{t}^a(k) = \frac{1}{k}\sum_{h=1}^k t^a(h)$;
      end If
    Until $\{\lambda(k)\}$ converges to λ^* and $\{y(k)\}$ converges to y^*;
    Adjust data split variables $\phi_i^a(n+1)$ ($\forall a \in \mathcal{A}$) by Eq. (6.30);
  Until reach the equilibrium, i.e., $\{\phi(n)\}$ converges to ϕ^*;
end For

Let $\phi^* = \{\phi_i^{a*}\}$ be the optimal data split matrix. We can characterize the optimal solution ϕ^* to the problem in (6.23) by the following optimality condition.

For each sensor i, we have

$$\phi_i^{a*} > 0 \Rightarrow \frac{\partial U(\phi^*)}{\partial \phi_i^{a'}} \leq \frac{\partial U(\phi^*)}{\partial \phi_i^a}, \text{ for all } a' \in \mathcal{A}.$$

That is, sensor i sends its data to the SenCar only at those anchor points that have the maximum marginal utility. This is similar to the wardrop equilibrium [99].

For sensor i, let $\tilde{a}_i$ be the anchor point with the maximum marginal utility, i.e., $\tilde{a}_i = \arg\max_{a\in\mathcal{A}} \frac{\partial U(\phi)}{\partial \phi_i^a}$. Sensor i updates ϕ_i^a according to the following principles.

$$\phi_i^a(n+1) = \phi_i^a(n) + \delta_i^a(n), \text{ with} \tag{6.30}$$

$$\delta_i^a(n) = \begin{cases} -\min\left\{\phi_i^a(n), \frac{\kappa(n)}{y_i}\left(\frac{\partial U(\phi)}{\partial \phi_i^{\tilde{a}_i}}(n) - \frac{\partial U(\phi)}{\partial \phi_i^a}(n)\right)\right\} \\ \qquad\qquad \text{if } a \neq \tilde{a}_i \\ -\sum_{a\neq\tilde{a}_i, a\in\mathcal{A}} \delta_i^a(n) \qquad \text{if } a = \tilde{a}_i, \end{cases}$$

where n stands for the iteration index for the higher-level optimization and $\kappa(n)$ is

a small positive scalar stepsize.

By using a similar approach to that in [106], we can show that such updating algorithm on ϕ guarantees the convergence to the optimal solution of the NUM-VT problem. It is straightforward to verify that the updating algorithm in (6.30) satisfies that

$$\sum_a \delta_i^a(n) = 0, \text{ and } \sum_a \frac{\partial U(\phi)}{\partial \phi_i^a}(n)\delta_i^a(n) \geq 0, \forall i \in \mathcal{N}.$$

And for sensor i, $\sum_a \frac{\partial U(\phi)}{\partial \phi_i^a}(n)\delta_i^a(n) = 0$ only if $\delta_i^a(n) = 0$, which requires that $\phi_i^a(n)(\frac{\partial U(\phi)}{\partial \phi_i^{\bar{a}_i}} - \frac{\partial U(\phi)}{\partial \phi_i^a}) = 0$ for all $a \in \mathcal{A}$. If we consider the continuous version, we have that for each sensor,

$$\sum_a \dot{\phi}_i^a = 0 \text{ and } \sum_a \frac{\partial U(\phi)}{\partial \phi_i^a}\dot{\phi}_i^a \geq 0. \tag{6.31}$$

The updating algorithm in (6.30) can be considered as a specific discrete time implementation of (6.31). Since $U(\phi) = \min_{\lambda,\mu} D(\phi, \lambda, \mu)$ and $D(\phi, \lambda, \mu)$ are the non-smooth functions with respect to λ and μ based on the expression of (6.29), the differential of $U(\phi)$ can be written as

$$\begin{aligned} dU(\phi) = &\left(\lim_{h\to 0^+} \frac{\partial D(\phi,\lambda^*+hd\lambda,\mu^*)}{\partial \lambda}\right) d\lambda + \\ &\left(\lim_{h\to 0^+} \frac{\partial D(\phi,\lambda^*,\mu^*+hd\mu)}{\partial \mu}\right) d\mu + \frac{\partial D(\phi,\lambda^*,\mu^*)}{\partial \phi} d\phi, \end{aligned} \tag{6.32}$$

where $(\lambda^*, \mu^*) = \arg\min_{\lambda,\mu} D(\phi, \lambda, \mu)$. As λ^* and μ^* minimize $D(\phi, \lambda, \mu)$ for a given ϕ, $\lim_{h\to 0^+} \frac{\partial D(\phi,\lambda^*+hd\lambda,\mu^*)}{\partial \lambda}$ and $\lim_{h\to 0^+} \frac{\partial D(\phi,\lambda^*,\mu^*+hd\mu)}{\partial \mu}$ cannot be in a deceasing direction, i.e., the first two terms in (6.32) are non-negative. Hence, we have

$$dU(\phi) \geq \frac{\partial D(\phi,\lambda^*,\mu^*)}{\partial \phi} d\phi = \sum_i \sum_a \frac{\partial U(\phi)}{\partial \phi_i^a} d\phi_i^a. \tag{6.33}$$

Accordingly, by (6.31) and (6.33), $\dot{U}(\phi) \geq \sum_i \sum_a \frac{\partial U(\phi)}{\partial \phi_i^a}\dot{\phi}_i^a \geq 0$, which means that the updating on ϕ by (6.30) always improves the overall network utility. And for each sensor, when all its positive marginal utility approaches the same value, ϕ would reach an equilibrium ϕ^* such that $\dot{U}(\phi^*) = 0$.

Finally, we summarize the distributed algorithm for the NUM-VT problem in

Table 6.5.

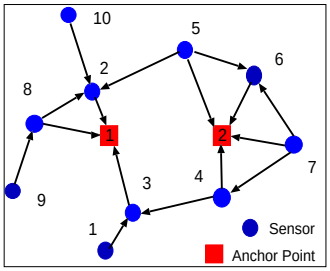
6.6 Numerical Results

In this section, we provide some numerical results to demonstrate the usage and efficiency of the proposed distributed algorithms and compare their performance with other data gathering strategies. We adopt the data utility function as $U_i = w_i \log(1 + \sum_a q_i^a t^a)$ or equivalently $U_i = w_i \log(1 + y_i)$, indicating that the data from a sensor with a larger weight w_i would have more impact on the overall performance. In the following simulations, since we focus on the convergence property of the algorithms and their relative performance, we simply use the dimensionless quantities in the figures for demonstration purpose only. It is worth pointing out that the solution exploration procedure for each distributed algorithm only needs to be executed when the energy budget is updated or the topology of the network changes, thus does not need to be frequently repeated by sensors. Finally, note that if a centralized execution is preferred, the proposed algorithms can also be executed by a central controller in a multi-threading fashion. For example, we can use the Sencar to collect the necessary information on the entire network, execute the proposed algorithm and distribute the solution to sensors.

6.6.1 Convergence

We first examine the convergence property of the algorithms for the NUM-FT and NUM-VT problems. For illustration purpose, we use a small generic network in Fig. 6.3 to show how the algorithms work. In fact, due to the distributed nature, the algorithms are readily applicable to large scale networks. In Fig. 6.3, there are ten sensors and two anchor points distributed over the sensing field. The links in the network are assumed to be directed as indicated by arrows in the figure and all have equal capacity. The energy consumption for transmitting a unit flow over link (i, j), e_{ij}, is proportional to the square of d_{ij}, where d_{ij} is the physical distance between sensors i and j. For clarity, we list all the parameter settings in Fig. 6.3(b).

Fig. 6.4 shows the evolution of the recovered flow rate $\hat{f}_{ij}^a$ and the Lagrangian multiplier λ_i^a versus the number of iterations in the algorithm for the NUM-FT problem. We set $t^1 = t^2 = 25$ as an example for the network with given fixed

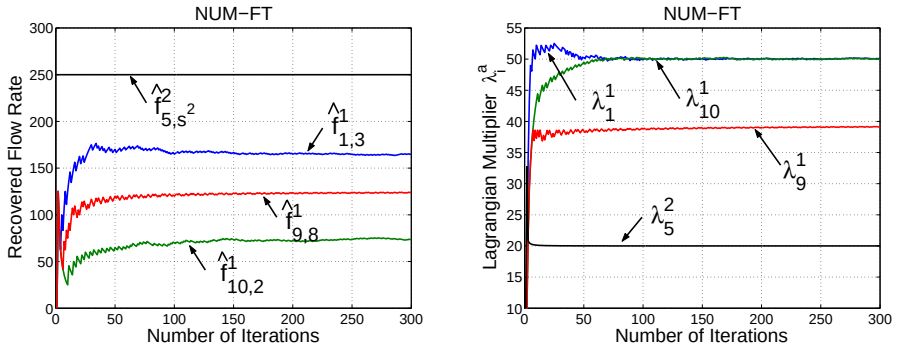


(a) Network Configuration.

Notation	Value	Notation	Value
$E_i' \ \forall i$	1.25×10^4	ΔT	50
e_{ij} $\forall (i,j)$	$0.007 d_{ij}^2$	$w = \{w_i\}$	$10^3 \times \{10, 10, 5$ $5, 5, 5, 5, 5, 5\}$
F_{ij} $\forall (i,j)$	250	$\theta(k)$ (NUM-FT)	$\frac{1}{1+10k}$
$Q_i \ \forall i$	750	$\theta(k)$ (NUM-VT)	$\frac{1}{1+20k}$
$Y_i \ \forall i$	2×10^5	$\kappa(n)$ (NUM-VT)	0.005

(b) Parameter Settings.

Figure 6.3: An example network with ten sensors and two anchor points.



(a) (b)

Figure 6.4: Numerical results of the algorithm for the NUM-FT problem: (a) Evolution of recovered flow rate $\hat{f}_{ij}^a$ vs. subgradient iterations; (b) Evolution of Lagrangian multiplier λ_i^a vs. subgradient iterations.

sojourn time for each anchor point. In Fig. 6.4(a), we examine the flow rates on some selected links, links $(1, 3)$, $(9, 8)$ and $(10, 2)$ destined to anchor point 1 and link $(5, s^2)$ destined to anchor point 2. It can be seen that the recovered flow rates are well within 5% of their optimal values after only 100 iterations. This observation is also applicable to the evolution of the Lagrangian multipliers as shown in Fig. 6.4(b). Moreover, we also notice that as sensor 1 has a larger utility weight, the flow rate of its outgoing link destined to anchor point 1 is higher than those of sensors 9 and 10 to achieve higher network utility. Since sensor 5 has a direct path to the SenCar at anchor point 2, the optimal flow rate over link $(5, s^2)$ is able to reach the capacity bound.

We now consider the same network for the NUM-VT problem. The performance of the algorithm is shown in Fig. 6.5. For each sensor, we set the data split variables for different anchor points with equal initial values, i.e., $\phi_i^1 = \phi_i^2 = 0.5$. We use a constant stepsize for the higher-level iterations, i.e., $\kappa(n) = 0.005$. In the simulation, the lower-level optimization runs 5000 iterations before each run of the higher-level updating on the data split variables. To dampen the number of lower-

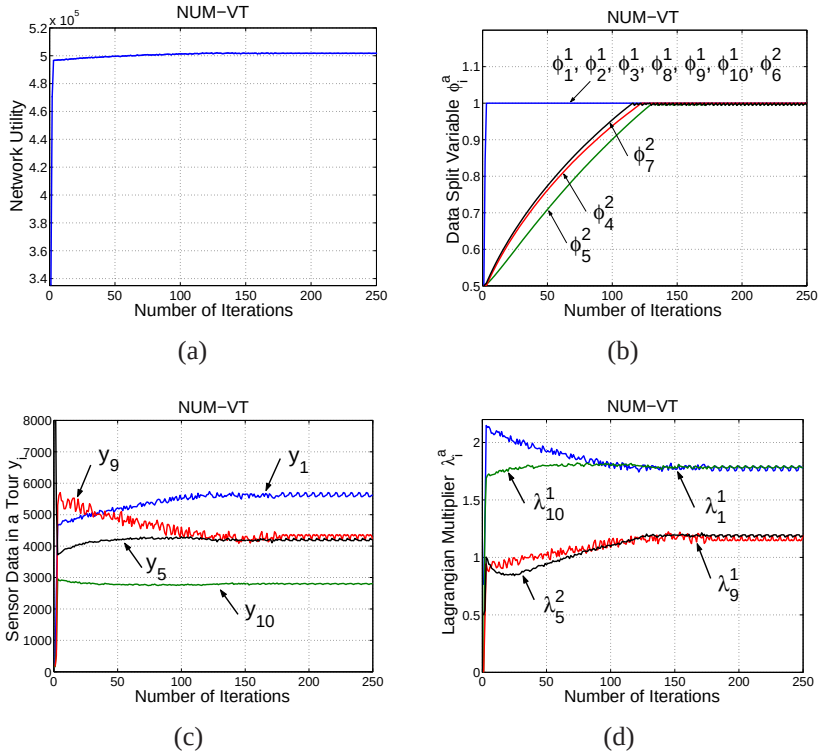


Figure 6.5: Numerical results of the algorithm for the NUM-VT problem: (a) Evolution of network utility vs. higher-level iterations; (b) Evolution of data split variable ϕ_i^a vs. higher-level iterations; (c) Evolution of total amount of data y_i gathered from sensor i in a data gathering tour vs. higher-level iterations; (d) Evolution of Lagrangian multiplier λ_i^a vs. higher-level iterations.

level iterations, we can set the initial values of Lagrangian multipliers λ_i^a and μ_{ij}^a by their final values in the previous run of higher-level optimization. Comparable performance is observed when the number of lower-level iterations is as low as several hundreds. In Fig. 6.5(a), we notice that the network utility first increases sharply during early iterations and then slowly increases until it reaches the optimum after about 100 iterations. It effectively validates that the updating on ϕ always improves the overall network utility and guarantees its convergence to the optimum. In Fig. 6.5(b), we plot some ϕ_i^a's versus the number of iterations, which clearly shows that in the optimal solution, sensors 1-3 and 8-10 would only send data to the SenCar when it arrives at anchor point 1 and sensors 5-7 alternatively choose to upload data to the SenCar when it is located at anchor point 2. Fig. 6.5(c) and (d) depict the evolution of the total amount of data y_i generated by sensor i in a data gathering tour and Lagrangian multiplier λ_i^a versus the number of iterations, respectively. We can obtain similar observations to those for the NUM-FT problem. As sensor 1 has a larger utility weight compared to other sensors, in a data gathering tour, the SenCar preferentially gathers more data from it. In contrast, the SenCar gathers the least amount of data from sensor 10 since it has a small utility weight and its data have to be relayed by sensor 2 to reach the SenCar at anchor point 1. As sensor 2 has a larger weight than that of sensor 10, sensor 10 would refrain from generating more data to avoid congesting the common link with sensor 2. Furthermore, it is clear that, to gather more data from the sensors with a large utility weight, the SenCar would stay longer at anchor point 1 than at anchor point 2. As the results of our algorithm, the optimal sojourn times for the two anchor points are $t^1 = 33.37$ and $t^2 = 16.63$.

