Stony Brook University

Building a High Performance Perpetual Wireless Sensor Network by Wireless Charging

A Dissertation Presented

by

Cong Wang

to

The Graduate School

in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

in

Electrical Engineering

Stony Brook University

May 2017

Stony Brook University

The Graduate School

Cong Wang

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

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

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

Fan Ye
Assistant Professor, Department of Electrical and Computer Engineering

Samir Das
Professor, Department of Computer Science

This dissertation is accepted by the Graduate School.

Charles Taber
Dean of the Graduate School

Abstract of the Dissertation

Building a High Performance Perpetual Wireless Sensor Network by Wireless Charging

by

Cong Wang

Doctor of Philosophy

in

Electrical Engineering

Stony Brook University

2017

With an increasing demand of sensing applications, energy has been one of the top concerns in wireless sensor networks. Most of the previous works study energy conservation to extend network lifetime and a variety of schemes have been proposed that can elongate network lifetime to some extent. However, with limited energy storage, sensor's battery would deplete eventually and replacing those batteries requires tremendous human efforts. My dissertation investigates a novel approach to replenishing sensor's battery via wireless charging without wires or plugs. We start with a complete overview of the recent developments in wireless charging technologies and their applications in wireless sensor networks to highlight their features and capabilities. A *mobile charger* (MC) is adopted and we call these networks *Wireless Rechargeable Sensor Networks (WRSNs)*. Then we address several important issues and propose a suite of algorithms to guarantee perpetual

operation of the network. First, we discuss several principles from theoretical aspects for perpetual operation. To guarantee that the recharge decisions are made based on accurate information, we then consider the problem of how to gather energy information from the network efficiently. A distributed, on-demand communication protocol is proposed. Based on the energy information collected, recharge scheduling algorithms are developed to minimize the moving cost of MCs. Second, due to physical limits, an MC can only recharge one sensor at a time. We explore the feasibility of multi-hop wireless charging via resonant repeaters and demonstrate tremendous performance improvements. A new recharge scheduling algorithm based on multi-hop wireless charging is proposed. Finally, we exploit the combination of wireless energy with renewable environmental energy for extra cost savings. In particular, we propose a network that relies on hybrid energy sources (both wireless and solar). We further consider a set of interesting problems such as solar-powered sensor deployments, energy re-balance clustering and recharge/data gathering in a joint tour. A complete network performance evaluation is presented in various criteria such as non-functional node percentage, network latency, energy overhead, etc.

To my wife and parents, the anchors in a nomadic life.

Contents

List of Figures

List of Tables

Acknowledgements

First, I would like to express my sincere gratitude to my advisor Prof. Yuanyuan Yang for her guidance and continuous support in this odyssey. She intrigued my interest and led me into networking research. From a naive student to a junior researcher, she conveyed a spirit of adventure in regard to research and scholarship. Her everlasting passion and indomitable perseverance for basic research became the ultimate source of power for me to explore the unknown unknowns. 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 endeavor.

Second, I would like to thank Prof. Fan Ye for his valuable advice, time and effort. Without his persistent help, this dissertation would not have been possible.

Third, I would also like to thank my defense committees, Prof. Sangjin Hong and Prof. Samir Das. Thanks for their precious time and useful suggestions to improve my dissertation quality.

I am also grateful to my labmates: Prof. Songtao Guo, Prof. Miao Zhao, Dr. Zhiyang Guo, Dr. Dawei Gong, Prof. Zhemin Zhang, Ji Li, Jun Duan, Zhenhua Li, Yang Yang and Pengzhan Zhou in the Mobile Computing Laboratory for all their helps. I would also like to express my gratitude to the staffs in the department, Rachel Ingrassia, Tony Olivo, have been especially helpful.

Finally, my special thanks goes to my wife, who has been with me through thick and thin. I am forever indebted to my parents for their unyielding love, endless patience and great encouragement. The achievements of my dissertation are not possible without their tremendous support.

Publications

Book/Book Chapters

- **Book Chapters**: Wireless Power Transfer Algorithms and Applications in Ad hoc Communication Networks, Springer, 2016.
- **Book**: Y. Yang and C. Wang, Wireless Rechargeable Sensor Networks, Springer, 2015.

Journal Publications

- C.Wang, J. Li, Y. Yang and F. Ye, "Combining Solar Energy Harvesting with Wireless Charging for Hybrid Wireless Sensor Networks", *IEEE Transactions on Mobile Computing*, under review.
- C. Wang, J. Li, F. Ye and Y. Yang, "A Novel Framework of Multi-hop Wireless Charging for Sensor Networks using Resonant Repeaters", *IEEE Transactions on Mobile Computing*, 2016.
- C. Wang, S. Guo and Y. Yang, "An optimization framework for mobile data collection in energy harvesting wireless sensor networks", *IEEE Transactions on Mobile Computing*, 2016.
- C. Wang, J. Li, F. Ye and Y. Yang, "A mobile data gathering framework for wireless rechargeable sensor networks with vehicle movement costs and capacity constraints", *IEEE Transactions on Computers*, 2015.
- S. Guo, Y. Yang, C. Wang, "DaGCM: A concurrent data uploading framework for mobile data gathering in wireless sensor networks", *Transactions on Mobile Computing*, 2015.

- S. Guo, C. Wang and Y. Yang, "Joint mobile data gathering and energy provisioning in wireless rechargeable sensor networks", *IEEE Transactions on Mobile Computing*, 2014.

- M. Zhao, Y. Yang, C. Wang, "Mobile data gathering with load balanced clustering and dual data uploading in wireless sensor networks", *Transactions on Mobile Computing*, 2014.

- C. Wang, J. Li, F. Ye and Y. Yang, "NETWRAP: An NDN based real-time wireless recharging framework for wireless sensor networks", *IEEE Transactions on Mobile Computing*, 2013.

Conference Publications

- C. Wang, J. Li, Y. Yang and F. Ye, "A hybrid framework combining solar energy harvesting and wireless charging for wireless sensor networks", *IEEE INFOCOM*, San Francisco, CA, 2016, Acceptance Rate: 18%.

- C. Wang, J. Li, F. Ye and Y. Yang, "Improve charging capability for wireless rechargeable sensor networks using resonant repeaters", *IEEE ICDCS*, Columbus, OH, 2015, Acceptance Rate: 12%.

- C. Wang, J. Li, F. Ye and Y. Yang, "Recharging schedules for wireless sensor networks with vehicle movement costs and capacity constraints", *IEEE SECON*, Singapore, 2014, Acceptance Rate: 28%.

- C. Wang, J. Li, F. Ye and Y. Yang, "Multi-vehicle coordination for wireless energy replenishment in sensor networks", *IEEE IPDPS*, Cambridge, MA, 2013, Acceptance Rate: 22%.

- S. Guo, C. Wang and Y. Yang, "Mobile Data Gathering with Wireless Energy Replenishment in Rechargeable Sensor Networks", *IEEE INFOCOM*, Turin, Italy, 2013 Acceptance Rate 17%.

- (Invited Paper) Y. Yang, C. Wang and J. Li, "Power sensor networks by wireless energy - current status and future trends", *IEEE ICNC*, Garden Grove, CA, 2015.

- G. Gao, C. Wang and Y. Yang, "Joint wireless charging and sensor activity management in wireless rechargeable sensor networks", *IEEE ICPP*, Beijing, China, 2015.

- C. Wang, S. Guo and Y. Yang, "Energy-efficient mobile data collection in energy-harvesting wireless sensor networks", *IEEE ICPADS*, Hsinchu, Taiwan, 2014.

- C. Wang, Y. Yang and J. Li, "Low-latency mobile data collection for wireless rechargeable sensor networks", *IEEE ICC*, London, UK, 2014.

- J. Li, Y. Yang and C. Wang, "Mobility assisted data gathering in heterogeneous energy replenishable wireless sensor networks", *IEEE ICCCN*, Best Paper Run-up, Shanghai China, 2014.

- C. Wang, J. Li and Y. Yang, "Stochastic mobile energy replenishment and adaptive sensor activation for perpetual wireless rechargeable sensor networks", *IEEE WCNC*, Shanghai, China, 2013.

Chapter 1

Introduction

1.1 Motivation

The future of Internet-of-Things relies on the usage of sensors to detect, identify and track objects of interests. With options to mount various types of detectors from pressure, magnetic, chemical, acoustic and seismic sensors to complicated devices such as infrared, video-camera and hyperspectral imaging, sensors have become a bridge between the physical world and cyber space[1, 2]. They begin to play an increasingly important role from our daily lives to many mission-critical tasks. In our daily lives, temperature and humidity sensors deployed indoors that can automatically control the climate. In mission-critical tasks such as volcano or forest fire monitoring [3, 4], sensors are deployed in inaccessible areas for providing information on time. For example, the traditional forest fire monitoring system depends on the analysis of satellite images. However, the accuracy of these systems is usually limited by image quality and weather conditions. Hence, these analysis are usually error-prone and postponed. Sensors equipped with thermal imaging and temperature detectors can be deployed and transmit *real-time* data for fast decision making[4].