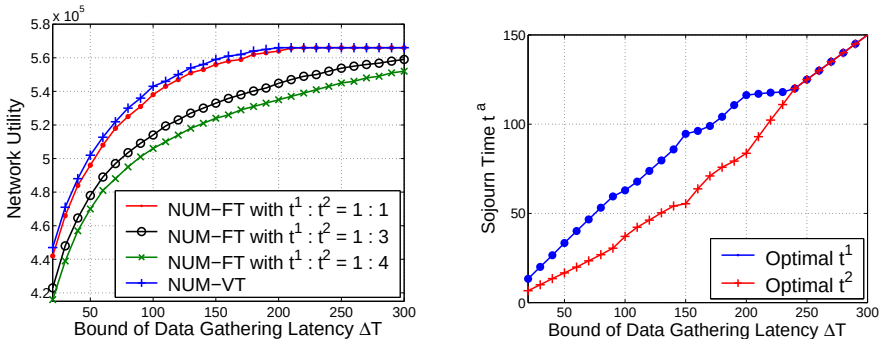
6.6.2 Performance Comparison between NUM-FT and NUM-VT

In this subsection, we compare the performance of the algorithms for the NUM-FT and NUM-VT problems. We still use the same network and parameter settings as in Fig. 6.3. We consider three instances of the NUM-FT problem where $t^1 : t^2$ is fixed to $1 : 1$, $1 : 3$ and $1 : 4$, respectively. In contrast, the algorithm for the NUM-VT problem dynamically pursues the optimal sojourn time allocation for each anchor point. Fig. 6.6(a) plots the network utility as the function of ΔT. From the results,

we can draw some observations. First, as ΔT increases, the network utility of all the cases under investigation increases until it reaches the maximum value. This is intuitive since the longer the SenCar stays at each anchor point, the more data it would gather. Once ΔT becomes sufficiently large, some sensors would ultimately deplete their energy budget for the current data gathering tour. As a result, the network utility stays unchanged as no more data can be extracted by the SenCar. Second, for a given ΔT, we find that the network utility achieved by NUM-VT is always larger than that of NUM-FT. This is expected since the utility region for the cases with fixed sojourn time is a subset of the utility region with variable sojourn time. Such superiority is especially notable for the case with a small ΔT, which implies that the proper sojourn time allocation is particularly critical to the efficiency of data gathering with low latency requirement. Furthermore, we plot the optimal sojourn time allocation obtained for the NUM-VT problem in Fig. 6.6(b). From the figure, we notice that the optimal value of t^1 is larger than that of t^2 in the initial phase, then the difference between them gradually becomes smaller with the increase of ΔT, and finally t^1 and t^2 are set to the same value. The reasons for such tendency can be explained as follows. When ΔT is small, the sensors are mostly energy-rich and the amount of gathered data from each sensor primarily depends on ΔT. However, as sensors 1 and 2 have larger utility weights and would route data only to anchor point 1, the SenCar is apt to stay longer at anchor point 1 than anchor point 2, i.e., $t^1 > t^2$. However, as ΔT gradually increases, some sensors around anchor point 1 become energy-constrained. Then the SenCar would shift some sojourn time from anchor point 1 to anchor point 2 so as to gather more data. And finally, when ΔT becomes large enough, no more data can be obtained regardless where the SenCar stays for more sojourn time. In our simulation, when $t^1 > 120$ and $t^2 > 110$, the gathered data from each sensor would no longer change. Thus, when $\Delta T \geq 240$, there are multiple options for the optimal sojourn time allocation. One option is to equally divide ΔT for t^1 and t^2, which is as shown in the figure.

6.6.3 Performance Comparison with Other Strategies

In this subsection, we investigate the network utility achieved by NUM-FT and NUM-VT in large sensor networks, and compare the results with other two strategies. The first strategy used for comparison employs optimal data rates and ran-



(a) (b)

Figure 6.6: Comparison between NUM-FT and NUM-VT: (a) Network utility vs. ΔT. (b) Optimal sojourn time allocation for the NUM-VT problem vs. ΔT.

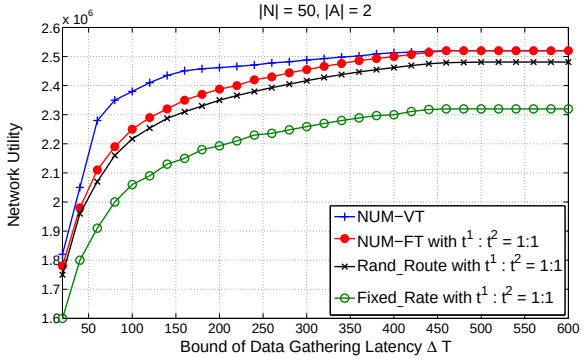


Figure 6.7: Performance comparison among NUM-VT, NUM-FT, Rand_Route, and Fixed_Rate.

dom routing, called Rand_Route, in which each sensor randomly chooses a routing path and transmits data to the SenCar at one of reachable anchor points. Another strategy for comparason is the fixed data rate and shortest path routing, denoted by Fixed_Rate, where all sensors have a homogeneous data rate and each of them preferentially chooses a shortest path to an anchor point for data transmissions. For fair comparison, we assume that the data rate each sensor uses in Fixed_Rate is the maximum possible rate that can avoid the traffic congestion in the network.

The parameters used in the simulations are set as follows. A total of 50 sensors are randomly scattered in a 200×200 field, and two anchor points are located at $(66.7, 100)$ and $(133.3, 100)$, respectively. The transmission range of each sensor

is set to 40. One-fifth of sensors have larger utility weight with the value of 10^4 and all others have smaller utility weight with the value of 5×10^3. There are 100 directed links in the network with equal link capacity of 250. These links are randomly distributed among the neighboring nodes and in the meanwhile they ensure that each sensor can reach at least one anchor point. We assume that each sensor holds an energy budget of 3×10^4 and $e_{ij} = 0.002 d_{ij}^2$. Moreover, in NUM-FT, Rand_Route, and Fixed_Rate, the sojourn time at two anchor points is fixed and set to the same value, i.e., $t^1 : t^2 = 1 : 1$.

Fig. 6.7 plots the network utility of NUM-VT, NUM-FT, Rand_Route, and Fixed_Rate when ΔT varies from 20 to 600. Considering the topology randomness, each performance point shown in the figure is the average of 20 simulations. The results demonstrate that NUM-VT always outperforms other strategies. For example, when $\Delta T = 100$, NUM-VT achieves 10% and 20% higher network utility compared to Rand_Route and Fixed_Rate. Such superiority of NUM-VT over others is ascribed to the joint design on rate control, routing, and sojourn time allocation such that system-wide optimum can be achieved. In contrast, Rand_Route may result in "hot" common links shared on many routing paths and this would greatly limit the data rates of the sensors on the paths due to the link capacity constraint. Similarly, Fixed_Rate typically causes the sensors to use relatively low data rate to ensure that all the links in the network are not congested. Moreover, as all sensors have a homogeneous data rate in Fixed_Rate, it prevents the sensors with larger utility weight from sending more data to the SenCar such that the network utility is still far below the fully extracted value. It is also observed from the figure that the network utility of all these strategies would reach their respective stable status when ΔT becomes sufficiently large. This is because that when the sensors close to the anchor points deplete their energy, the SenCar cannot successfully gather more data from the sensors far away from them such that network utility cannot be further increased.

6.7 Conclusions

In this chapter, we have considered finding optimal strategies for anchor based mobile data gathering in WSNs. We formalized this problem as two convex optimization problems, i.e., NUM-FT and NUM-VT, which consider the cases that the

SenCar spends fixed and variable sojourn time at each anchor point, respectively. Both NUM-FT and NUM-VT problems aim to maximize the overall network utility while guaranteeing the given network lifetime and data gathering latency. The NUM-FT problem essentially involves the joint design of rate control and optimal routing, and the NUM-VT problem is the integration of data control, routing and sojourn time allocation problems. Based on their decomposable nature, we correspondingly proposed two efficient distributed algorithms to solve them. Finally, we provided extensive numerical results to validate the efficiency of the proposed algorithms and complement our theoretical analysis.

Chapter 7

Joint Mobile Energy Replenishment and Data Gathering in Wireless Rechargeable Sensor Networks

In previous chapters, we focus our study on the conventional sensor networks, where sensor batteries can not be recharged or replaced after the initial deployment. However, recent advance in sensor technologies has made it possible for sensors to obtain renewable energy supply to sustain their operations. In this chapter, we consider applying wireless energy transmission to sensor energy replenishment and extend our study on mobile data gathering to such rechargeable sensor networks. Wireless energy transmission is generally referred to as the transfer of electrical energy from a power source to an electrical load without interconnecting wires, which is carried out using resonant inductive coupling. We consider using the SenCar to play as such power source to recharge the sensors in need of energy. In other words, SenCar is employed to serve not only as a data collector that roams over the field to gather data via short-range communication but also as an energy transporter that charges static sensors on its migration tour via wireless energy transmissions. Taking advantages of the SenCar's controlled mobility, we propose a joint design of energy replenishment and data gathering, which aims to provide steady and high recharging rates, and achieve high-utility data gathering simultaneously. In particular, we give a two-step approach to implement the joint design. In the first step, the locations of a subset of sensors are periodically selected as *anchor points*, where the

SenCar will sequentially visit to charge the located sensors at these locations and gather data from nearby sensors in a multi-hop fashion. In order to achieve a desirable balance between the energy replenishment amount and data gathering latency, we provide a selection algorithm to search for a maximum number of anchor points where sensors hold the least battery energy, and meanwhile by visiting them the tour length of the SenCar is no more than a threshold. In the second step, we consider data gathering performance when the SenCar migrates among these anchor points. We formulate the problem into a network utility maximization problem and propose a distributed algorithm to adjust data rates, link scheduling and flow routing so as to adapt to the up-to-date energy replenishing status of sensors. The effectiveness of our approach is validated by extensive numerical results. When compared with solar harvesting networks, our solution can improve the network utility by 48% on the average.

The remainder of this chapter is organized as follows. Section 7.1 introduces the motivation and contribution of this research work. Section 7.2 discusses the related work. Section 7.3 provides the design overview. Sections 7.4 gives the design details by presenting a two-step approach. Section 7.5 gives the performance evaluation results and finally Section 7.6 concludes the chapter.

7.1 Introduction

Recent studies have shown that energy harvesting wireless sensor networks have the potential to provide perpetual network operations by capturing renewable energy from the external environment. A variety of ambient energy, such as mechanical, thermal, photovoltaic, and electromagnetic energy can be converted into electrical energy to drive sensors or recharge sensor batteries, such that prolonged network lifetime or perpetual operations can be achieved [72][73]. However, as all these energy sources are from external environment and their spatial-temporal profiles exhibit great variations, the strength of harvested energy is typically low [69], and especially sensitive to the environment dynamics. For example, in a solar harvesting system, the output power of a sensor is determined by solar radiation arrives at the equipped solar panel, which drastically varies with time and weather. Statistics has shown that the difference can be up to three orders of magnitude among the available solar power in shadowy, cloudy and sunny environments [79]. As there

is generally lack of priori knowledge of the energy profile, such dynamics imposes much difficulty on the design of protocols that must keep sensors from running out of energy. This is, however, very critical for many applications, especially environmental monitoring applications where the main task is to periodically collect data from all sensors. In case that some sensors deplete their energy and cannot get recharged in time, the network would ultimately become fragmented and the data from some parts of the sensing field can no longer be extracted.