As we can see, the increasing demand for more complex sensors leads to higher energy consumption on the sensor nodes. To this end, energy conservation has been one of the primary focuses in *Wireless Sensor Network (WSN)* research in the past decade. Since replacing sensor's battery is infeasible or risky in many applications [3, 4], most of the research aims to maximize network lifetime. For a single node,

duty cycling is one of the most effective methods to save energy [5]. It puts radio transceivers/CPU in sleep mode whenever there is no communication/computation. To adopt this method in a network, wakeup/sleep scheduling of sensors is required to guarantee end-to-end communications [6, 7]. Battery-aware routing and scheduling based on battery recovery property have been studied to extend sensor node lifetime [8–10]. At the network level, researchers have considered to maximize network lifetime by optimizing either flow routing [11] or sensor missions [12, 13].

Essentially, how data is collected also has a great impact on network lifetime. Traditional approaches to aggregating sensed data through a static data sink is known to be less energy efficient since nodes close to the sink consume more energy to relay packets. These nodes usually form a bottleneck around the sink and put an upper limit on the network lifetime while other nodes may still have energy. This is regarded as the infamous "energy hole problem" [14]. A solution is to introduce a mobile data sink for data gathering [15–21]. It has been shown in [20] that by carefully planning trajectory of the mobile sink, energy consumptions on sensor nodes can be balanced and network lifetime is extended significantly.

Although these methods can prolong network lifetime to some extent, sensor's battery would deplete eventually and cause service interruptions. A promising technique is to replenish sensor's battery by harvesting environmental energy such as solar and wind [22–24]. Solar harvesting can provide energy from an external solar panels of similar size to sensor nodes [25]. Further, multiple such ambient energy sources can be combined to power sensor nodes [26]. However, due to the inherent dynamics of energy sources, environmental energy is unpredictable. When energy sources are not available, nodes would stop working and cause network interruptions.

Recently, finding a convenient and reliable way to replenish sensor's battery begins to attract more attentions in the sensor network research community. The breakthroughs in wireless charging technology offers a bright new alternative to power sensor nodes (in distance) without any wires or plugs. Pioneered by Nikola Tesla [27] a century ago, it is only recently wireless charging enjoys so much popularity after the experimental realization by Kurs, et al. [28]. It has been shown in [28] that a total of 60 watts energy can be transferred between two magnetically coupled coils over an air gap of 2 meters with 40% efficiency. Fast development of mobile devices soon as well as the relatively stagnant battery technology deliver

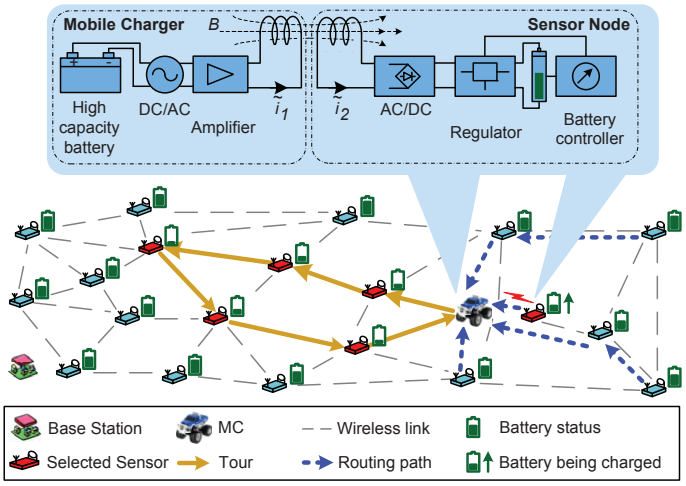


Figure 1.1: Overview of wireless rechargeable sensor network.

the impetus to drive this technology into commercialization stage. Many products are available in the market that can recharge cell phones, PDAs without charging coils. The prototype is soon extended to power multiple devices in [29]. For example, a charging pad called "Powermat" can recharge multiple cell phones and PDAs simultaneously by simply putting them on the pad [30]. Powercast systems realize wireless charging for sensing devices up to several meters away [31]. This technology has demonstrated not only the strengths to power small portable devices, but also the potentials to recharge Electrical Vehicles (EVs). With the ability to deliver hundreds watts of energy at high efficiency, wireless charging systems can be launched at power stations, parking lots or even beneath road surface to recharge EVs without any physical contact [32].

Based on this new technology, we propose a novel framework comprised of one or more multi-functional Mobile Chargers (MCs) for delivering energy to sensor nodes, gathering sensed data and engaging management activities. Fig. 1.1 shows an overview of the wireless rechargeable sensor network. The MC equips with high-capacity battery packs, a DC/AC (Direct Current/Alternating Current) converter and resonant coils. To deliver energy, the MC converts the energy stored in its battery into alternating current using a DC/AC converter when it moves into close proximity of sensors. Then an oscillating magnetic field is induced around the transmitting coil of the MC. The receiving coil on the sensor is tuned to resonate at the same frequency to capture energy from the magnetic field and utilize the

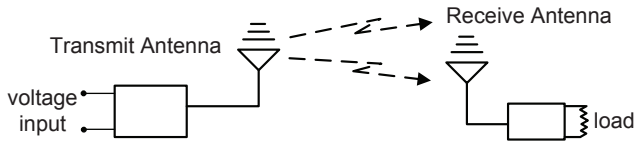


Figure 1.2: Electromagnetic radiation based wireless charging.

AC/DC converter to recharge its battery with DC current. To achieve temporal efficiency, the MC can perform other activities such as data gathering, network management while wireless energy replenishment is in progress. Next, we present some background information on wireless charging technologies.

1.2 Wireless Charging Technologies

In this section, we introduce two major techniques of wireless charging: *electromagnetic radiation* and *magnetic resonant coupling*, and their applications in WSNs.

1.2.1 Electromagnetic Radiation

Electromagnetic waves have been used for communications since the last century. Recently, upon discovering the energy resides in the electromagnetic waves can be captured to power ultra-low power devices, a great amount of research efforts have been devoted to scavenge energy in the ubiquitous electromagnetic waves. There are plenty of such energy sources such as TV towers, cellular base stations or even local Wi-Fi access points. However, due to the nature of isotropic wave propagation and multi-path fading, received signal strength decreases dramatically with transmission distance. Thus only a very small fraction of energy can be effectively captured from the air. Fig. 1.2 shows a sketch of an electromagnetic radiation based wireless charging.

A popular commercial product currently available on the market is the Powercast wireless charging system [31]. It consists of a wireless energy transmitter operating at 850-950 MHz and a number wireless energy receivers. There have been some previous works on applying such systems in WRSNs. In [33], the impact of wireless charging on current routing and node deployment schemes in WSNs is

studied. In [34], the problems of how to place and mobilize wireless chargers to sustain network operations are studied. First, a point provisioning problem is proposed to ensure all the locations in the network can receive enough energy. Then a path provisioning problem is studied to further reduce the number of wireless chargers. The problem is extended in [35] to minimize charging delay by optimally planning the moving trajectory of mobile chargers. In [36], an $\mathcal{O}(k^2k!)$ (where k is the number of nodes in the network) algorithm is designed to schedule recharge activities such that network lifetime is maximized. In [37], a joint routing and wireless charging scheme is proposed by guiding routing and recharge activities. In addition, problems in WRSNs other than recharge scheduling are studied in [38, 39]. In [38], an important safety issue of using electromagnetic radiation based wireless charging is studied. Since absorption of overdosed electromagnetic radiation poses great risks to human body, a placement problem on how to place wireless chargers to sustain network operations while the radiation level of all positions is below a threshold is studied in [38]. Other than the safety issue, in [39], it is shown that traditional localization strategies in WSNs can be further improved by measuring the wireless charging time of sensors.

As pointed out in [38], a limitation of electromagnetic radiation based wireless charging is due to health concerns. Although it is desired to increase the emitted energy at the power source, the Federal Communication Commission's (FCC) has a regulation of maximum effective isotropic radiated power (EIRP) at 4W [40]. In addition, the isotropic nature of omni-directional antenna emits energy that attenuates quickly over distance. Therefore, this technique usually has very low efficiencies and only supports low-power sensing applications such as simple temperature, humidity monitoring, etc. Therefore, in this dissertation, we mainly focus on wireless charging based on magnetic resonant coupling described next.

1.2.2 Magnetic Resonant Coupling

In contrast to the low-efficiency in electromagnetic radiation based wireless charging techniques, magnetic resonant coupling can transfer a large amount of energy over an air gap at high efficiency [28, 29]. Fig. 1.3 shows a wireless charging system with magnetic resonant coupling. To guarantee high charging efficiencies, a mobile charger with high-density battery packs is usually adopted to approach

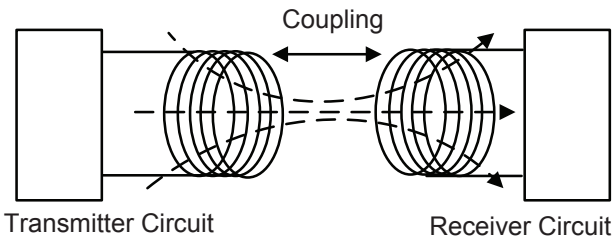


Figure 1.3: Magnetic resonant coupling based wireless charging.

sensors in close proximity.

The potentials of using magnetic resonant coupling in WRSNs is studied in [41–45]. In [42], an optimization problem to maximize the ratio between MC's idling and working time is studied. A Hamiltonian cycle through all the sensor nodes is proved to be the shortest recharge path. Instead of recharging all the nodes, in [41], only a number of nodes request for recharge are serviced, and this number is upper bounded by a tour length threshold to guarantee data latency. During recharge, the MC simultaneously gathers data from the neighborhood in multi-hops and uploads all collected data to the base station after a recharging cycle is completed. A system-wide optimization is performed to maximize network utility by selecting optimal data rates and flow routing. In [44], optimal allocation of MC's stopping time to recharge sensors at different locations is studied. Upon realizing the dynamics in sensors' energy consumptions, to provide more accurate recharge decisions, a recharge framework is proposed in [45]. An NP-hard problem to minimize the movement cost of MCs is studied and several heuristic algorithms are proposed. In sum, magnetic resonant coupling is a promising technology ready to support many complex, energy-demanding multimedia applications with enormous data communication and sensing activities.

1.2.3 Research Goals

This dissertation aims to achieve the following research goals.

Perpetual Operation: Our ultimate goal is to achieve perpetual operation of the network. That is, no sensor would deplete its energy during the run. This requires the MCs to fulfill energy requests on time. We can see that it poses great research challenges in dynamic network environments, in which the patterns of

energy consumptions may vary. The MCs have to schedule and coordinate their recharge activities to make sure the battery deadlines from sensors are met.

Minimal System Cost: Another important research goal is to minimize system cost. It mainly consists of manufacturing cost of the MCs, wireless-powered or solar-powered sensors, MC's operating cost and extra cost incurred such as computation, communications on either sensors or the MCs. Our solutions need to bring these factors in the designs of the algorithms for minimal cost or achieve optimality to balance different types of costs.

Bounded Latency: Our approach needs to achieve bounded latency for data gathering. Traditional approaches adopts the MCs for data gathering. Since mobile data gathering is usually time-consuming (delay is governed by MC's moving/recharging time), we should develop new schemes to minimize such latency or provide an upper bound of packet delay for different applications.

Distributed Operations: Although the computation-intensive tasks can be executed by the MCs and disseminate the decisions to sensors, we still aim to achieve distributed designs of protocols. For example, sensors can communicate between each other for clustering and sharing energy information. Some of these operations can be done at the lowest sensor level to adapt dynamics of the network and save computation resources.

1.2.4 Contributions

In this dissertation, we have made the following contributions.

Recharge Scheduling of Mobile Chargers based on Real-time Energy Information We propose a novel real-time recharging framework for wireless sensor networks, consisting of a set of scalable and efficient energy aggregation and gathering protocols. The protocols satisfy both normal and emergency recharging needs for multiple MC. We identify that the emergency recharge optimization with multiple MCs is an Orienteering Problem and we formalize the normal recharge problem into an m-TSP problem. Efficient on-line algorithms with low computation complexity are proposed. Extensive evaluations demonstrate our algorithms can achieve perpetual operation of the network effectively.

Mobile Data Gathering and Recharge Scheduling with Mobile Charger's Movement Costs and Capacity Constraints We further study limitations in the

existing works on important issues of data latency, MC's moving cost, recharge capacity, and their impact on existing recharge scheduling algorithms. We establish a mathematical model to quantify the relationship between data latency and the number of MC needed. We also present several theoretical results such as node lifetime and adaptive recharge thresholds. Second, we formulate recharge optimization into a Profitable Traveling Salesmen Problem with Capacity and Battery Deadline constraints, and propose two algorithms. The Adaptive Algorithm takes a systematic approach to capture all constraints in the problem. Finally, we conduct extensive simulations comparing the two proposed algorithms. Although we are not able to prove approximation bounds for the Adaptive Algorithm theoretically, simulations show that it is only 1.06 to the optimal solutions and saves an additional 8% on vehicle's moving energy compared to the on-line algorithm. Moreover, when the number of MCs is sufficient, the Adaptive Algorithm can keep all the nodes alive at all times. Compared to the Greedy Algorithm, the Adaptive Algorithm can reduce nonfunctional nodes by 30-50% while saving 10-20% energy on MCs. We validate our theoretical results and justify the system cost, data latency of our framework compared to other schemes.

Multi-hop Wireless Charging via Resonant Repeaters We adopt resonant repeaters to improve charging capability based on realistic modeling of charging efficiency under physics laws and formulate recharge scheduling into a bi-objective optimization problem. A two-step approximation algorithm with bounded approximation ratios for each objective is proposed. In addition, We discover the subtle relations between cost objectives and propose a post-optimization approach to further reduce the system cost while retaining nodes' battery deadlines. Our evaluation shows that the post-optimization algorithm can reduce the system cost by an additional 25% and the proposed framework can cover more than 3 times of nodes and has significantly less service interruptions compared to previous works. We also demonstrate trade-offs between multi-hop and single-node recharging methods, and relations between different optimization objectives.

Combine Wireless Charging and Solar Energy Harvesting We propose a hybrid framework to overcome the constraints of wireless charging and environmental harvesting techniques and formulate the solar-powered sensor placement problem into a facility location problem. Both centralized and distributed algorithms are proposed with bounded approximation ratios. Then, we propose a method to main-

tain network robustness by reducing energy consumption and a route improvement algorithm that can save an extra 25% moving energy on MCs and surpasses the algorithm in [120] by additional 5%. The algorithm can also be used in a general setting for the Traveling Salesmen Problem with Neighborhood (TSPN) and provide solutions very close to the exact solutions found by exhaustive search. We also give MCs more flexibility to only partially refill the battery in case of high energy demand and propose an efficient algorithm that yields solutions within 5% to optimality. Finally, we conduct extensive simulations to evaluate the performance of the framework compared to WSNs that are solely wireless-powered.

1.2.5 Dissertation Outline

The rest of this dissertation is organized as follows. Chapter 2 lays the foundations of our analysis by proposing an energy information gathering protocol and on-line recharge algorithms. Chapter 3 establishes a mathematical model based on energy balance in the network and considers MC's moving cost and recharge capacities in recharge activities. Chapter 4 studies multi-hop wireless charging to improve charging capabilities. Chapter 5 presents a new framework of hybrid energy sources and Chapter 6 concludes this dissertation.

Chapter 2

Recharge Scheduling of Mobile Chargers based on Real-time Energy Information

This chapter presents a real-time energy monitoring and recharging framework that optimizes the recharging policies for one and multiple MCs under dynamic network conditions. The recharging policy - when and which MC should recharge which nodes and in what order - critically impacts the efficiency and thus the lifetime of the network. So far only a few works [41, 42] have studied the recharging policy problem. Basically, nodes report their energy levels periodically, and a centralized algorithm computes a specific order so a single MC recharges all nodes in the next cycle. Although commendable first steps, they do not fully consider important practical issues, which significantly limit their applicability in a real environment.

First, it takes nontrivial (e.g., 30-60 min) time to recharge a commercial off-the-shelf battery, such that finishing one round of recharging for a network of a few hundred nodes may take several days. During this time the energy levels of nodes may have changed significantly due to unpredictable external events that can trigger extensive activities and quickly drain the battery. The recharging policy computed at the beginning of the cycle is no longer optimal. This can cause energy depletion on some nodes, leading to network disconnection or application failures. Second, the timely, efficient and scalable gathering of energy information of nodes to a mobile vehicle is an important and challenging issue in itself. The previous works

do not consider this issue and assume such information is readily available. Finally, they use centralized algorithms that have high complexity and may not scale to large network sizes. A distributed solution is more desirable in real network settings.

Instead of nodes reporting their energy levels only after a long period[41, 42], a scalable and efficient energy information aggregation protocol gathers battery levels continuously from all sensor nodes upon requests by MC. The MC receives such information and makes (on-line) recharging decisions based on the latest energy information. To deal with unpredictable emergencies where nodes may dramatically drain the battery in short time, the recharging of sensor nodes whose energy levels are below a critical threshold has higher priority and takes precedence over those that can work for relatively long time with their residual energy.

The rest of this chapter is organized as follows. Section 2.1 describes the basic network components. Section 2.2 presents a distributed protocol for node status reporting and Section 2.3 develops on-line recharge scheduling algorithms for different scenarios. Section 2.5 provides simulation results and Section 2.6 concludes this chapter.

2.1 Network Components

We assume that sensor nodes are uniformly and randomly distributed in the network. Nodes are stationary and each node knows its deployed location. For scalable performance, the network is divided into several *areas* and each area is further divided to generate some new *sub-areas*. A new level is generated in each division. The divisions are based on geographical coordinates of the sensing field. An example of a 2-level WRSN network is shown in Fig. 2.1. The two areas represented by solid lines are generated at the first level. Then each area is further split into two sub-areas represented by dashed lines on the second level. Several key network components are explained below.