In order to provide steady and high recharging rate for the power supplies of sensors, and meanwhile effectively alleviate energy expenditure on data gathering, in this work, we alternatively propose a joint design of energy replenishment and data gathering by exploiting mobility, which is referred to as J-MERDG. In particular, a multi-functional SenCar, is employed, which is equipped with a powerful transceiver and high capacity battery. The SenCar will periodically choose a subset of sensors to visit. While migrating among these sensors, it delivers energy to the visited sensors by utilizing wireless energy transmissions and meanwhile it collects data from nearby sensors via short-range multi-hop communication. This way, the SenCar, serving as both an energy transporter and a mobile data collector, performs the tasks of energy replenishment and data gathering simultaneously. In contrast to the conventional energy harvesting networks, the mobility brings us many benefits. First, since sensors receive energy supplement directly from the SenCar, the replenishment will no longer suffer from environmental variations. Second, as long as the SenCar moves close enough to sensors, high charging efficiency can be achieved to ensure high-rate data services. Third, as the SenCar takes the responsibility of energy delivery, it is commercially appealing that no complex energy harvesting devices are needed at each sensor, which significantly reduces the cost of sensors. Finally, by exploiting controlled mobility, the SenCar can efficiently perform energy delivery and data gathering simultaneously. This is extremely desirable as such combination makes double contribution to the energy management of the network. On one hand, the SenCar infuses steady and abundant renewable energy into the network almost at no additional cost. On the other hand, mobility alleviates the routing burden at sensors for data uploading so that great energy can be saved to further leverage the refilled energy.

The objective of our work is to design an adaptive solution that jointly selects the sensors to be charged and finds the optimal data gathering strategies, such that

network utility can be maximized while maintaining perpetual operations of the network. To that end, we propose a two-step approach for the joint design. In the first step, we determine the mobility pattern of the SenCar for each time period, i.e., where the SenCar will move to charge the located sensors and gather the data from the neighborhood. For convenience, we still refer to the locations that the SenCar visits for energy delivery and data gathering as *anchor points*. In the second step, we study the strategies on how to achieve optimal data gathering performance when the SenCar migrates among different anchor points, considering the up-to-date energy replenishing status of sensors. We formulate this problem as an optimization problem and adjust data rates, link scheduling and flow routing to achieve maximum network utility.

The main contributions of our work can be summarized as follows.

- We propose a joint design of energy replenishment and data gathering (J-MERDG) by exploiting mobility. To the best of our knowledge, this is the first work that explores such joint design and systematically provides solutions to optimize its performance.

- We develop an algorithm for the SenCar to determine the anchor points in each time period, which achieves a desirable balance between the energy replenishing range and data gathering latency.

- We build a flow-level network utility maximization model to characterize the data gathering performance when the SenCar moves over different anchor points. We propose a proximal approximation based algorithm to obtain the system-wide optimum by adjusting data rates, link scheduling and flow routing in a distributed manner.

- We provide extensive numerical results to validate the effectiveness of J-MERDG, which not only guarantees perpetual operations of the network but also significantly outperforms solar harvesting system by 48% in network utility.

7.2 Related Work

As we have already discussed the mobile data gathering schemes in previous chapters, in this section, we mainly review some related work on energy replenishment in wireless networks.

Kar, et al. [74] considered a network with redundantly deployed rechargeable sensors, and addressed the problem of how sensors should be activated dynamically so as to maximize a global coverage metric. They proposed a threshold activation policy and demonstrated its performance for the cases that the coverage areas of sensors are completely or partially overlapped. Lin, et al. [75] developed a model to characterize the performance of multihop radio networks in the presence of energy replenishment and designed an energy-aware routing algorithm that is asymptotically optimal with respect to the network size. Liu, et al. [76] studied the resource allocation problem for energy-harvesting sensors. They first explored the optimal sampling rates based on the average (long term) energy replenishment rate and then designed a local algorithm for each sensor to adjust the rate according to the instantaneous battery state in order to cater to the recharging fluctuations. Sharma, et al. [77] presented a model for a single energy harvesting sensor node and identified two energy management policies for it. One policy is throughput-optimal, which ensures that the data queue stays stable for the highest possible data rate, and the other policy aims at minimizing the mean delay of the data queue. Vigorito, et al. [78] considered the variability of the harvested environmental energy and designed an adaptive duty-cycling mechanism that achieves energy neutral operation, performance maximization and duty cycle stability. Rahimi, et al. [79] studied the feasibility of exploiting mobility to extend network lifetime, in which a small number of network nodes are autonomously mobile, allowing them to move in search of energy from the environment, recharge and deliver energy to immobile, energy-depleted nodes. All the above works focus on energy harvesting networks and try to provide adaptive mechanisms to conquer the environment variations. Besides [79], other works did not consider exploiting mobility.

Based on these studies of exiting energy harvesting networks and aforementioned mobile data gathering schemes [22]-[59] in previous chapters, we find that energy replenishment and data gathering are always separately designed. None of existing work makes these two tasks in couple to balance their performance.

This observation motivates us to provide novel scheme that jointly considers these two important issues in WSNs. By taking advantage of controlled mobility, our work explores the data gathering performance gain when recharging is possible. It computes the migration tour of the SenCar based on the joint consideration of recharging demand and data gathering performance, and also adapts adjustable system parameters to the up-to-date energy replenishing status to optimize the data gathering scheme.

7.3 Design Overview of J-MERDG

In this section, we provide an overview on J-MERDG. The timing structure and the architecture of this joint design are illustrated in Fig. 7.1 and Fig. 7.2, respectively.

As each sensor has different energy statuses at different times, it is required for the SenCar to properly arrange which sensor gets recharged at what time. Due to such time-varying nature of the energy replenishment demand, to facilitate our study, we divide the time into fixed time intervals of length T. At the beginning of each time interval, the SenCar determines which sensors to be charged in this interval. We assume that the possible candidate locations for the SenCar to visit are the locations of all sensors, such that the SenCar can move sufficiently close to charge sensors with high efficiency. Based on some specified criteria (to be discussed in a subsequent section), the locations of a subset of sensors, i.e., anchor points, are selected. In a time interval, the sensors located at the anchor points would be recharged. As shown in Fig. 7.1, in each time interval, the SenCar will migrate among the anchor points back and forth. We assume that there are a total of q migration tours in each time interval. The same data gathering strategy is used for these q migration tours. Along each tour, the SenCar would sojourn at each anchor point to gather data from nearby sensors via multi-hop communication. Without loss of generality, we assume that the SenCar sojourns for the same amount of time at every anchor point in a tour. However, it should be pointed out that as anchor points in each time interval vary in quantity and positions, the sojourn time may be different from one time interval to another. During the last tour in a time interval, each sensor will report its up-to-date battery status to the SenCar. This information is transmitted piggyback with the data to the SenCar, which will be used for the anchor point selection at the beginning of next time interval.

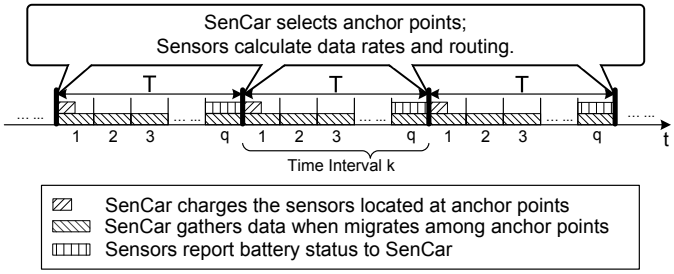


Figure 7.1: Timing of joint mobile energy replenishment and data gathering (J-MERDG).

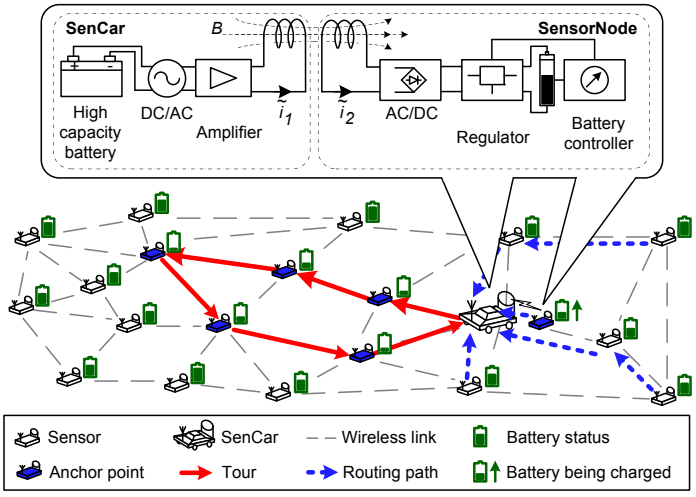


Figure 7.2: Architecture of joint mobile energy replenishment and data gathering (J-MERDG).

While the SenCar arrives at an anchor point, it will quickly charge the sensor located there. The recharging structure is depicted in Fig. 7.2. The SenCar, as the energy transmitter, is equipped with a high-capacity rechargeable battery, a DC/AC converter and a resonant coil. For energy delivery, an oscillating magnetic field is first induced around the transmitter coil on the SenCar. The sensor node, mounted with a receiver coil, is then tuned to resonate at exactly the same frequency, and use the AC/DC converter to generate DC current to recharge its battery. Such a structure is feasible and of high efficiency, which is supported by recent breakthroughs in two areas. The first area is the technology of highly-efficient wireless energy transmission. Wireless energy transmission has already been used to recharge small appliances, such as electric toothbrushes. Recent progresses also show application prospect of non-radiative energy transmission over midrange [1]. The work in [65] and [66] has shown that through strongly coupled magnetic resonances, the efficiency of transferring 60 watts of power over a distance in excess of 2 meters is as high as 40%. Intel also demonstrated that it is possible to improve transferring 60 watts of power over a distance of up to two to three feet with efficiency of 75% [67]. At present, commercial products utilizing mid-range wireless energy transmission have been available on the market. The second area is the new battery material for ultra-fast charging. Ultra-fast charging was recently realized in $LiFePO_4$ by creating a fast ion-conducting surface phase through controlled off-stoichiometry [86]. It inherits and combines the advantages of both conventional Li-ion batteries and supercapacitors, which brings high energy density and can be charged at the rate as high as 400C [2]. Thus, this can shorten the time to fully charge a battery to few seconds.

While the SenCar arrives at an anchor point, it will also act as a data collector to gather the data from nearby sensors. Since the batteries can be charged in a very short time which can be almost neglected compare to the sojourn time, we assume that sensor batteries can be instantly fully-charged for use and the charging operation does not affect data gathering. In a particular time interval, as the SenCar moves over the anchor points in a tour, each sensor has the choice to send its data to the SenCar at any anchor point along low-cost routes. Moreover, in order to max-

[1] Midrange refers to the distance between the transmitter and the receiver that is longer than the size of the devices by a factor of at least 2 to 3 [65].

[2] C is determined by the nominal capacity of the battery. For a battery with the capacity of 1000mAh, C=1000mA.

imize network utility while maintaining perpetual operations, each sensor employs rate control to not only achieve high performance gain but also avoid draining out of energy before it can get recharged in a subsequent time interval. Next, we will present a carefully designed data gathering scheme that takes all these factors into consideration.

7.4 Joint Mobile Energy Replenishment and Data Gathering (J-MERDG)

Having outlined the basic idea of J-MERDG, in this section, we provide a two-step approach to efficiently implementing the design.

In the first step, the SenCar selects the anchor points for the current time interval by finding which sensors are to be recharged, where the SenCar will sojourn for data gathering, and how the SenCar moves over the field. In the second step, based on the information of SenCar's sojourn locations and the energy replenishing status, each sensor self determines how to transmit data to the SenCar when it arrives. The details of the approach are given in the following two subsections. Since similar methods are adopted in different time intervals, in the following discussions, we will focus on a typical time interval.

7.4.1 Anchor Point Selection

Since energy replenishment and data gathering are jointly considered, the selection of anchor points falls into following two aspects. First, the senors located at the selected anchor points should be those with most urgent needs of energy supplement. Second, as the SenCar moves over the anchor points back and forth for data gatherings during a time interval, the length of each migration tour, which implies the data gathering latency, is expected to be short. To better enjoy the benefit of the energy supply provided by the SenCar, more anchor points should be selected such that more sensors can timely get recharged. However, this would adversely prolong the migration tour. Therefore, there is an inherent tradeoff between the number of sensors to be recharged and data gathering latency. Based on this observation, the anchor point selection problem for a particular time interval k can be described as

Table 7.1: Anchor point selection algorithm for time interval k.