- *Mobile Charger*: The MCs have positioning systems (GPS) and know their locations. The sensor locations are pre-processed during network initialization and known to the MCs. The MCs are equipped with high density battery packs and charging coils. They also have communication capability by launching powerful antennas. In this way, they can not only query the net-

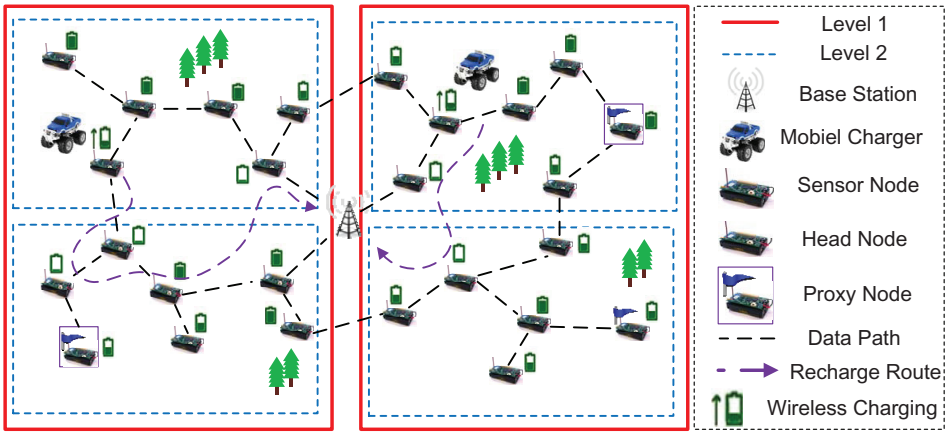


Figure 2.1: Network Architecture.

work for node status information but also communicate among themselves or to the base station via long range communication technologies (e.g., cellular, WiMax).

- *Base Station*: The base station is used for collecting sensing data and performing network management. The MCs can be commanded remotely by the network administrator via the base station. It also has computing capabilities to perform the tasks of calculating recharge sequences and dispatching MCs. When an MC almost depletes its own energy, it returns to the base station for a quick battery replacement.

- *Head Nodes*: A head node is a sensor node that aggregates node's status information in its subordinate area. When requested by an MC or the head node of its superior level, it aggregates node status information from the subordinate sub-areas at the lower levels and sends to the requester.

- *Proxy Nodes*: An emergency occurs when a node's battery energy falls below a threshold (e.g., 10%). It needs to be handled by the MCs immediately. The head nodes on the top-level are selected as proxies so they can aggregate emergency information from sensor nodes directly without propagating through the network hierarchy.

- *Normal Nodes*: A sensor node not selected as a head is a normal node. It reports its status information to its superior head node, or sends emergency

information directly to its proxy when the battery energy drops below the recharge threshold.

The network has a number of m MCs. Once the voltage at the sensor's output circuit is enough to provide a charge, the recharge time is governed by battery characteristics. The typical recharge time required to bring battery energy from zero to full capacity is T_r time (e.g., for a Panasonic Ni-MH AAA battery [46] of battery capacity C_s=780 mAh, T_r = 78 minutes). Note that how to determine the number of MCs m will be further discussed in the next chapter.

2.2 Distributed Node Status Reporting Protocol

This section introduces a distributed communication protocol that can gather node status information from the network on-demand. To guarantee scalability, it first elects a head node on each network level and establishes routes to the head nodes. Since the MCs' locations are constantly changing, the protocol allows the them to send out recharge requests that naturally reflect their current locations and build routing paths for the status packets. The protocol categorizes recharge missions according to their time urgency and aggregates node status efficiently at the MCs.

2.2.1 Overview

To perform effective recharge and maintain network operations, MCs should obtain global node status information of sensors. This information includes residual battery energy, node lifetime, identification, location, etc. Since sensors do not keep track of MC's locations during operations, a trivial way is to flood the network with status packets periodically. However, for a network with N nodes, $\mathcal{O}(N^3)$ packet transmissions might be needed in the worst case. This is because that the number of edges in a completely connected graph is $\frac{N(N-1)}{2}$ and there are N status packets from different nodes on all the edges. Apparently, the cost becomes prohibitive for any network contains more than a few hundreds of nodes. Indeed, for each recharge maneuver, the MC only picks a small subset of nodes with immediate energy demands for recharge, status information from other regions could be regarded as useless. If the useless information can be filtered out before reported to the MC,

a great amount of communication overhead can be saved. Therefore, we introduce a real-time communication protocol for node status gathering in the network.

The MCs obtain the real-time node status information before making any recharge decisions. Node status information is aggregated on *head* nodes at different levels. For robustness, the head node is usually elected with the maximum battery energy in its subordinate area. The head election process is initiated in the network startup phase through propagation of *head election* packets. During the operation, when a head node is low on energy, it will appoint another node with high energy in its area, and send out a *head notification* packet to notify the new head node. The details will be discussed in the next subsection.

To start the information gathering process, MCs send out *status request* packets to poll the *head* nodes on the top-level first. Once the head nodes receive such packets, they generate new *status request* packets for the lower level head nodes in respective subordinate areas. This process repeats down the network hierarchy until the bottom-level *status request* packets reach all the nodes in the bottom-level subareas.

Once a sensor node receives a bottom-level *status request*, it responds by sending out a *status* packet that contains its current energy level, estimated lifetime, identification and position, etc. When the bottom-level head nodes receive such *status* packets, they select sensor nodes with energy level below their corresponding recharge thresholds, and forward their status information in a combined *status* packet to their superior head nodes. This process repeats from the bottom up along the hierarchy until the top-level head nodes successfully aggregate all the status information from designated areas. This information is then sent to the requested MC. In the case that there are more than one MC send out such request simultaneously, the top-level head nodes send the aggregated node status information to the one with fewer communication hops. For overhead reduction, the head nodes take partial responsibilities to pre-select nodes for recharge. On the bottom level, the head nodes only report those nodes with energy level below the threshold.

Once a node's energy falls below an emergency threshold (e.g., 10% of full capacity), without waiting for the MCs to send out request, it preemptively transmits an *emergency* packet to the proxy node that manages its area. The route from each node to its proxy is established by *head election* messages from the proxy and updated during the operation accordingly. Once an MC finishes recharging a node,

it sends out an *emergency request* packet to see whether there is emergency. These packets are directed to the proxy nodes where updated emergency lists are stored and they respond by sending back identifications, lifetime estimations and energy levels to the MC. The MC receives this packet and adopts an appropriate recharge scheduling algorithm to decide the recharge sequence.

The mechanism in the head election protocol shares some similarities with [47, 48]. In the following, we describe details of the new protocols for communication between head nodes on different levels.

2.2.2 Protocol Design

We describe the protocol design in this section for a network with l levels.

Head Election

At the initialization phase, the network performs head election starting from the bottom l-th level and this process is propagated up to the top level. Each node generates a random number x and compares it with a pre-determined threshold K. If $x > K$, it floods a *head election* packet in its subarea at the l-th level. The packet contains the random number x and its identification. Then the node sets it as its maximum random number at its local record $x_{max} = x$. Otherwise, if $x \leq K$, the node waits for receiving packets from other nodes.

Upon receiving a *head election* packet, a node first compares the random number field in the packet with its local record x_{max}. If its local record is larger, the packet is discarded. Otherwise, the sensor updates x_{max} to that in the packet accordingly and records the identifier in the packet. Then it sends out the packet to all its neighbors except the one where packet is received from. This process can be regarded as a distributed fashion to elect the node with the maximum x in each subarea on the bottom level.

On the $(l-1)$-th level, the newly elected head nodes compete for the heads on this level following a similar manner. They flood new *head election* packets in their subareas on the $(l-1)$-th level. Nodes follow the same procedure to compare the received random number x and finally the head nodes are elected. This process is repeated until the heads on all the levels are elected.

To build intermediate routing information from each node to its head, the head election packets that do not succeed in the comparison are not discarded except for the bottom level. Instead, they are propagated throughout the respective subarea. This ensures the intermediate nodes to know the routes to the head nodes. Once an upper level head node wants to communicate with its subordinate head nodes, these entries in the routing tables on each intermediate node can be utilized.

Status Request

The hierarchical head structure is constructed to facilitate the propagation of *status request* packets. These packets collect the current status from nodes to offer MCs a global view of the network. The status information is gathered on demand. That is, it can be either sent out after an MC finishes recharging every node or once in a while to reduce communication overhead in the network.

After the head hierarchy is constructed, the MCs send *status request* packets to query nodes that need recharge. Upon receiving such packets, intermediate nodes use the routing tables established during head election process to forward the packets to all top-level head nodes. At the same time, an intermediate node also leaves an entry in its routing table pointing to the neighbor from which the status request packet is received. This entry is used to guide *status* packets back to the MCs. In Fig. 2.2, the propagation status request of a network with two levels is illustrated. After a status request is sent by an MC, status information is converged from the bottom level to the top level and finally delivered to the MC.

After receiving a *status request* packet, a top-level head generates a new status request packet and transmits it to its child-heads. These packets use the routing entries set up during the head election process to find the lower-level head nodes. Similarly, nodes also set up routing entries where these packets are coming from so that later status packets can be aggregated at the upper level heads. This process repeats down the head hierarchy until the bottom level heads are reached. Those heads then flood the *status request* packets in their respective subareas.

It could be the case that two or more MCs are requesting node status simultaneously. To avoid receiving duplicated information, we direct the status packets towards the MC with fewer hop counts. The *status request* packet carries a field to count the hops from the MC, i.e., the field grows by one at each intermediate node.

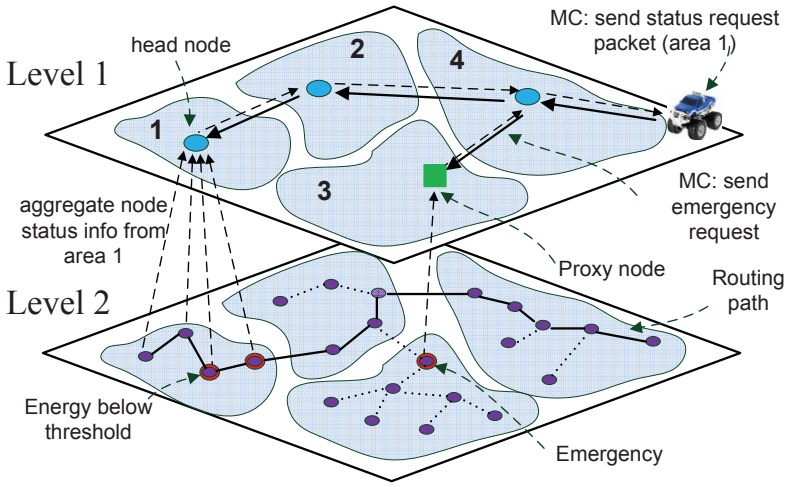


Figure 2.2: Illustrating propagation of different types of packets.

Once multiple *status request* packets are received by a head node, an intermediate node updates its routing entries by recording only the neighbor with the smallest hop count. In this way, status information from a head node follows the route to reach the MC with the smallest hop count. Since the MCs are moving during the operation, these routing entries are updated for each status request.

Status Report and Recharge

Once a node receives a bottom level *status request* packet, it responds with information including its current energy level, estimated lifetime, identification and position. These packets are easily routed back to the bottom level heads based on the routing entries set up earlier. The head nodes quickly check if the reported energy level of a node is less than the node's recharge threshold. If so, the identification of the node is added to a local recharge list at the head node, and the energy demand is also added to a cumulative summation counter. Once the head finishes collecting status packets in its subarea, it sends out an aggregated *status* packet to its upper level head node. The aggregated *status* packet contains the information from nodes with energy below their recharge thresholds. The recharge threshold could be either predetermined or selected adaptively. The details will be discussed in the next chapter.

Upon receiving aggregated status packets from the lower level head node, a

head node always selects the one with the largest cumulative energy demand and forwards it upwards the head hierarchy. Finally, the MC close to a head on the top level receives which subarea has the largest energy demand and proceeds to recharge the nodes based on the recharge algorithms discussed later.

By entitling the head node some responsibilities to filter out some sub-areas, communication overhead can be minimized during the process of gathering node status. This is important since node status information is gathered every once in a while, redundant information would not only enlarge the packet length but also increase the computation complexity of recharge schedules.

Fig. 2.2 gives a pictorial illustration of a network with two levels. The MC sends out a status request to poll all the node status information from area 1. The energy request packet is relayed towards the head node in area 1 by nodes in areas 2 and 4. Upon receiving the energy request, the head node in area 1 aggregates node status in its area and reports to the MC. The packet is routed back following the same route taken by the status request packet.

Emergency Report and Recharge

Emergency occurs when a node's energy falls below the emergency energy threshold. These nodes should be taken care immediately to prevent them from depleting battery energy. Once an emergency is detected, the node immediately sends out an *emergency* packet with its identification and energy level to the proxy node in its area. The proxy nodes are top level head nodes so the emergency packets do not need to propagate through the head hierarchy. The routing information established earlier during the head election process can be used to direct these packets towards the proxy nodes.

The MC should frequently check whether there is emergency situation by polling the proxy nodes through *emergency request* packets. In principle, to avoid any missing emergency, the MCs should send out such packets after finishing recharging the current node. Once an intermediate node receives an *emergency request* packet, it updates the local routing entries to record where this packet is coming from. This entry is used to route the emergency report packets from the proxy nodes back to the MCs. Since there could be multiple emergency nodes reported while there are also other normal recharge requests, an MC needs to handle all the emergency situations

within a specified time (e.g., the expected time before next emergency occurs). We introduce several recharge scheduling strategies in the next chapter.

Fig. 2.2 also shows an example with a node having emergency in area 3. The node immediately reports to the proxy node and the packet is further forwarded to the MC upon an emergency request.

Head Hierarchy Maintenance

A head node may run out of energy since it usually engages in more activities than other nodes. In this situation, head re-election is needed. In fact, only the head nodes on the bottom levels compete with each other for the head node on an upper level. Since a head node receives status report from all the nodes in its bottom level subarea, it knows the updated node status in its subarea. To reduce overhead, it can easily appoint the node with the highest energy as the *new head node*. A *head notification* packet is then flooded in the bottom level subarea to notify all the nodes of the new head node.

The generation of the new head triggers a new head election process up the head hierarchy. It floods a new *head election* packet in its subarea. Instead of a random number, the packet carries the current energy level of the participating head node. Following the same procedure, nodes in the subarea compare the energy level in the incoming packet and only store the information with the maximum energy. Then the head node with the highest energy level is elected. If this is the same head node, the process stops to avoid unnecessary overhead. Otherwise, the new head triggers a sequence of head election in the upper level and this process repeats until a new head node is elected on the top level.

2.3 Recharge Scheduling Problem

This section discusses the important recharge scheduling problems. Our primary objective is to find a recharge policy with minimal energy cost on the MCs while maintaining the perpetual operation of the network. We first discuss the *emergency recharge scheduling problem* and provide an efficient algorithm to solve it. Then we investigate the normal recharge problem by formulating it into an optimization problem, which is NP-hard. We present an on-line algorithm for this problem

which leverages a weighted sum of node lifetime and MC's traveling time to make recharge decisions.

2.3.1 Emergency Recharge Scheduling Problem

First, we discuss the optimal recharge policy to handle multiple emergencies. According to Section 2.2.2, a node that is on the verge to deplete its battery energy will send an emergency recharge request to the proxy node on the top level. In addition, when the MC is idle, it polls the proxy node to obtain an emergent recharge node list if there is any. Here, we consider the scenario where there are n emergent nodes to be recharged in T_e time. T_e is defined as the average inter-arrival time of emergencies during operations. The value of T_e can be measured and updated iteratively through the operation by MCs. We assume that the sum of their energy demands is much less than the recharging capacity of the vehicle.

Since the MC may not finish recharging all n nodes within T_e time, our objective is to maximize the amount of energy refilled into the network in T_e. The problem can be formulated as a classic Orienteering Problem (OP) [49]. OP involves a set of points in the field with different rewards to be visited by a player before time expiration. The objective is to maximize the rewards collected before the time expires. To model OP into our problem, the MC visits sensor nodes for maximizing energy replenishment (reward) within inter-arrival period of emergency T_e. We consider a graph $G = (V, E)$ where vertex V_i represents the emergent sensor locations. The MC starts from the original location V_0. E is the edges among sensor nodes. The recharging reward r_i of sensor i is defined as the amount of energy replenished from the current energy level to full capacity. The edge cost is defined to be the traveling time t_{ij} between i and j plus the recharge time of node i (denoted as t_i). In order to be consistent with the original OP formulation, we virtually make the MC return to the starting location after T_e by adding an edge of zero weight, i.e., the traveling time is $t_{i0} = 0$. A decision variable x_{ij} for edge e_{ij} is introduced. $x_{ij} = 1$ if the edge E_{ij} is visited, otherwise, it is 0. Variable u_i is defined as the position of vertex i in the recharging path. The emergency recharge scheduling problem is formulated as follows.

$$\mathbf{P1}: \quad \max \sum_{i=1}^{n} \sum_{j=1}^{n} r_i x_{ij}, \tag{2.1}$$

Subject to

$$\sum_{i=1}^{n} x_{0i} = \sum_{i=1}^{n} x_{i0} = 1, \tag{2.2}$$

$$\sum_{i=1}^{n} x_{ik} = \sum_{j=1}^{n} x_{kj} \leq 1, \forall k = 1, 2, \ldots, n \tag{2.3}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} (t_{ij} + t_i) x_{ij} \leq T_e, \tag{2.4}$$

$$x_{ij} \in \{0, 1\}, \forall i, j = 1, 2, \ldots, n, \tag{2.5}$$

$$1 \leq u_i \leq n, \forall i = 2, 3, \ldots, n, \tag{2.6}$$

$$u_i - u_j + 1 \leq n(1 - x_{ij}), \forall i, j = 2, 3, \ldots, n. \tag{2.7}$$

Constraint (2.2) guarantees that the recharging path starts from starting position 0 and ends at starting position 0. Constraint (2.3) ensures the connectivity of the path and that every node is visited at most once. Constraint (2.4) makes sure that the time threshold T_e is not exceeded. Constraint (2.5) imposes decision variable x_{ij} to be 0-1 valued. Constraints (2.6) and (2.7) eliminate subtours in the planned route. These subtour elimination constraints are formulated according to [50, 57].