//$\mathcal{S}$ is the set of sensors, $\mathcal{B}_e^{(k-1)}$ is the set of energy states of sensors at the end of time interval $k-1$, and L is the tour length bound
Input: $\mathcal{S} = \{1, 2, \ldots, N\}$, $\mathcal{B}_e^{(k-1)} = \{\check{b}_i^{(k-1)} | i \in \mathcal{S}\}$, and L
Output: Anchor point list $\mathcal{A}^{(k)}$ for time interval k
Sort the battery states in $\mathcal{B}_e^{(k-1)}$ in an increasing order and record the result in another set $\mathcal{B}'$;
Map $\mathcal{S}$ to another set $\mathcal{S}'$ by rearranging the sensors in the sequence corresponding to their respective battery states in $\mathcal{B}'$;
$u \leftarrow 1$;
$v \leftarrow |\mathcal{S}'|$;
$m \leftarrow 0$;
$p \leftarrow 0$;
while *true* **do**
  if $u > v$
    $p \leftarrow v$; break;
  end if
  $m = \lfloor \frac{1}{2}(u+v) \rfloor$;
  // We use $\mathcal{S}'(m)$ to represent the m_{th} element in $\mathcal{S}'$
  $\mathcal{A}^{(k)} \leftarrow \{\mathcal{S}'(1), \mathcal{S}'(2), \ldots, \mathcal{S}'(m)\}$;
  Find an approximate shortest tour among the anchor points in $\mathcal{A}^{(k)}$ and denoted the result by TSP($\mathcal{A}^{(k)}$);
  case
    TSP($\mathcal{A}^{(k)}$) $< L$: $u \leftarrow m+1$;
    TSP($\mathcal{A}^{(k)}$) $= L$: $p \leftarrow m$; break;
    TSP($\mathcal{A}^{(k)}$) $> L$: $v \leftarrow m-1$;
  end case
end while
$\mathcal{A}^{(k)} \leftarrow \{\mathcal{S}'(1), \mathcal{S}'(2), \ldots, \mathcal{S}'(p)\}$;
Find an approximate shortest tour among the anchor points in $\mathcal{A}^{(k)}$;

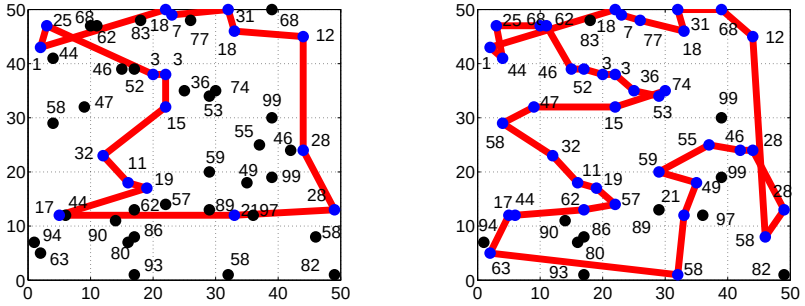
follows. Given the up-to-date energy states of sensors obtained by the SenCar at the end of time interval $k-1$, find the maximum number of anchor points for time interval k such that the sensors located at these anchor points hold the least battery energy, and meanwhile by visiting these anchor points, the tour length of the SenCar is no more than a threshold.

Considering that the possible candidate anchor points are the locations of all the sensors, this problem is equivalent to finding a target sensor, by visiting the locations of all the sensors with the battery energy less than or equal to which, the length of shortest migration tour among them is bounded by the threshold. Motivated by this observation, we propose a selection algorithm to search for the anchor points with the pseudo code shown in Table 7.1. Given the set of sensors $\mathcal{S}$, the set of energy states of sensors at the end of the previous time interval $\mathcal{B}_e^{(k-1)}$, and the tour

length bound L, the algorithm finds the anchor point list $\mathcal{A}^{(k)}$ for time interval k as follows.

As a pretreatment, the algorithm sorts the sensors with their battery energy in an increasing order. We record this sorted sensor list by $\mathcal{S}'$ and use $\mathcal{S}'(i)$ to represent the i_{th} element in the list. The problem is now converted to finding a target sensor $\mathcal{S}'(p)$ such that by visiting the senors with the index no more than p, i.e., $\mathcal{S}'(1), \mathcal{S}'(2), \ldots, \mathcal{S}'(p)$, the tour length is no more than L. To this end, the algorithm first finds the middle element of $\mathcal{S}'$, denoted by $\mathcal{S}'(m)$ and inspects the shortest migration tour among the locations of the sensors $\mathcal{S}'(1), \mathcal{S}'(2), \ldots, \mathcal{S}'(m)$. The migration tour can be found by an approximate solution to Traveling Salesman Problem (TSP). If the migration tour length equals the bound L, then the target sensor has been found; otherwise, the upper half or the lower half of the list is chosen to further search for the target sensor based on whether L is greater than or less than the migration tour among $\mathcal{S}'(1), \mathcal{S}'(2), \ldots, \mathcal{S}'(m)$. The algorithm reduces the number of elements needed to be checked to half each time, which is similar to the binary search algorithm [88]. We use $[u, v]$ to indicate the search range for the target sensor, where u and v are the indices of boundary elements. When there is no valid search range, i.e., $u > v$, it implies that there is no way to find a tour with the length exactly equal to L. Then, p is set by v and $\mathcal{S}'(p)$ is selected as the target sensor. By visiting the locations of $\mathcal{S}'(1), \mathcal{S}'(2), \ldots, \mathcal{S}'(p)$, the tour length will be closest to and less than L. It is clear that at most $\lceil \log(|\mathcal{S}|) \rceil$ rounds are needed to search for the target sensor. In each round, we need to calculate a tour among at most $|\mathcal{S}|$ sensors, which can be done in $O(|\mathcal{S}|^2)$ time. Therefore, the time complexity of the selection algorithm is $O(|\mathcal{S}|^2 \log(|\mathcal{S}|))$.

An example of the selection algorithm is illustrated in Fig. 7.3. There are 50 sensors in the network and their battery energy follows uniform distribution over $[0, 100]$. The nearest neighbor algorithm [88] for the TSP problem is used in our implementation to find the shortest tour among anchor points. The sensors on the migration tour are those to be charged in the current time interval. We can observe that regardless of the value of L, a higher precedence of sensors with lower energy are charged than other sensors. In addition, more sensors will be charged in the case of a larger L. For example, when $L = 200$m, 34% of the sensors, with the battery energy lower than or equal to 32, are charged and this results in a tour length of 193.2m. In contrast, when $L = 300$m, 78% of the sensors, with the battery energy



(a) $L = 200m$. (b) $L = 300m$.

Figure 7.3: An example to illustrate the selection algorithm to search for the anchor points in a time interval.

lower than or equal to 77, are charged, which leads to a tour length of 295.4m.

7.4.2 Optimal Mobile Data Gathering Scheme

After the anchor points are determined, the remaining work is how to gather the data from sensors when the SenCar migrates among the anchor points. We study such a mobile data gathering problem by formulating it into a utility maximization problem based on a flow-level network model. In the following, we first provide the problem formulation and then propose an optimization-based distributed algorithm for it.

Problem Formulation

For time interval k, consider a network with a set of static sensors, denoted by $\mathcal{S}$, and a set of anchor points, denoted by $\mathcal{A}^{(k)}$. To capture the characteristics of the SenCar movements over different anchor points in this time interval, we model the sensor network with the SenCar located at an anchor point a ($a \in \mathcal{A}^{(k)}$) by a directed acyclic graph $G_a^{(k)}(V_a^{(k)}, E_a^{(k)})$. $V_a^{(k)} = \mathcal{S} \bigcup \{\Lambda_a\}$ and represents the set of nodes, including all the sensors and the SenCar at anchor point a (denoted by Λ_a). $E_a^{(k)} = \{(i,j)|i,j \in V_a^{(k)}\}$, which is the set of directed links among the sensors and the SenCar. Sensor i generates data for the SenCar at a data rate of $r_{i,a}^{(k)}$ when the SenCar moves to anchor point a. The SenCar stays at each anchor point for a period of sojourn time $\tau^{(k)}$ in each of q tours in this time interval to collect data routed to

Table 7.2: List of notations.

Notation	Definition
$\mathcal{S}$	Set of sensors, i.e., $\mathcal{S} = \{1, 2, \ldots, N\}$
$\mathcal{B}^{(k)}$	Set of available battery energy of sensors for time interval k, i.e., $\mathcal{B}^{(k)} = \{b_i^{(k)} \mid i \in \mathcal{S}\}$
$\mathcal{B}_e^{(k)}$	Set of battery energy states of sensors at the end of time interval k, i.e., $\mathcal{B}_e^{(k)} = \{\check{b}_i^{(k)} \mid i \in \mathcal{S}\}$
B_i	Battery capacity of sensor i
$\mathcal{A}^{(k)}$	Set of anchor points for time interval k
$\mathcal{P}_{i,a}^{(k)}$	Set of parent nodes of sensor i for anchor point a in time interval k, i.e., $\mathcal{P}_{i,a}^{(k)} = \{j \mid (i,j) \in E_a^{(k)}\}$
$\mathcal{C}_{i,a}^{(k)}$	Set of child nodes of sensor i for anchor point a in time interval k, i.e., $\mathcal{C}_{i,a}^{(k)} = \{j \mid (j,i) \in E_a^{(k)}\}$
Λ_a	SenCar located at anchor point a
T	Length of each time interval
q	Number of migration tours in each time interval
L	Maximum migration tour length for SenCar
$\tau^{(k)}$	Sojourn time of SenCar at each anchor point in a migration tour during time interval k
$r_{i,a}^{(k)}$	Data rate of sensor i when SenCar sojourns at anchor point a during time interval k
$f_{ij,a}^{(k)}$	Flow rate over link (i,j) when SenCar is located at anchor point a in time interval k
$\Pi_a^{(k)}$	Feasible region of link capacity variable $f_{ij,a}^{(k)}$
σ	Portion parameter for the energy budget
e_{ij}	Energy consumed for transmitting a unit flow over link (i,j)
v_s	Moving velocity of the SenCar

it in multiple hops.

In our model, we use utility function $U_i(\cdot)$ to characterize the impact of the data from a sensor on the overall data gathering performance. We define $U_i(\cdot)$ as a strictly concave, increasing and twice-differentiable function with respect to the total amount of data gathered from sensor i in the current time interval (i.e., $\sum_{a \in \mathcal{A}^{(k)}} r_{i,a}^{(k)} q \tau^{(k)}$). Accordingly, the network utility is defined as the aggregation utility of all sensors. We are interested in maximizing the network utility while maintaining the perpetual operation of the network. To achieve this objective, we will address three critical issues: (1) what is the optimal data rate of a sensor for the SenCar sojourning at a particular anchor point; (2) how to schedule the link transmissions based on the interference model; (3) how to route the data to the SenCar at each anchor point taking into account of energy and link capacity constraints.

Now the mobile data gathering problem for time interval k can be formulated

as follows. For clarity, all the notations used are summarized in Table 7.2.

$$\textbf{MDG:} \quad \max_{\mathbf{r}^{(k)},\mathbf{f}^{(k)}} \sum_{i\in\mathcal{S}} U_i \left(\sum_{a\in\mathcal{A}^{(k)}} r_{i,a}^{(k)} q\tau^{(k)} \right) \tag{7.1}$$

Subject to

$$r_{i,a}^{(k)} + \sum_{j\in\mathcal{C}_{i,a}^{(k)}} f_{ji,a}^{(k)} = \sum_{j\in\mathcal{P}_{i,a}^{(k)}} f_{ij,a}^{(k)}, \forall i \in \mathcal{S}, \forall a \in \mathcal{A}^{(k)} \tag{7.2}$$

$$q\tau^{(k)} \sum_{a\in\mathcal{A}^{(k)}} \sum_{j\in\mathcal{P}_{i,a}^{(k)}} f_{ij,a}^{(k)} e_{ij} < \sigma b_i^{(k)}, \forall i \in \mathcal{S} \tag{7.3}$$

$$r_{i,a}^{(k)} \in R^+, f_{ij,a}^{(k)} \in \Pi_a^{(k)}, \forall i \in \mathcal{S}, \forall j \in \mathcal{P}_{i,a}^{(k)}, \forall a \in \mathcal{A}^{(k)} \tag{7.4}$$

where

$$b_i^{(k)} = \begin{cases} B_i, & \text{if } i \in \mathcal{A}^{(k)} \\ \check{b}_i^{(k-1)} & \text{otherwise} \end{cases} \quad \text{and } \tau^{(k)} = \frac{T - q\text{TSP}(\mathcal{A}^{(k)})/v_s}{q|\mathcal{A}^{(k)}|} \tag{7.5}$$

The constraints can be explained as follows.

- Flow conservation constraint (7.2) states that at each sensor for each anchor point, the aggregated outgoing link flow rates equal the local data rate plus incoming link flow rates.
- Energy constraint (7.3) enforces the energy cost at each sensor in a time interval bounded by its energy budget, which is a portion of available battery energy.
- Capacity constraint (7.4) specifies that the capacity allocated on a link for a particular anchor point must fall in the feasible capacity region $\Pi_a^{(k)}$. Based on the node exclusive interference model in [106], $\Pi_a^{(k)}$ can be similarly defined as the convex hull of all the rate vectors of the matchings in $G_a^{(k)}$.

In the formulation, $b_i^{(k)}$ and $\tau^{(k)}$ represent the available battery energy of sensor i for time interval k and the sojourn time at each anchor point in a tour during time interval k, respectively. As the sensors located at anchor points can get fast recharging, they are considered to have the full battery energy for use, i.e., $b_i^{(k)} = B_i$, if $i \in \mathcal{A}^{(k)}$.

Distributed Algorithm for MDG Problem

It is observed that the objective function of MDG problem is concave, however, not strictly concave with respect to $r_{i,a}^{(k)}$. Directly solving it with the dual approach [98] may incur oscillation before the system enters the equilibrium, which is not amenable for practical implementations. Therefore, we resort to the proximal optimization algorithm [97], which can be explained as follows.

Proximal Approximation Based Algorithm: A quadratic term $-\frac{1}{2c}\|\mathbf{r}^{(k)} - \mathbf{x}^{(k)}\|_2^2 = -\frac{1}{2c}\sum_{i\in\mathcal{S}}\sum_{a\in\mathcal{A}^{(k)}}\left(r_{i,a}^{(k)} - x_{i,a}^{(k)}\right)^2$ is added to the original objective function to make it strictly concave, where $\mathbf{r}^{(k)} = \{r_{i,a}^{(k)}\}$, $\mathbf{x}^{(k)}$ is an additional matrix and c is a positive scalar parameter. The proximal approximation algorithm runs in iterations, which alternatively maximizes the updated network utility over $\mathbf{r}^{(k)}$ while keeping $\mathbf{x}^{(k)}$ fixed, then over $\mathbf{x}^{(k)}$ while keeping $\mathbf{r}^{(k)}$ fixed, and repeats. In particular, the t_{th} iteration of the algorithm performs the following two steps.

Step 1: Fix $x_{i,a}^{(k)} = x_{i,a}^{(k)}[t]$ for all $i \in \mathcal{S}$ and $a \in \mathcal{A}^{(k)}$ and solve the following problem to obtain the optimal $r_{i,a}^{(k)}[t]$ and $f_{ij,a}^{(k)}[t]$.

$$\max_{\mathbf{r}^{(k)},\mathbf{f}^{(k)}} \sum_{i\in\mathcal{S}} U_i\left(\sum_{a\in\mathcal{A}^{(k)}} r_{i,a}^{(k)} q\tau^{(k)}\right) - \frac{1}{2c}\|\mathbf{r}^{(k)} - \mathbf{x}^{(k)}\|_2^2 \tag{7.6}$$

subject to constraints (7.2), (7.3) and (7.4).

Step 2: Set $x_{i,a}^{(k)}[t+1] = r_{i,a}^{(k)}[t]$ for all $i \in \mathcal{S}$ and $a \in \mathcal{A}^{(k)}$.

Now, the remaining work is to solve problem (7.6). Since it is a strictly concave problem with respect to $\mathbf{r}^{(k)}$, we apply the subgradient method based on dual decomposition for it, which is an efficient technique for convex programs and can naturally achieve the distributed implementation.

Dual Decomposition: We relax constraint (7.2) by introducing Lagrangian multiplier $\lambda_{i,a}^{(k)}$. Then, we can obtain the partial Lagrangian

$$\begin{aligned}
&L(\mathbf{r}^{(k)}, \mathbf{f}^{(k)}, \lambda^{(k)}) \\
&= \sum_i U_i(\sum_a r_{i,a}^{(k)} q\tau^{(k)}) - \tfrac{1}{2c}\cdot\sum_i\sum_a (r_{i,a}^{(k)} - x_{i,a}^{(k)})^2 - \sum_i\sum_a \lambda_{i,a}^{(k)}(r_{i,a}^{(k)} + \sum_j f_{ji,a}^{(k)} - \sum_j f_{ij,a}^{(k)}) \\
&= [\sum_i U_i(\sum_a r_{i,a}^{(k)} q\tau^{(k)}) - \tfrac{1}{2c}\sum_i\sum_a (r_{i,a}^{(k)} - x_{i,a}^{(k)})^2 - \sum_i\sum_a \lambda_{i,a}^{(k)} r_{i,a}^{(k)}] + \sum_i\sum_a\sum_j (\lambda_{i,a}^{(k)} - \lambda_{j,a}^{(k)}) f_{ij,a}^{(k)}
\end{aligned} \tag{7.7}$$

By duality, the dual problem is therefore

$$\min_{\lambda^{(k)} \succeq 0} g(\lambda) = \min_{\lambda^{(k)}} \max_{\mathbf{r}^{(k)}, \mathbf{f}^{(k)}} L(\mathbf{r}^{(k)}, \mathbf{f}^{(k)}, \lambda^{(k)}) \tag{7.8}$$

Note that the dual problem has a good separable property, which can be decomposed into two subproblems. One is the rate control subproblem in terms of rate variables $\mathbf{r}^{(k)}$, and another is the joint scheduling and routing subproblem to find optimal flow variables $\mathbf{f}^{(k)}$.

Rate Control Subproblem Given $\lambda_{i,a}^{(k)}$, each sensor solves a local optimization as follows by adjusting its data rates for different anchor points in time interval k.

$$\max_{r_{i,a}^{(k)} \geq 0} U_i(\sum_a r_{i,a}^{(k)} q\tau^{(k)}) - \frac{1}{2c}\sum_a (r_{i,a}^{(k)} - x_{i,a}^{(k)})^2 - \sum_a \lambda_{i,a}^{(k)} r_{i,a}^{(k)} \tag{7.9}$$

This local optimization can be solved by a similar approach to that in [102] with the complexity of $O(|\mathcal{A}^{(k)}| \log(|\mathcal{A}^{(k)}|))$, which is explained as follows.

Let μ_a be the Lagrangian multiplier for constraint $r_{i,a}^{(k)} \geq 0$. For each $a \in \mathcal{A}^{(k)}$, the Karush-Kuhn-Tucker conditions [98] are given by

$$\mu_a \geq 0, \tag{7.10}$$

$$\mu_a \cdot r_{i,a}^{(k)} = 0, \tag{7.11}$$

$$U_i'(\sum_a r_{i,a}^{(k)} q\tau^{(k)}) q\tau^{(k)} - \frac{1}{c} r_{i,a}^{(k)} + \frac{1}{c} x_{i,a}^{(k)} - \lambda_{i,a}^{(k)} + \mu_a = 0. \tag{7.12}$$

Let $m_{i,a}^{(k)} = \frac{1}{c} x_{i,a}^{(k)} - \lambda_{i,a}^{(k)}$. Then (7.12) can be rewritten as $U_i'(\sum_a r_{i,a}^{(k)} q\tau^{(k)}) q\tau^{(k)} - \frac{1}{c} r_{i,a}^{(k)} + m_{i,a}^{(k)} + \mu_a = 0$. If we sort the rates in the order that $m_{i,1}^{(k)} \geq m_{i,2}^{(k)} \geq \cdots \geq m_{i,|\mathcal{A}_i^{(k)}|}^{(k)}$, we have that for any $1 \leq a' \leq a \leq |\mathcal{A}_i^{(k)}|$, $r_{i,a'}^{(k)} \geq r_{i,a}^{(k)}$. This result trivially holds when $r_{i,a}^{(k)} = 0$. And when $r_{i,a}^{(k)} > 0$, we have $\mu_a = 0$ and $r_{i,a'}^{(k)} - r_{i,a}^{(k)} = c(m_{i,a'}^{(k)} - m_{i,a}^{(k)}) + \mu_{a'} \geq 0$. Therefore,zhao-energy by using this result and the KKT conditions, there exists an A such that

$$r_{i,a}^{(k)} = \begin{cases} c\left[q\tau^{(k)} U_i'(R_i^{(k)} q\tau^{(k)}) + m_{i,a}^{(k)}\right] > 0, & a \leq A \\ 0, & a > A, \end{cases} \tag{7.13}$$

where $R_i^{(k)} = \sum_{a=1}^{A} r_{i,a}^{(k)}$ denotes the sum of data rates for different anchor points. This implies that to find the optimal value for every $r_{i,a}^{(k)}$, we only need to find

optimal $R_i^{(k)}$ and the A.

When A is given, $R_i^{(k)}$ can be easily computed by solving the summation in (7.12) for all $a \leq A$. As $r_{i,a}^{(k)} > 0$ for all $a \leq A$, we have that $\mu_a = 0$ so that $R_i^{(k)}$ is the solution of $Aq\tau^{(k)} \cdot U_i'(R_i^{(k)} q\tau^{(k)}) - \frac{1}{c} R_i^{(k)} + \sum_{a=1}^{A} m_{i,a}^{(k)} = 0$. This way, all the work now becomes finding a proper A. As $r_{i,a}^{(k)}$ is decreasing in a and A is the boundary to indicate whether the rate is greater than or equal to zero, we search for A from $|\mathcal{A}_i^{(k)}|$ down to 1, and when $r_{i,A}^{(k)} = c\left[q\tau^{(k)} U_i'(R_i^{(k)} q\tau^{(k)}) + m_{i,A}^{(k)}\right] \geq 0$, we say A is found.

Finally, we summarize the rate control algorithm for subproblem (7.9) in Table 7.3.

Table 7.3: Distributed rate control algorithm at sensor i.

Sort anchor points in $\mathcal{A}^{(k)}$ such that $m_{i,a}^{(k)}$ is in the decreasing order;
$A \leftarrow |\mathcal{A}^{(k)}|$;
$M \leftarrow \sum_{a=1}^{|\mathcal{A}^{(k)}|} m_{i,a}^{(k)}$;
While $A \geq 1$ **do**
  // Use $R_i^{(k)}$ to denote $\sum_{a=1}^{A} r_{i,a}^{(k)}$
  Solve $Aq\tau^{(k)} \cdot U_i'(R_i^{(k)} q\tau^{(k)}) - \frac{1}{c} R + M = 0$ for optimal $R_i^{(k)}$;
  Compute $r_{i,A}^{(k)} = c\left[q\tau^{(k)} U_i'(R_i^{(k)} q\tau^{(k)}) + m_{i,A}^{(k)}\right]$;
  If $r_{i,A}^{(k)} \geq 0$
    Compute the data rate for each anchor point as

$$r_{i,a}^{(k)} = \begin{cases} c\left[q\tau^{(k)} U_i'(R_i^{(k)} q\tau^{(k)}) + m_{i,a}^{(k)}\right] & a \leq A \\ 0 & a > A \end{cases};$$

    Break;
  else
    $M \leftarrow M - m_{i,A}^{(k)}$;
    $A \leftarrow A - 1$;
  end If
end While

Joint Scheduling and Routing Subproblem Given $\lambda_{i,a}^{(k)}$, the how to schedule the link activation and how to allocate the flow rate on each scheduled link can be

determined by solving the following subproblem.

$$\begin{aligned} \max \quad & \sum_i \sum_a \sum_j (\lambda_{i,a}^{(k)} - \lambda_{j,a}^{(k)}) f_{ij,a}^{(k)} \\ \text{s.t.} \quad & q\tau^{(k)} \cdot \sum_a \sum_j f_{ij,a}^{(k)} e_{ij} < \sigma b_i^{(k)}, \forall i \in \mathcal{S} \\ & f_{ij,a}^{(k)} \in \Pi_a^{(k)}, \forall i \in \mathcal{S}, \forall j \in \mathcal{P}_{i,a}^{(k)}, \forall a \in \mathcal{A}^{(k)} \end{aligned} \tag{7.14}$$

We first ignore the energy constraint when consider the link schedule, i.e., for each anchor point a, choose $\tilde{f}_{ij,a}^{(k)}$ such that

$$\tilde{f}_{ij,a}^{(k)} \in \arg \max_{f_{ij,a}^{(k)} \in \Pi_a^{(k)}} \sum_i \sum_j (\lambda_{i,a}^{(k)} - \lambda_{j,a}^{(k)}) f_{ij,a}^{(k)} \tag{7.15}$$

If we consider $\lambda_{i,a}^{(k)} - \lambda_{j,a}^{(k)}$ as the weight of link (i, j) destined for the SenCar at anchor point a, this scheduling problem is equivalent to the maximum weighted matching problem under the node exclusive interference model. We can utilize the heuristic distributed algorithms in [106] and [16] to solve this problem in $O(|E_a^{(k)}|)$ time.

Then problem (7.14) is reduced to a routing problem that determines how much flow rate on each link based on the schedule and energy constraints. Each sensor only needs to solve the following local problem

$$\begin{aligned} \max \quad & \sum_a \sum_j (\lambda_{i,a}^{(k)} - \lambda_{j,a}^{(k)}) f_{ij,a}^{(k)} \\ \text{s.t.} \quad & q\tau^{(k)} \cdot \sum_a \sum_j f_{ij,a}^{(k)} e_{ij} < \sigma b_i^{(k)} \\ & f_{ij,a}^{(k)} \leq \tilde{f}_{ij,a}^{(k)}, \forall j \in \mathcal{P}_{i,a}^{(k)}, \forall a \in \mathcal{A}^{(k)}, \end{aligned} \tag{7.16}$$

where each $\tilde{f}_{ij,a}^{(k)}$ can be considered as the link capacity based on a specified schedule. This routing problem can be easily solved by an algorithm described in Table 7.4. The basic idea of the algorithm can be intuitively explained as follows. Each sensor always allocates a maximum flow rate under the energy and link capacity constraints to a scheduled link that has the largest link gain among all its outgoing links to different anchor points. Apparently, in the worst case, each sensor needs to consider the link gains of all its outgoing links for different anchor points. Thus, the time complexity of the routing algorithm at sensor i is $O(\sum_{a \in \mathcal{A}^{(k)}} \deg_a^+(i))$, where $\deg_a^+(i)$ is the outdegree of sensor i in the directed acyclic graph $G_a^{(k)}(V_a^{(k)}, E_a^{(k)})$.

Lagrangian Multiplier Update In each iteration of the subgradient algorithm, sensor i solves the subproblems in (7.9) and (7.14) with the current Lagrangian

Table 7.4: Distributed routing algorithm at sensor i.