If time T_e is set to infinity, OP is reduced to the classic Traveling Salesmen Problem with Profit which is known to be an NP-hard problem [51]. Therefore, adopting heuristic algorithms can achieve a balance between performance and computation complexity. A few algorithms have been proposed in [52–55] and a survey of the problem is available in [49]. Tsiligirides [52] has developed a stochastic Monte Carlo technique to generate a large number of routes and used the divide-and-conquer method to select the best among them. A center-of-gravity heuristic algorithm is proposed in [53]. Another algorithm consisting of five steps is proposed in [54]. Optimal solutions to the OP using the brand-and-cut method is introduced in [55]. However, these algorithms are quite complex in terms of efficiency and computational time. Given energy restrictions in the network and the urgency to resolve the emergent nodes, a fast and efficient algorithm is more desirable in the context of our problem.

Next, we show OP can be approximated into a Knapsack problem [56]. The Knapsack problem aims to maximize the value of items into a knapsack with limited size. Each item is associated with a known size. In fact, the recharge time of a node

i is much more than the traveling time from vehicle's current location k to i (i.e., $t_i \gg t_{ki}$). For example, replenishing a node to full capacity usually takes around an hour, the traveling time only takes a few minutes at most. Therefore, to maximize the amount of energy replenished within T_e, we can focus on the recharge time of node i. Thus, Constraint (2.4) in the original OP formulation can be rewritten as $\sum_{i=1}^{n} t_i y_i \leq T_e$. Here, recharge time t_i corresponds to the item size and recharge reward r_i is the item value in the Knapsack problem respectively. With this reduction, we have a much simpler formulation.

$$\mathbf{P2}: \quad \max \sum_{i=1}^{n} r_i y_i, \tag{2.8}$$

Subject to

$$\sum_{i=1}^{n} t_i y_i \leq T_e. \tag{2.9}$$

Although Knapsack problem is known to be NP-complete [56], we can solve it in polynomial time using dynamic programming techniques. Dynamic programming is a strategy to break down a problem into many recurring small subproblems and solve them in a recursive manner. We define a table R with entry $R(i,t)$ to represent the maximum recharging reward attained with total time duration less than t where $1 \leq i \leq n$ and $1 \leq t \leq T_e$. Our goal is to compute every entry in the table towards the maximum value of $R(n, T_e)$. We set all the entries $R(0,t)$ for $1 \leq t \leq T_e$ to zero initially. For all the i and t in the table, if picking a new node for recharge exceeds T_e, the reward remains unchanged $R(i,t) = R(i-1,t)$; otherwise, $R(i,t) = \max(R(i-1,t), r_i + R(i-1, t-t_i))$. The pseudo-code of the algorithm is shown in Table 2.1. As there are two loops of size n and T_e, the complexity of the algorithm is $\mathcal{O}(nT_e)$, which is much lower than directly implementing those algorithms designed for OP.

Finally, it is important to examine the accuracy of such approximation. To see how accurate this approximation achieves in our problem, we use brute force to calculate the optimal solution to OP thereby providing a baseline for comparison. Due to exponentially increasing combinations of larger datasets, we manage to test several cases for n varies from 3 to 16. We define the accuracy as $1 - \left|\frac{R_k - R_{op}}{R_{op}}\right|$,

Table 2.1: Algorithm to approximate Orienteering Problem

Input: T_e, recharge time t_i, table R with entry $R(i,t)$, $1 \leq i \leq n$ and $1 \leq t \leq T_e$
Output: maximum recharge reward and recharging nodes
Initialize $R(0,t) = 0$, $1 \leq t \leq T_e$
For i from 1 to n
 For t from 1 to T_e
  If $t_i \leq t$, $R(i,t) = \max(R(i-1,t), r_i + R(i-1,t-t_i))$
  Else $R(i,t) = R(i-1,t)$
  End If
 End For
End For

Table 2.2: Accuracy of Knapsack Approximations to Optimal Solutions

# Emergencies n	3	4	5	6	7	8	9
$T_e = 300$ min	1	1	1	1	1	1	1
$T_e = 400$ min	1	1	1	1	1	0.997	0.996
$T_e = 500$ min	1	1	1	1	1	1	1
# Emergencies n	10	11	12	13	14	15	16
$T_e = 300$ min	1	1	1	1	1	1	0.998
$T_e = 400$ min	0.999	0.998	0.997	0.995	0.997	0.998	0.997
$T_e = 500$ min	0.996	1	1	1	1	0.995	0.996

where R_k is the solution by Knapsack approximation and R_{op} is the optimal solution by brute force. Table 2.2 shows that the accuracy is over 99% for different T_e.

2.4 Normal Recharge Scheduling

Next, we discuss how to schedule multiple MCs for normal battery recharge. In the process of normal recharge, it is also necessary to prevent nodes in the recharge sequence from depleting battery energy. The objective is to minimize the overall moving cost of MCs while maintaining the perpetual network operation and satisfying a few constraints. The first constraint comes from MC's limited capacity whereas most of the previous works have ignored the moving energy of the vehicle and the limit of its recharge capacity [41, 45]. These simplifications may cause the

MC to deplete energy en route, become stranded and unable to return to the base station. The second constraint is to meet sensors' dynamic battery deadlines. This would require the vehicle to recharge some nodes earlier than others. For example, depending on the size of recharge sequences, some nodes may need *prioritized recharge* to avoid depleting battery energy. How to place these nodes in the recharge sequence to guarantee optimal and feasible solution is an interesting, yet difficult problem. We formalize it into an optimization problem with these constraints and provide two algorithms to tackle the problem.

After each time the node status information is reported to the MCs, a recharge scheduling problem is formed as follows. We denote the set of MCs as $\mathcal{S} = \{1, 2, \ldots, m\}$ and the set of nodes requesting for recharge as $\mathcal{N} = \{1, 2, \ldots, n\}$. Consider a graph $G = (V, E)$, where vertex V_i ($i \in \mathcal{N}$) is the location of node i requests for recharge, and E is the set of edges. During the operation, the vehicles could have different starting positions. We introduce an virtual vertex V_0^a as the starting position of vehicle a. The weight of each edge E_{ij} is associated with the moving energy cost c_{ij}, which is proportional to the distance between nodes i and j. c_{0i}^a represents the cost from initial position V_0^a of vehicle a to node i. Since different MCs might have different energy during the run, we denote the battery energy of MC a as C_a ($C_a \leq C_h$). The value of C_a determines the number of nodes it can recharge before it goes back to the base station for its own battery replacement. The energy demand for node i is denoted as d_i (demand equals a node's total battery capacity minus its residual energy). Each sensor node i has lifetime L_i and A_i is the arrival time of a vehicle at node i. We further introduce two decision variables x_{ij}^a for edge E_{ij} and y_{ia} for vertex V_i. The decision variable x_{ij}^a is 1 if an edge is visited by vehicle a, otherwise, it is 0. The decision variable y_{ia} is 1 if and only if node i is served by vehicle a, otherwise, it is 0. u_i is the position of vertex i in the recharge tour. The objective is to minimize the total moving cost of the MCs while guaranteeing that the recharge capacities of MCs are not exceeded and no sensor node depletes battery energy.

$$\mathbf{P1}: \quad \max\Big(\sum_{a=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}^a + \sum_{a=1}^{m}\sum_{i=1}^{n} c_{0i}^a x_{0i}^a\Big) \tag{2.10}$$

Subject to

$$\sum_{j=1}^{n} x_{0j}^{a} = 1, a \in \mathcal{S}, \tag{2.11}$$

$$\sum_{i=1}^{n} x_{ik} = \sum_{j=1}^{n} x_{kj} = 1, k \in \mathcal{N}, \tag{2.12}$$

$$\sum_{i=1}^{n} d_i y_{ia} + \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} x_{ij}^{a} + \sum_{i=1}^{n} c_{0i}^{a} x_{0i}^{a} \leq C_a, a \in \mathcal{S} \tag{2.13}$$

$$\sum_{a=1}^{m} y_{ia} = 1, i \in \mathcal{N}, \tag{2.14}$$

$$A_i \leq L_i, i \in \mathcal{N}, \tag{2.15}$$

$$x_{ij}^{a} \in \{0,1\}, i,j \in \mathcal{N}, a \in \mathcal{S}, \tag{2.16}$$

$$y_{ia} \in \{0,1\}, i \in \mathcal{N}, a \in \mathcal{S}, \tag{2.17}$$

$$1 \leq u_i \leq n, i \in \mathcal{N}, \tag{2.18}$$

$$u_i - u_j + (n-m)x_{ij} \leq n - m - 1, i,j \in \mathcal{N}, i \neq j. \tag{2.19}$$

In the above formulation, Constraint 2.11 states that the recharge path for each MC starts at an initial position 0. Constraint 2.12 ensures the connectivity of the path and every vertex is visited at most once. Constraints 2.13 and 2.14 guarantee the vehicle's battery energy is not depleted and each sensor is recharged by only one MC. Constraint 2.15 guarantees arrival time of an MC is within each sensor's lifetime. Constraints 2.16 and 2.17 impose x_{ij} and y_{ia} to be 0-1 valued. Constraints 2.18 and 2.19 eliminate the subtour in the planned routes, which is formulated according to [57]. The problem can be reduced to the classic Traveling Salesmen Problem (TSP) with unlimited recharge capacity and unspecified node's battery deadline. Clearly, since TSP is a well known NP-hard problem [56], the recharge scheduling problem is also NP-hard.