Set $f_{ij,a}^{(k)}$ to zero, for all $j \in P_{i,a}^{(k)}$, $a \in \mathcal{A}^{(k)}$;
Set $\mathbf{W}_i = \left\{ \lambda_{i,a}^{(k)} - \lambda_{j,a}^{(k)} > 0 \middle| \forall j \in P_{i,a}^{(k)}, \forall a \in \mathcal{A}^{(k)} \right\}$ and sort it in the decreasing order;
Initialize the battery energy for allocation: $b_r = \sigma b_i^{(k)}$;
For $iter = 1; iter <= |P_{i,a}^{(k)}| \cdot |\mathcal{A}^{(k)}|; iter++$
  If $\mathbf{W}_i = \Phi$ or $b_r = 0$
    Break;
  end If
  $(\tilde{j}, \tilde{a})_i \leftarrow \arg\limits_{(j,a)} \mathbf{W}_i[iter]$;
  $f_{i\tilde{j},\tilde{a}}^{(k)} = \min \left\{ \frac{b_r}{q\tau^{(k)} e_{i\tilde{j}}}, \tilde{f}_{i\tilde{j},\tilde{a}}^{(k)} \right\}$;
  Remove $\mathbf{W}_i[iter]$ from $\mathbf{W}_i$;
  $b_r \leftarrow b_r - q\tau^{(k)} f_{i\tilde{j},\tilde{a}}^{(k)} e_{i\tilde{j}}$;
end For

multiplier $\lambda_{i,a}^{(k)}[n]$. Then, sensor i updates the Lagrangian multipliers as follows and sends them to its direct neighbors to facilitate the computing of $r^{(k)}$ and $f^{(k)}$ in the next iteration.

$$\lambda_{i,a}^{(k)}[n+1] = \left[\lambda_{i,a}^{(k)}[n] + \theta[n] \left(r_{i,a}^{(k)}[n] + \sum_j f_{ji,a}^{(k)}[n] - \sum_j f_{ij,a}^{(k)}[n] \right) \right]^+ \tag{7.17}$$

where $[\cdot]^+$ denotes the projection onto the non-negative orthant and $\theta[n]$ is a properly chosen scalar stepsize for subgradient iteration n. In our algorithm, we choose the diminishing stepsizes, i.e., $\theta[n] = d/(b+ck)$, $\forall k, c, d > 0, b \geq 0$, where b, c and d are adjustable parameters that regulate the convergence speed. The diminishing stepsize can guarantee the convergence regardless of the initial value of λ [98].

Recovery of Primal Solutions Since the subproblem of joint scheduling and routing is linear, which implies that the values in the optimal solution of the Lagrangian dual cannot be directly applied to the primal problem. In view of this, we still apply the method introduced in [100] to recover the optimal primal values for variables $f_{ij,a}^{(k)}$. When the variables $x_{i,a}^{(k)}$ converge in the higher level optimization, during the subgradient iterations in the lower level we construct the primal feasible

sequences $\{\hat{f}_{ij,a}^{(k)}[n]\}$ as follows.

$$\begin{aligned}\hat{f}_{ij,a}^{(k)}[n] &= \tfrac{1}{n}\sum_{h=1}^{n} f_{ij,a}^{(k)}[h] \\ &= \begin{cases} f_{ij,a}^{(k)}[1] & n=1 \\ \frac{n-1}{n}\hat{f}_{ij,a}^{(k)}[n-1] + \frac{1}{n} f_{ij,a}^{(k)}[n] & n>1 \end{cases}\end{aligned} \tag{7.18}$$

Optimal flow rates can be obtained when $\{\hat{f}_{ij,a}^{(k)}\}$ converges to $\hat{f}_{ij,a}^{(k)^*}$.

Finally, we summarize the proximal approximation based algorithm in Table 7.5, which is described in the context of sensor i for time interval k. As it is performed by each sensor and the information is only exchanged among direct neighbors, the algorithm is in a fully distributed manner.

Table 7.5: Summary of proximal approximation based algorithm.

Initialize $\mathbf{x}_i^{(k)} = \{x_{i,a}^{(k)} | a \in \mathcal{A}^{(k)}\}$ to non-negative values;
// *iterations for proximal approximation*
Repeat
  Initialize Lagrangian multipliers $\lambda_i^{(k)} = \{\lambda_{i,a}^{(k)} | a \in \mathcal{A}^{(k)}\}$ to non-negative values;
  // *iterations for subgradient and dual decomposition for* (7.6)
  Repeat
    Compute $\mathbf{r}_i^{(k)}[n] = \{r_{i,a}^{(k)}[n] | a \in \mathcal{A}^{(k)}\}$ by solving rate control subproblem (7.9);
    Compute $\mathbf{f}_i^{(k)}[n] = \{f_{ij,a}^{(k)}[n] | j \in \mathcal{P}_{i,a}^{(k)}, a \in \mathcal{A}^{(k)}\}$ by solving joint scheduling and routing subproblem (7.14);
    Update lagrangian multiplier and send them to direct neighbors;
    If $\mathbf{x}_i^{(k)}$ get converged in the high-level proximal iterations
      Compute primal feasible $\hat{f}_{ij,a}^{(k)}[n]$ by (7.18);
    end If
  Until $\lambda_i^{(k)}$ and $\mathbf{r}_i^{(k)}$ converge;
  Set $\mathbf{x}_i^{(k)}[t+1] = \mathbf{r}_i^{(k)}[t]$ in the t_{th} proximal iteration;
Until $\mathbf{x}_i^{(k)}$ converges and get the final optimal $\mathbf{r}_i^{(k)}$ and $\mathbf{f}_i^{(k)}$

7.5 Numerical Results

In this section, we evaluate the effectiveness of J-MERDG with extensive numerical results, and compare it with the performance of the solar harvesting sensor system.

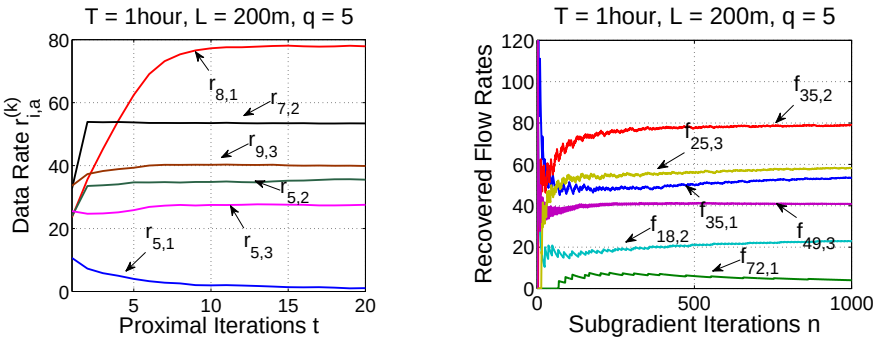
Table 7.6: Parameter settings.

Parameter	Value	Parameter	Value
B_i	2100mAh	e_{ij}	0.3mJ/Kbit
w_i	100	$e_{i\Lambda_i}$	0.02mJ/Kbit
$\theta(n)$	$\frac{1}{1+10n}$	L	200m
v_s	1m/s	σ	0.9

In the evaluation, we use a network consisting of 10 wireless rechargeable sensors distributed over a 100m$\times$100m area for demonstration purpose. In fact, due to the SenCar's capability of obtaining the sensor energy states along its migration tour and the distributed nature of the data gathering strategies, our design can be readily applicable to large scale networks. The utility function $U_i(\cdot)$ is defined as $w_i \log(\sum_a r_{i,a}^{(k)} q\tau + 1)$, where w_i is the weight of utility at sensor i. If not specified otherwise, the time interval length T is set to one hour and the number of migration tours q in each time interval is equal to 5. For clarity, all other parameter settings are summarized in Table 7.6.

7.5.1 Convergence of Proximal Approximation Based Algorithm

We first examine the convergence of the proximal approximation based algorithm. For a particular time interval k, we assume that each sensor randomly holds 80%-100% of full battery capacity and the anchor points are set to the locations of sensors 1, 2 and 3. Fig. 7.4(a) shows the evolution of data rates $r_{i,a}^{(k)}$ versus the number of proximal iterations. It can be seen that all data rates approach to the stable status after only 10 iterations. Fig. 7.4(b) shows the evolution of recovered flow rates $\hat{f}_{ij,a}^{(k)}$ on some selected links versus the number of subgradient iterations. It is noticed that the recovered flow rates are well within 5% of their optimal values after only 500 iterations. To further dampen the number of subgradient iterations, which is the lower-level optimization of the proximal approximation based algorithm, we can set the initial values of Lagrangian multipliers $\lambda_{i,a}^{(k)}$ by their final values in the previous run of higher-level proximal iterations. Comparable performance can be observed when the number of subgradient iterations is as low as around 100.



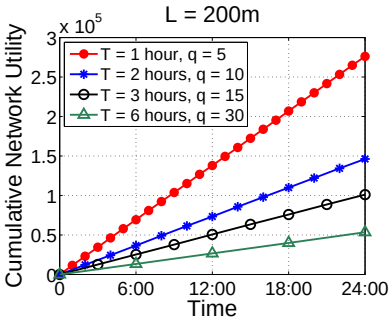
(a) Evolution of data rates versus proximal iterations t.

(b) Evolution of recovered flow rates versus subgradient iterations n.

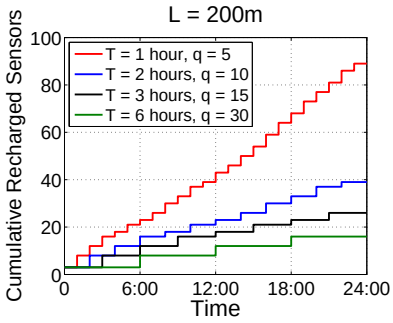
Figure 7.4: Convergence of the proximal approximation based algorithm.

7.5.2 Performance of J-MERDG

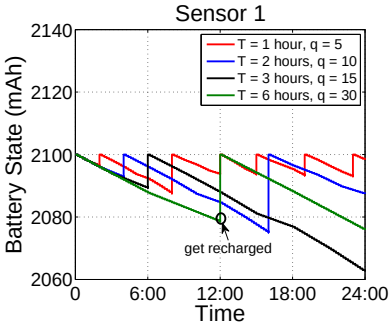
We now study the performance of J-MERDG varying with the time interval length T. Fig. 7.5(a) and 7.5(b) respectively demonstrate the evolution of cumulative network utility and cumulative number of recharged sensors in consecutive 24 hours under different settings of T. We assume that each sensor initially holds full battery capacity and the number of migration tours q in each time interval is proportional to T. From the figures, we can see that higher cumulative network utility can be obtained and there are more recharged sensors in the cases with a smaller T. This is reasonable since a smaller T leads to shorter waiting time for the sensors to get recharged. However, a small T may cause sensors to frequently calculate their data rates, schedule and routing so as to achieve the optimal data gatherings. Therefore, a proper setting for T is actually to balance the tradeoff between the computation overhead and achievable performance. Fig. 7.5(c) and 7.5(d) depict the battery states of sensor 1 and sensor 8 evolving with time, respectively. It is shown that sensors could timely get recharged to avoid energy depletion such that perpetual operations of the network can be guaranteed. It is apparent that sensors have more chances for energy replenishment under a smaller T. For example, sensor 8 is recharged for 9 times within 24 hours when $T = 1$ while it only gets recharged twice when $T = 6$.



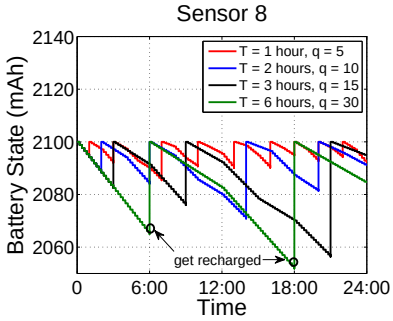
(a) Cumulative network utility.



(b) Cumulative number of recharged sensors.



(c) Battery states of sensor 1.



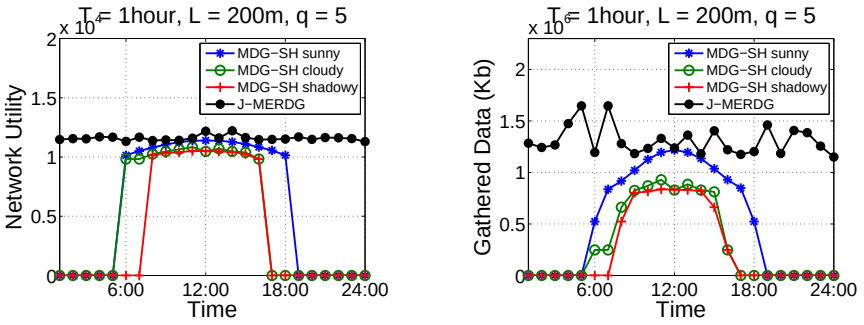
(d) Battery states of sensor 8.

Figure 7.5: Performance of J-MERDG as the function of T.

7.5.3 Comparison with Solar Harvesting Sensor System

In this subsection, we compare J-MERDG with the mobile data gathering in the solar harvesting sensor system, denoted by MDG-SH for short. In MDG-SH scheme, each sensor harvests solar irradiance to self-support its energy consumption and a mobile collector visits each anchor point only for data gatherings. In order to sustain network operations, the energy consumption rate of each sensor can not be higher than the energy harvesting rate [76]. We build the recharging profiles of sensors for MDG-SH by using real solar irradiation measurements collected by the Baseline Measurement System (Global 40-South PSP) at National Renewable Energy Laboratory [84]. In particular, we choose 30 days of year 2009 and categorize them into three sets based on the irradiance statistics. We use the average irradiance of these three sets to represent the solar power on sunny, cloudy and shadowy days, respectively. Based on the irradiance, the recharging rate can be derived as follows

$$\pi_r = I \times \eta_p \times \rho_e \times A, \tag{7.19}$$



(a) Network utility. (b) Amount of gathered data in a tour.

Figure 7.6: Performance comparisons between J-MERDG and MDG-SH.

where I represents the solar irradiance, η_p is the efficiency of the solar panel to convert solar irradiance to electrical power, ρ_e is the electrical regulating and charging efficiency, and A is the size of solar panel. In our simulations, each sensor is assumed to have an equal recharging rate. And we set $\eta_p \times \rho_e$ to 0.06 and let A equal 37mm$\times$37mm.

Fig. 7.6 compares J-MERDG with MDG-SH in consecutive 24 hours in terms of network utility and the amount of gathered data. The amount of gathered data is used as one of the metrics because it more evidently visualizes the difference among different solutions than network utility, which is in logarithmic scale and results in a small slope at high data rates [76]. It is apparent in Fig. 7.6(a) that J-MERDG greatly outperforms MDG-SH all the time. On the average, J-MERDG achieves 48%, 59% and 66% higher network utility than MDG-SH in sunny, cloudy and shadowy days, respectively. These observations are also applicable to the amount of gathered data. It is shown in Fig. 7.6(b) that, during the day time (8:00-18:00), J-MERDG can collect 21%, 48% and 54% more data than MDG-SH in sunny, cloudy and shadowy days, respectively. Moreover, during night time, MDG-SH has no chance to harvest solar energy and thus cannot extract any data, while J-MERDG can still keep working and maintain the network utility almost as high as that of day time. This fact signifies the importance of our work, which provides a recharging scheme immune to the environmental variations and also achieves high-performance data gathering.

7.6 Conclusions

In this chapter, we have studied the joint design of energy replenishment and data gathering (J-MERDG) in wireless sensor networks by exploiting mobility. Specifically, a multi-functional SenCar is introduced to the wireless rechargeable sensor networks, which migrates among selected anchor points, charges the located sensors via wireless energy transmissions, and collects data from nearby sensors in multi-hop routing. In J-MERDG, we first presented a selection algorithm to determine the anchor points, which achieves a desirable balance between the energy replenishing amount and data gathering latency. Then, we explored the optimal data gathering performance when the SenCar moves over different anchor points. Each sensor self-tunes the data rate, scheduling, and routing based on the up-to-date energy replenishing status such that the system-wide network utility can be maximized. Numerical results demonstrated that J-MERDG can effectively maintain perpetual network operations and outperform solar harvesting systems by 48% in network utility.

Chapter 8

Conclusions

This dissertation focuses its study on mobile data gathering in wireless sensor networks. A suite of scheme design and performance optimization algorithms have been presented to identify and address some critical issues in this research field. Specifically, first, a joint design of controlled mobility and space-division multiple access technique has been proposed [108]-[110] for mobile data gathering. It leverages the mobility to alleviate the non-uniformity of energy consumption among sensors as well as utilizing the concurrent data transmission to shorten data gathering time. Second, a region division and tour planning algorithm is presented [111] to extend the joint design of mobility and SDMA technique to large-scale sensor network with multiple SenCars. The data gathering latency can be minimized by balancing the data gathering time among different regions. Third, the bounded relay hop mobile data gathering scheme [113] is introduced to explore the tradeoff between energy saving and data gathering latency. It incorporates the multi-hop relays of local data aggregation into mobile data gathering while keeping the relay hop count constrained to a certain level to limit the energy consumption at sensors. Fourth, a pricing-based distributed algorithm [115][116] is proposed for performance optimization on the anchor-based range traversing data gathering scheme, which is modeled under cost minimization framework. The optimal performance can be achieved at the equilibrium that reconciliates the payment and the shadow price between each sensor and the SenCar for communication opportunities. Fifth, two distributed algorithms [117][118] have been presented to effectively adjust data rate, routing, and sojourn time allocation so as to achieve system-wide

network utility maximization with fast convergence. Finally, mobile data gathering is jointly considered with energy replenishment in wireless rechargeable sensor networks. The SenCar plays as not only a data collector but also a transporter for energy delivery, which enable steady recharging rate and high-energy data gathering to be achieved simultaneously [119]. A two-step approach is applied to implement the design, which computes the migration tour of the SenCar based on the joint consideration of recharging demand and data gathering performance, and also adapts adjustable data rate, link schedule, and flow routing to the up-to-date energy replenishing status so as to optimize the data gathering utility.

To summarize, in this dissertation, we have conducted comprehensive studies on mobile data gathering in wireless sensor networks. We have proposed efficient solutions to improve data gathering performance from both systematic and theoretical points of view, which combine algorithm design, optimization, analysis and simulation techniques. The outcome of this research can be applicable to a wide spectrum of applications, including environmental monitoring, field surveillance, home automation and other commercial areas. Therefore, our research would have a significant impact on fundamental design principles and infrastructures for the development of future sensor networks.

Bibliography

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, “A survey on sensor networks,” *IEEE Commun. Magazine*, pp. 102-114, 2002.

[2] E.Biagioni and K. Bridges,”The application of remote sensor technology to assist the recovery of rare and endangered species,” Special Issue on Distributed Sensor Networks, *International Journal of High Performance Computing Applications,* vol.16, no. 3, Aug. 2002.

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

[4] A. Mainwaring, J.Polastre, R. Szewczyk, D. Culler, and J. Anderson, ”Wireless sensor networks for habitat monitoring,” *ACM International Workshop on Wireless Sensor Networks and Applications (WSNA ’02)*, Atlanta, GA, Sept. 2002.

[5] S. Chessa and P. Santi, ”Crash faults identification in wireless sensor networks”, *Computer Communications,* vol. 25, no. 14, pp. 1273-1282, Sept. 2002.

[6] L. Schwiebert, S.K.S. Gupta, and J. Weinmann, ”Research challeneges in wireless networks of biomedical sensors,” *ACM/IEEE Conf. on Mobile Computing and Networking (MobiCom)*, pp. 151-165, 2001.

[7] K. Romer and F. Mattern, “The design space of wireless sensor networks,” *IEEE Wireless Communications*, vol. 11, no. 6, pp. 54-61, Dec. 2004.

[8] A. Tiwari, P. Ballal, and F. L. Lewis, “Energy-efficient wireless sensor network design and implementation for condition-based maintenance,” ACM *Transactions on Sensor Networks (TOSN)*, vol. 3, no. 1, March 2007.

[9] A. Scaglione and S. D. Servetto, "On the interdependence of routing and data compression in multi-hop sensor networks," *MobiCom*, 2002.

[10] D. Marco, E.J. Duarte-Melo, M. Liu and D. Neuhoff, "On the many-to-one transport capacity of a dense wireless sensor network and the compressibility of its data," *IPSN*, April 2003.

[11] D. England, B. Veeravalli, and J. Weissman, "A robust spanning tree topology for data collection and dissemination in distributed environments," *IEEE Trans. Parallel and Distributed Systems*, vol. 18, no. 5, pp. 608-620, 2007.

[12] K. Jain, J. Padhye, V. N. Padmanabhan and L. Qiu, "Impact of interference on multi-hop wireless network performance," *MobiCom*, 2003.

[13] E. J. Duarte-Melo and M. Liu, "Data-gathering wireless sensor networks: organization and capacity," *Elsevier Computer Networks*, vol. 43, pp. 519-537, 2003.

[14] H. E. Gamal, "On the scaling laws of dense wireless sensor networks: the data gathering channel," *IEEE Trans. Information Theory*, vol. 51, no. 3, pp. 1229-1234, March 2005.

[15] J. Zhao and R. Govindan, "Understanding packet delivery performance in dense wireless sensor networks," *ACM Sensys*, Nov. 2003.

[16] U. Akyol, M. Andrews, P. Gupta, J. Hobby, I. Saniee and A. Stolyar, "Joint scheduling and congestion control in mobile ad-hoc networks," *IEEE INFOCOM*, 2008.

[17] W. R. Heinzelman, A. Chandrakasan and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," *Hawaii Int. Conf. System Sciences (HICCS)*, Jan. 2000.

[18] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: a hybrid, energy-efficient approach," *IEEE INFOCOM*, 2004.

[19] A. Manjeshwar and D. P. Agrawal, "Teen: a routing protocol for enhanced efficiency in wireless sensor networks," *IEEE IPDPS*, April 2001.

[20] Z. Zhang, M. Ma and Y. Yang, "Energy efficient multi-hop polling in clusters of two-layered heterogeneous sensor networks," *IEEE Trans. Computers*, vol. 57. no. 2, pp. 231-245, Feb. 2008.

[21] B. Sheng, Q. Li and W. Mao, "Data storage placement in sensor networks," *in Proc. of ACM Mobihoc*, Florence, Italy, May 2006.

[22] R. Shah, S. Roy, S. Jain and W. Brunette, "Data MULEs: modeling a three-tier architecture for sparse sensor networks," *Elsevier Ad Hoc Networks Journal*, vol. 1, pp. 215-233, Sept. 2003.

[23] W. Zhao, M. Ammar and E. Zegura, "A message ferrying approach for data delivery in sparse mobile ad hoc networks," *ACM MobiHoc*, 2004.

[24] A. Pentland, R. Fletcher and A. Hasson, "Daknet: rethinking connectivity in developing nations," *IEEE Computer*, Jan. 2004.

[25] A. Chakrabarty, A. Sabharwal and B. Aazhang, "Using predictable observer mobility for power efficient design of a sensor network," *Second International Workshop on Information Processing in Sensor Networks (IPSN)*, April 2003.

[26] D. Jea, A. A. Somasundara and M.B. Srivastava, "Multiple controlled mobile elements (data mules) for data collection in Sensor Networks," *IEEE/ACM DCOSS*, June 2005.

[27] M. Ma and Y. Yang, "SenCar: an energy-efficient data gathering mechanism for large-scale multihop sensor networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 18, no. 10, Oct. 2007.

[28] M. Ma and Y. Yang, "Data gathering in wireless sensor networks with mobile collectors," *IEEE IPDPS*, Miami FL, April 2008.

[29] W. Zhao, M. Ammar and E. Zegura, "Controlling the mobility of multiple data transport ferries in a delay-tolerant network," *IEEE INFOCOM 2005*.

[30] A. A. Somasundara, A. Ramamoorthy, and M. B. Srivastava, "Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines," *IEEE Realtime Systems Symposium*, Dec. 2004.

[31] E. Ekici, Y. Gu, and D. Bozdag, "Mobility-Based Communication in Wireless Sensor Networks," *IEEE Communications Magazine*, July 2006.

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

[33] H. Nakayama, N. Ansari, A. Jamalipour and N. Kato, "Fault-resilient sensing in wireless sensor networks," *Computer Communications*, vol. 30, pp. 2375-2384, Sept. 2007.

[34] S. Nesamony, M. K. Vairamuthu, M. E. Orlowska, "On optimal route of a calibrating mobile sink in a wireless sensor network," *Fourth International Conference on Networked Sensing Systems*, June 2007.

[35] S. Basagni, A. Carosi, E. Melachrinoudis, C. Petrioli and Z. M. Wang. "Controlled sink mobility for prolonging wireless sensor networks lifetime". *ACM Wireless Networks*, 2007.

[36] G. Xing, T. Wang, W. Jia, and M. Li, "Rendezvous design algorithm for wireless sensor networks with a mobile base station," *ACM Mobihoc*, May 2008.

[37] K. Dantu, M. Rahimi, H. Shah, S. Babel, A. Dhariwal and G. S. Sukhatme, "Robomote: enabling mobility in sensor networks," *Fourth International Workshop on Information Processing in Sensor Networks (IPSN)*, April 2005.

[38] O. Chipara, Z. He, G. Xing, Q. Chen, X. Wang, C. Lu, J. Stankovic, and T. Abdelzaher, "Real-time power-aware routing in sensor networks," *in Proc. of IEEE IWQoS*, 2006.

[39] R. Pon, M. A. Batalin, J. Gordon, A. Kansal, D. Liu, M. Rahimi, L. Shirachi, Y. Yu, M. Hansen, W. J. Kaiser, M. Srivastava, G. Sukhatme and D. Estrin, "Networked infomechanical systems: a mobile embedded networked sensor platform," *in Proc. of ACM/IEEE IPSN*, 2005.

[40] M. A. Batalin, M. Rahimi, Y. Yu, D. Liu, A. Kansal, G. S. Sukhatme, W. J. Kaiser, M. Hansen, G. J. Pottie, M. Srivastava and D. Estrin, "Call and response: experiments in sampling the environment," *in Proc. of ACM SenSys*, 2004.

[41] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh and D. Rubenstein, "Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet," *in Proc. of Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2002.

[42] T. Small and Z. Haas, "The shared wireless infostation model - a new ad hoc networking paradigm (or where there is a whale, there is a way)," *in Proc. of ACM MobiHoc* 2003.

[43] B. Kusy, H. Lee, M. Wicke, N. Milosavljevic, and L. Guibas, "Predictive QoS routing to mobile sinks in wireless sensor networks,"*in Proc. of International Conference of Information Processing in Sensor Networks (IPSN)*, 2009.

[44] K. Karenos and V. Kalogeraki, "Traffic management in sensor networks with a mobile sink," *IEEE Trans. on Parallel and Distributed systems*, vol. 21, no. 10, pp. 1515 - 1530, Oct. 2010.

[45] X. Xu, J. Luo and Q. Zhang, "Delay tolerant event collection in sensor networks with mobile sink," *in Proc. of IEEE INFOCOM*, 2010.

[46] Y. Yun and Y. Xia, "Maximizing the lifetime of wireless sensor networks with mobile sink in delay-tolerant applications," *IEEE Trans. on Mobile Computing*, vol. 9, no. 9, pp. 1308 - 1318, Sept. 2010.