A direct solution to the recharge scheduling problem that accounts for both vehicle's capacity and node's deadline is rare in existing literature due to its hardness. Therefore, we first review some literatures that have partially solved the problem. A similar problem to TSP is the Vehicle Routing Problem (VRP) [58]. In VRP, a fleet

of vehicles start from the same depot and visit client locations to deliver goods. The difference between VRP and TSP is that the salesmen in TSP are allowed to start from different locations whereas vehicles usually start from the same location. In addition, the number of vehicles could be undetermined in VRP and more vehicles can be added in order to meet the demands from clients. The Capacitated Vehicle Routing Problem (CVRP) is studied in [59–61]. In [59], a method is proposed to decompose the problem into a convex combination of TSP tours and the tours are examined if the capacity constraint is violated. In [60], tree-based CVRP is studied and a 2-approximation algorithm is proposed. In [61], exact solutions of CVRP are explored by a combination of branch-and-cut and Lagrangian relaxation methods. Time constraint is also important in many VRPs. For example, a store may only accept goods delivery from 9:00AM-5:00PM during regular business hours. How to schedule the fleet of vehicles to make the deliveries within clients' specified time windows is called Vehicle Routing Problem with Time Windows (VRPTW). The problem is studied in [62–65]. In [62], a local search algorithm is proposed to reduce the computation of checking the feasibility of the time constraint. In [63], a theoretical approach of $3 \log n$-approximation algorithm is sought based on established subroutines (where n is the number of nodes). In [64, 65], a relaxed time constraint that allows late arrivals is considered.

Most of these works adopt standard optimization techniques that are effective for datasets with small size and static inputs. Therefore, the optimization can be done offline by computers with strong computing power. In contrast, the wireless sensing environment is statistical in nature. That is, the inputs of energy request would change for each run and the size of such request could be large. Besides, the MC's energy declines while moving and recharging sensors. The existing solutions cannot handle these dynamic situations. Further, due to limited computing power on the vehicles, it is not cost-effective and efficient to implement algorithms with high complexity. To this end, our objective is to design algorithms that are suitable to the dynamic nature of the recharge scheduling problem.

There are several challenges to solve this complex problem. The first challenge is that the MCs' energy constantly decreases due to moving and recharging sensor nodes. Thus, the recharge route should be built with caution to reflect the vehicle's current energy level and traveling costs to node locations. The second challenge comes from the dynamics of energy consumption due to data transmissions. Some

nodes consume energy at higher rates and have shorter lifetime than others. These nodes usually lie on the main routing path and should be taken care of more frequently than others to maintain the operation of the network. The optimal solution to this problem is between achieving conflicting goals. On one hand, to keep all the nodes running, we need to push the MCs to recharge as many nodes as possible. On the other hand, the desire to reduce overall cost needs to minimize the moving distance of MCs. At the same time, the recharge decisions should account for node's lifetime and vehicle's own battery energy as well. We can see that an ideal solution should achieve a good balance between the two objectives without sacrificing either. In the next subsections, we present two such algorithms.

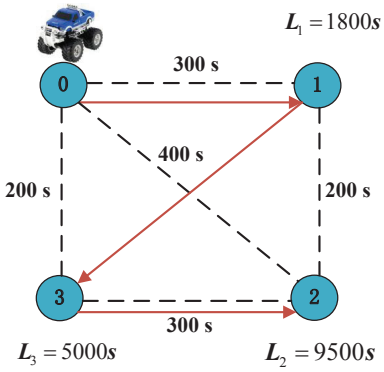
2.4.1 Weighted-Sum Algorithm

First, we present a fast algorithm that leverages the weighted sum of node's lifetime and MC's traveling time. Given an MC's current location at k and two nodes i and j, there are important metrics to affect their orders in the recharge sequence: the traveling time between k to i, j (t_{ki}, t_{kj}), and their lifetime l_i, l_j. If node j is bound to deplete its battery while node i can still last for a while, the vehicle should recharge j first even if j is located further away than i. Therefore, we can see that to maintain perpetual operations, a trade-off has to be made between meeting sensor's battery deadlines and minimizing vehicle's traveling cost. We introduce a weighted sum w_{ij} below

$$w_{ij} = \alpha t_{ij} + (1 - \alpha) l_j. \tag{2.20}$$

For an MC residing at node i, w_{ij} is used to decide which node j to recharge next. A node with a smaller weighted value is more desirable and should be visited with higher priority. The weight parameter α affects the choice of recharging schedules. When $\alpha = 1$, the algorithm reduces to the nearest neighbor algorithm that the vehicle always recharges the closest node first regardless of battery deadlines; when $\alpha = 0$, it picks the node with the earliest battery deadline first regardless of the traveling time. When the vehicle detects its own battery is about to deplete, it returns to the base station for battery replacement.

Fig. 2.3 shows an example of an MC with three sensor nodes. The lifetime and the traveling time on each edge are shown in the figure. For demonstration purpose, we vary α from 0, 0.5 to 1 and assume the recharge takes 3600 s (seconds)



Current time: 0s

Weight	$\alpha = 1$	$\alpha = 0.5$	$\alpha = 0$
w_{01}	300	**1050**	**1800**
w_{02}	400	4950	9500
w_{03}	**200**	2600	5000

Current time: 3900s

$L_2 = 5600\,s$ $L_3 = 1100s$

Weight	$\alpha = 1$	$\alpha = 0.5$	$\alpha = 0$
w_{12}	Infeasible	2900	5600
w_{13}	Infeasible	**750**	**1100**

Figure 2.3: An example of weighted sum algorithm with one MC and 3 sensor nodes.

to finish. At time 0 s, the vehicle calculates the weight for sensor nodes 1, 2 and 3. The minimum weights are circled. When $\alpha = 1$, node 3 has the minimum weight of 200 (purely traveling time); when $\alpha = 0.5$, node 1 has the minimum weight of 1050; when $\alpha = 0$, node 1 also has the minimum weight of 1800. At this point, if node 3 is visited next, node 1 would have depleted its energy after finishing recharge node 3. Therefore, the choice of $\alpha = 1$ is infeasible in this example and node 1 is visited first. After node 1 has been recharged, choosing node 3 results in the minimum weight for both $\alpha = 0.5$ and $\alpha = 0$. Therefore, the recharge schedule follows $0 - 1 - 3 - 2$ in this example.

We can see that the weight parameter α also affects the feasibility of the solution. Since the total distance is not simply inversely proportional to α, we cannot use a binary search method to locate the best α value. To this end, we find α by searching through a list of candidate α values, A. For example, $\alpha = 0, 0.05, 0.1, 0.15, \ldots, 1.0$, where $|A| = 21$. In this way, a desirable trade-off is achieved between optimality and complexity.

While there are multiple MCs calculating the recharge sequence together, they exchange their location information via long range radio communications. Current technologies such as cellular communications and WiMax can easily realize such coordinations. At the beginning of recharge scheduling, an updated recharge node list is synchronized on all the vehicles. We label the vehicles in orders so they begin the calculation of the next node sequentially. After a vehicle selects a node for recharge, it broadcasts its decision so other vehicles can remove this node in

Table 2.3: Recharge Scheduling - Weighted Sum Algorithm

Input: Weight parameter $\alpha \in [0, 1]$ in stepsize $1/(A-1)$,
current position of vehicle at node k,
set of energy requests $\mathcal{N}$, traveling time from i to j, t_{ij}, lifetime $l_i, \forall i, j \in \mathcal{M}$.
Output: Recharge sequence Q.
Initialize minDist $= \infty$, obtain updated recharge node list $\mathcal{N}$, set $Q_t = \emptyset$.
For $\alpha = 0, \ldots, 1$ in an increment of $1/(A-1)$
While $\mathcal{N} \neq \emptyset$
$\forall j \in \mathcal{N}$ Compute $w_{kj} \leftarrow \alpha t_{kj} + (1-\alpha) l_j$.
Find $j \leftarrow \arg\min_j w_{kj}$. Broadcast node j has been selected to other vehicles.
Update its local $\mathcal{N} \leftarrow \mathcal{N} - j$, add j to the end of Q_t,
move to position j for recharge.
Update lifetime of the rest nodes $\forall i \in \mathcal{N}$, $l_i \leftarrow l_i - t_{kj} - t_j$.
If $l_i \leq 0$,
Declare infeasible and inform base station.
End If
End While
If solution is feasible,
compute total cost dist(Q_t).
End If
If dist(Q_t) $<$ minDist,
minDist $\leftarrow$ dist(Q_t), $Q \leftarrow Q_t$.
End If
End For
If the vehicle's battery is about to deplete,
it returns to the base station for battery replacement.

their recharge node list at this point. This operation avoids possible conflicts where multiple MCs select the same node for recharge. Table 2.3 shows the pseudo-code of the entire algorithm.