[47] D. Bote, K. Sivalingam and P. Agrawal, "Data gathering in ultra wide band based wireless sensor networks using a mobile node," *in Proc. of Fourth International Conference on Broadband Communications, Networks, and Systems (BroadNets)*, Raleigh, NC, Sept. 2007.

[48] S. Thoen, L. V. D. Perre, M. Engels and H. D. Man, "Adaptive loading for OFDM/SDMA-based wireless networks," *IEEE Trans. Communications*, vol. 50, no. 11, pp. 1798-1810, Nov. 2002.

[49] B. Suard, G. Xu, H. Liu and T. Kailath, "Uplink channel capacity of space-division-multiple-access schemes," *IEEE Trans. Information Theory*, vol. 44, no. 4, pp. 1468-1476, July 1998.

[50] R. Madan and S. Lall, "Distributed algorithms for maximum lifetime routing in wireless sensor networks," *IEEE Trans. Wireless Communications*, vol. 5, no. 8, pp. 2185 - 2193, 2006.

[105] R. Madan, S. Cui, S. Lall and A. Goldsmith, "Modeling and optimization of transmission schemes in energy-constrained wireless sensor networks," *IEEE/ACM Trans. Networking*, vol. 15, no. 6, pp. 1359 - 1372, 2007.

[52] Y. Hou, Y. Shi and H. Sherali, "Rate allocation and network lifetime problems for wireless sensor networks," *IEEE/ACM Trans. Networking*, vol. 16, no. 2, pp. 321 - 334, 2008.

[53] C. Hua and T. Yum, "Optimal routing and data aggregation for maximizing lifetime of wireless sensor networks," *IEEE/ACM Trans. Networking*, vol. 16, no. 4, pp. 892 - 903, 2008.

[54] C. Zhang, J. Kurose, Y. Liu, D. Towsley and M. Zink, "A distributed algorithm for joint sensing and routing in wireless networks with non-steerable directional antennas," in *Proc. 14th IEEE International Conference on Network Protocols (ICNP)*, Nov. 2006.

[55] Y. Wu, S. Fahmy and N. B. Shroff, "On the construction of a maximum-lifetime data gathering tree in sensor networks: NP-completeness and approximation algorithm," in *Proc. of IEEE INFOCOM*, 2008.

[56] N. Sadagopan and B. Krishnamachari, "Maximizing data extraction in energy-limited sensor networks," in *Proc. IEEE INFOCOM*, 2004.

[57] S. Chen, Y. Fang and Y. Xia, "Lexicographic Maxmin Fairness for Data Collection in Wireless Sensor Networks," *IEEE Trans. on Mobile Computing*, vol. 6, no. 7, pp. 762 - 776, 2007.

[58] H. Liu, X. Jia, P. Wan, C. Yi, S. Makki and N. Pissinou, "Maximizing lifetime of sensor surveillance systems," *IEEE/ ACM Trans. Networking*, vol. 15, no. 2, pp. 334-345, 2007.

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

[60] F. Kelly, "Charging and rate control for elastic traffic," *European Transactions on Telecommunications*, vol. 8, 1997.

[61] F. Kelly, A. Maullo and D. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, no. 3, March 1998.

[62] I. H. Hou and P. R. Kumar, "Utility maximization for delay constrained QoS in wireless," in *Proc. of IEEE INFOCOM*, 2010.

[63] Y. Qiu and P. Marbach, "Bandwidth allocation in ad hoc networks: a price-based approach," *IEEE INFOCOM*, April 2003.

[64] N. Tesla, "Apparatus for transmitting electrical energy," *US patent number 1,119,732*, Dec 1914.

[65] A. Kurs, A. Karalis, R. Moffatt, J. D. Joannopoulos, P. Fisher, and M. Soljacic, "Wireless power transfer via strongly coupled magnetic resonances," *Science*, vol. 317, pp. 83, 2007.

[66] A. Karalis, J. D. Joannopoulos and M. Soljacic, "Efficient wireless non-radiative mid-range energy transfer," *Annals of Physics*, vol. 323, no. 1, pp 34-48, Jan 2008.

[67] Intel, *Wireless resonant energy link (WREL) demo*, Available online: http://software.intel.com/en-us/videos/wireless-resonant-energy-link-wrel-demo/.

[68] Q. Yuan, Q. Chen, L. Li, and K. Sawaya, "Numerical analysis on transmission efficiency of evanescent resonant coupling wireless power transfer system," *IEEE Transactions on Antennas and Propagation*, vol. 58, no. 5, May 2010.

[69] S. J. Roundy, "Energy scavenging for wireless sensor nodes with a focus on vibration to electricity conversion," *Ph.D. Dissertation, Dept. of EECS, UC Berkeley*, May 2003.

[70] T. Voigt, H. Ritter and J. Schiller, "Utilizing solar power in wireless sensor networks," in *Proc. of the 28th Annual IEEE International Conference on Local Computer Networks (LCN)*, 2003.

[71] C. Moser, L. Thiele, D. Brunelli and L. Benini, "Robust and low complexity rate control for solar powered sensors," in *Proc. of the Conference on Design, Automation and Test in Europe*, pp. 230-235, 2008.

[72] K. Lin, J. Yu, J. Hsu, S. Zahedi, D. Lee, J. Friedman, A. Kansal, V. Raghunathan and M. Srivastava, "Heliomote: enabling long-lived sensor networks through solar energy harvesting," in *Proc. of SenSys*, 2005.

[73] C. Park and P. Chou, "AmbiMax: autonomous energy harvesting platform for multi-supply wireless sensor nodes," in *Proc. of IEEE SECON*, Sept. 2006.

[74] K. Kar, A. Krishnamurthy and N. Jaggi, "Dynamic node activation in networks of rechargeable sensors," *IEEE/ACM Transactions on Networking*, vol. 14, no. 1, pp. 15-26, February 2006.

[75] L. Lin, N. B. Shroff and R. Srikant, "Asymptotically optimal energy-aware routing for multihop wireless networks with renewable energy sources," *IEEE/ACM Transactions on Networking*, vol. 15, no. 5, pp. 1021-1034, October 2007.

[76] R. S. Liu, P. Sinha and C. E. Koksal, "Joint energy management and resource allocation in rechargeable sensor networks," in *Proc. of IEEE INFOCOM*, March 2010.

[77] V. Sharma, U. Mukherji, V. Joseph and S. Gupta, "Optimal energy management policies for energy harvesting sensor nodes," *IEEE Transactions on Wireless Communications*, vol. 9, no. 4, April 2010.

[78] C. Vigorito, D. Ganesan and A. Barto, "Adaptive control of duty cycling in energy-harvesting wireless sensor networks," in *Proc. of IEEE SECON*, June 2007.

[79] M. Rahimi, H. Shah, G. Sukhatme, J. Heideman and D. Estrin,"Studying the feasibility of energy harvesting in a mobile sensor network," in *Proc. of IEEE International Conference on Robotics and Automation*, 2003.

[80] J. R. Piorno, C. Bergonzini, D. Atienza and T. S. Rosing, "Prediction and management in energy harvested wireless sensor nodes," *The 1st International Conference Wireless Vitae on Wireless Communications, Vehicular Technology, Information Theory and Aerospace and Electronic Systems Technology (VITAE)*, vol. 1, no.1, pp. 6-10, 2009.

[81] Y. Gu and T. He, "Bounding communication delay in energy harvesting sensor networks," in *Proc. of 2010 International Conference on Distributed Computing Systems (ICDCS)*, 2010.

[82] M. Tacca, P. Monti and A. Fumagalli, "Cooperative and reliable ARQ protocols for energy harvesting wireless sensor nodes," *IEEE Transactions on Wireless Communications*, vol. 6, no. 7, July 2007.

[83] Z. A. Eu, H. P. Tan and W. K. G. Seah, "Wireless sensor networks powered by ambient energy harvesting: an empirical characterization," in *Proc. of IEEE ICC*, 2010.

[84] National Renewable Energy Laboratory, http://www.nrel.gov.

[85] S. Dearborn, "Charging Li-ion batteries for maximum run times," *Power Electronics Technology Mag.*, pp. 40-49, April 2005.

[86] B. Kang and G. Ceder, "Battery materials for ultrafast charging and discharging," *Nature* 458, pp.190-193, March 2009.

[87] D. N. C. Tse and P. Viswanath, *Fundamentals of Wireless Communication*, Cambridge University Press, May 2005.

[88] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, *Introduction to Algorithms*, MIT Press, 2001.

[89] B. Gavish, "Formulations and algorithms for the capacitated minimal directed tree problem," *Journal of ACM*, vol. 30, pp. 118-132, 1983.

[90] D. B. West, *Introduction to graph theory*, Prentice-Hall, 1996.

[91] J. Edmonds, "Paths, trees, and flowers," *Canad. J. Math*, 1965.

[92] H. N. Gabow, "An efficient implementation of Edmonds' algorithm for maximum matching on graphs," *Journal of ACM*, 1976.

[93] R. K. Ahuja, T. L. Magnanti and J.B. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice-Hall, 1993.

[94] CPLEX package, *http://www.ilog.com/products/cplex/*.

[95] AMPL package, *http://www.ampl.com/*, 2007.

[96] D. Bertsekas, A. Nedic and A. Ozdaglar, *Convex analysis and optimization*, Athena Scientific, 2003.

[97] D. Bertsekas and J. Tsitsiklis, *Parallel and distributed computation: numerical methods*, Athena Scientific, 1997.

[98] S. Boyd and L. Vandenbergh, *Convex optimization*, Cambridge University Press, 2004.

[99] J. G. Wardrop, "Some theoretical aspects of road traffic research," *Inst. Civ. Eng.*, Part 2, 1952.

[100] H. Sherali and G. Choi, "Recovery of primal solutions when using subgradient optimization methods to solve Lagrangian duals of linear programs," *Operations Research Letters*, vol. 19, 1996.

[101] L. Xiao, M. Johansson and S. Boyd, "Simultaneous routing and resources allocation via dual decomposition," *IEEE Trans. Communications*, vol. 52, no. 7, July 2004.

[102] X. Lin and B. Shroff, "Utility maximization for communication networks with multipath routing," *IEEE Trans. Automatic Control*, vol. 51, no. 5, pp. 766 - 781, 2006.

[103] M. Chiang, S. Low, A. Calderbank and J. Doyle, "Layering as optimization decomposition: a mathematical theory of network architectures," *IEEE Proceedings*, 2007.

[104] C. Wu and B. Li, "Strategies of conflict in coexisting streaming overlays," in *Proc. of IEEE INFOCOM*, May 2007.

[105] R. Madan, Z. Luo, and S. Lall, "A distributed algorithm with linear convergence for maximum lifetime routing in wireless networks," *the Allerton Conf. on Comm., Control, and Computing*, Sep. 2007.

[106] L. Chen, T. Ho, S. Low, M. Chiang and J. Doyle, "Optimization based rate control for multicast with network coding," in *Proc. of IEEE INFOCOM*, 2007.

[107] Z. Q. Luo and P. Tseng, "On the rate of convergence of a distributed asynchronous routing algorithm," *IEEE Trans. Automatic Control*, vol. 39, no. 5, 1994.

[108] M. Zhao, M. Ma and Y. Yang, "Mobile data gathering with multiuser MIMO technique in wireless sensor networks," *IEEE Global Telecommunications Conference (GLOBECOM)*, Nov. 2007.

[109] M. Zhao, M. Ma and Y. Yang, "Mobile data gathering with space-division multiple access in wireless sensor networks," *The 27th IEEE International Conference on Computer Communications (INFOCOM)*, April 2008.

[110] Z. Zhang, Y. Yang and M. Zhao, "Enhancing downlink performance in wireless networks by simultaneous multiple packet transmission," *IEEE Transactions on Computers*, vol. 58, no. 5, pp. 706-718, 2009.

[111] M. Zhao, M. Ma and Y. Yang, "Efficient data gathering with mobile collectors and space-division multiple access technique in wireless sensor networks," to appear in *IEEE Transactions on Computers*, 2010.

[112] M. Zhao and Y. Yang, "A framework for mobile data gathering with load balanced clustering and MIMO uploading," accepted by *The 30th IEEE International Conference on Computer Communications (INFOCOM)*, April 2011.

[113] M. Zhao and Y. Yang, "Bounded relay hop mobile data gathering in wireless sensor networks," *The Sixth IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE MASS)*, Oct. 2009.

[114] M. Zhao and Y. Yang, "Bounded relay hop mobile data gathering in wireless sensor networks," (Journal Version) to appear in *IEEE Transactions on Computers*, 2010.

[115] M. Zhao, D. Gong and Y. Yang, "A cost minimization algorithm for mobile data gathering in wireless sensor networks," *The 7th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE MASS)*, Nov. 2010.

[116] M. Zhao, D. Gong and Y. Yang, "Cost minimization for mobile data gathering in wireless sensor networks," under submission.

[117] M. Zhao and Y. Yang, "An optimization based distributed algorithm for mobile data gathering in wireless sensor networks," *The 29th IEEE International Conference on Computer Communications (INFOCOM) mini-conference*, March 2010.

[118] M. Zhao and Y. Yang, "Optimization based distributed algorithms for mobile data gathering in wireless sensor networks," under submission.

[119] M. Zhao, J. Li and Y. Yang, "Joint mobile energy replenishment and data gathering in wireless rechargeable sensor networks," under submission.