2.5 Performance Evaluations

In this section, we use simulation to evaluate the effectiveness and efficiency of our framework. We have developed a discrete event-driven simulator using POSIX thread programming in C language. Message communications between sensor nodes are emulated using inter-process communication in our simulator. Our simula-

tor is fully capable of realizing message communication, information convergence, recharge and MC mobility. To evaluate the performance, we examine two network sizes of 500 and 1000 sensor nodes, uniformly randomly distributed over a $200 \times 200m^2$ and $282 \times 282m^2$ square field, respectively. The field size is chosen so that the two cases have the same node density. The network consists of 3-level hierarchy with 4^l number of subareas at the l-th level. The energy consumption on each sensor is a Bernoulli random variable with probability p to consume unit energy (37.5 mJ). If a sensor node works continuously at this rate, the battery can last for 5 days.

To evaluate energy overhead of the protocol, we use the model presented in [70], i.e., $e_t = (e_1 d_r^{\alpha} + e_0)l$, where e_t is the energy consumption while transmitting a message of l bits, d_r is the transmission range, e_1 is the loss coefficient per bit, α is the path loss exponent and e_0 is the excessive energy consumed on sensing, coding, modulations, etc.[1] The relationship between recharged energy and recharge time follows that of Panasonic Ni-MH AAA battery [46]. To understand the impact of the number of MCs on network performance, we show marginal cases where the number of MCs is not sufficient while adding one more MC would guarantee perpetual operations. These cases are $S = 2, 3$ for $N = 500$ and $S = 4, 5$ for $N = 1000$. Fig. 2.4 shows a function from curve fitting of recharge time vs. battery percentage. All the parameter settings in the simulation are listed in Table 2.4.

2.5.1 Evaluation of Weighted-sum Algorithm

In this subsection, we evaluate the effectiveness of the weighted-sum algorithm in finding the shortest path and achieving no node failure. We examine cases when 4 MCs are employed. We assume the locations of emergencies are randomly distributed in the field of $282 \times 282m^2$, and the residual energy uniformly distributed from zero to the emergency threshold. The corresponding residual lifetime is calculated by dividing the residual energy by pr_c, the expected energy consumption in unit time.

Table 2.5 shows the total distance of MCs when the number of concurrent emergencies M increases from 72 to 96 in a step of 8. Note that when the number reaches 96, the set of 4 MCs is not sufficient to resolve all the emergencies without

[1] $d_r = 15$m, $e_0 = 45 \times 10^{-9}$ J/bit, $e_1 = 10 \times 10^{-9}$ J/bit, $\alpha = 2$.

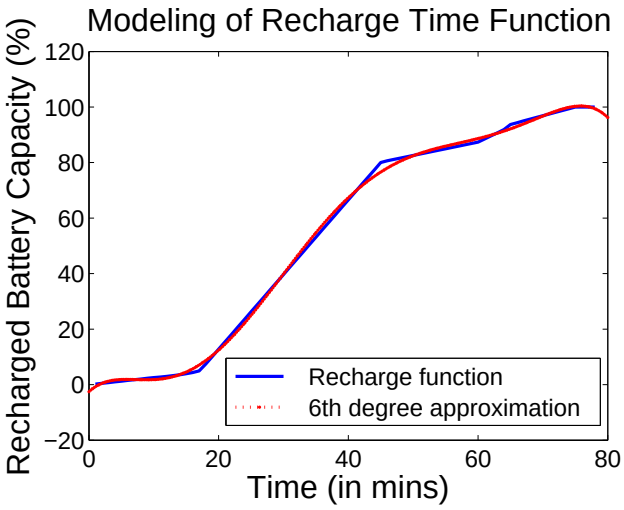


Figure 2.4: Curve fitting of battery recharge time

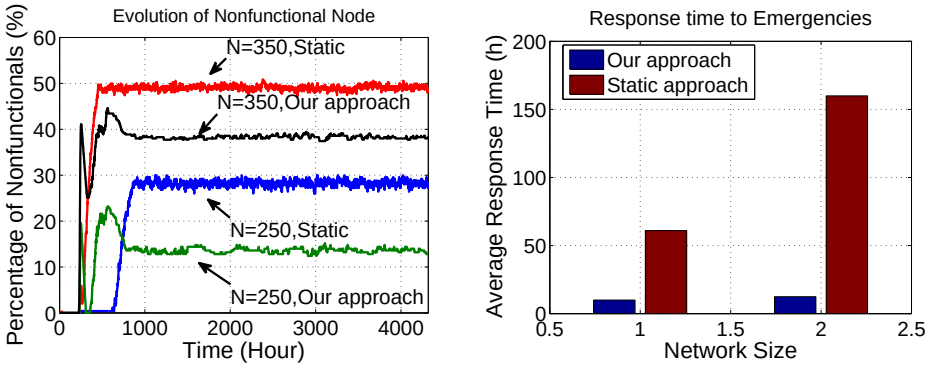
Table 2.4: Parameter Settings

Parameter	Value
Field Length	200×200, $282 \times 282 m^2$
Number of Nodes N	250, 350, 500, 1000
Number of MCs S	1, 2, 3, 4, 5
Number of Levels	3
Areas on l-th level	4^l
Battery Capacity	780 mAh
Transmission Range	18 m
Unit Energy Consumption r_c	37.5 mJ
Energy Consumption Probability p	0.5
MC's Speed	1 m/s
Maximum Recharge Time	78 mins
Normal Recharge Threshold	50%
Emergency Recharge Threshold	10%
Simulation Time	6 months

Table 2.5: Total Traveling Distance of MCs, D

M	D ($\alpha = 0$)	D ($\alpha = 0.2$)	D ($\alpha = 0.4$)
72	7524.1	7473.3	7740.2
80	7652.4	7578.9	7706.6
88	8662.6	8128.3	7251.6
96	Infeasible	Infeasible	Infeasible
M	D ($\alpha = 0.6$)	D ($\alpha = 0.8$)	D ($\alpha = 1$)
72	6843.5	**6390.6**	Infeasible
80	7271.8	**6941.0**	Infeasible
88	**6998.3**	Infeasible	Infeasible
96	Infeasible	Infeasible	Infeasible

complete battery depletion. For $M = 88$, weight parameter $\alpha = 0.8, 1$ are not feasible and for $M = 72, 88$, $\alpha = 1$ is not feasible either. We notice that in the case when $\alpha = 1$, some nodes that suffer from energy shortage may not get recharged in a higher priority thereby rendering the result infeasible to avoid battery depletion. As we can see from this example, the choice of α is critical, when α approaches 1, the total distance is decreased at the risk of becoming infeasible. Thus we need to search for α in our algorithm. In real applications, the value of α is subject to change and determined by real-time statistical data and parameters.



(a) (b)

Figure 2.5: Comparison of static and real-time approaches in terms of (a) Percentage of nonfunctional nodes; (b) Average response time to emergencies.

2.5.2 Performance Comparison with a Static Optimization Approach

In this subsection, we compare network performance of our real-time framework with the static optimization approach used in [41]. Since the static approach has only designed algorithms for a single MC, we set the number of MCs at 1 and compare the percentage of nonfunctional nodes and response time to emergencies when $N = 250, 300$. The percentage of nonfunctional nodes indicates how many nodes have depleted energy and are waiting for the MC. The response time to emergencies is measured from the time a node reports emergency until it is resolved by the MC. A shorter response time indicates that the MC can respond faster to emergencies.

In the static approach, the MC selects nodes with energy less than the normal recharge threshold, calculates the minimum traveling distance throughout these nodes and performs recharge one by one. Fig. 2.5(a) shows the percentage of nonfunctional nodes. We can see the number of nonfunctional nodes is much higher in the static optimization approach, e.g. when $N = 250$, there are around 15% nonfunctional nodes in our framework but nearly 30% in the static approach. This is because that some nodes in the pre-computed sequence may consume energy at faster rates, making the initial sequence computed in the static method no longer valid. Thus it cannot cover all the sensor nodes before energy depletion. Second, a node in emergency is not treated with priority in the static method. Thus a node in emergency may deplete its energy before the MC arrives, resulting in high percentage of nonfunctional nodes. The results in Fig. 2.5(a) clearly indicate that our real-time framework is more effective in recharging sensor nodes and resolving emergencies.

Fig. 2.5(b) shows the average response time to emergencies. We can see while our approach takes around 10 and 12.5 hours (for $N = 250$ and 350 respectively), the static approach takes drastically longer times (around 61 and 160 hours, almost one order of magnitude longer). This is because in [41] emergency and normal nodes are not differentiated. A pre-computed route containing both types of nodes would result in extremely long waiting times for emergency nodes. The approach degrades fast and becomes infeasible as the network size increases. In contrast, our approach prioritizes nodes in emergency; it resolves nonfunctional situations much faster than the static approach.

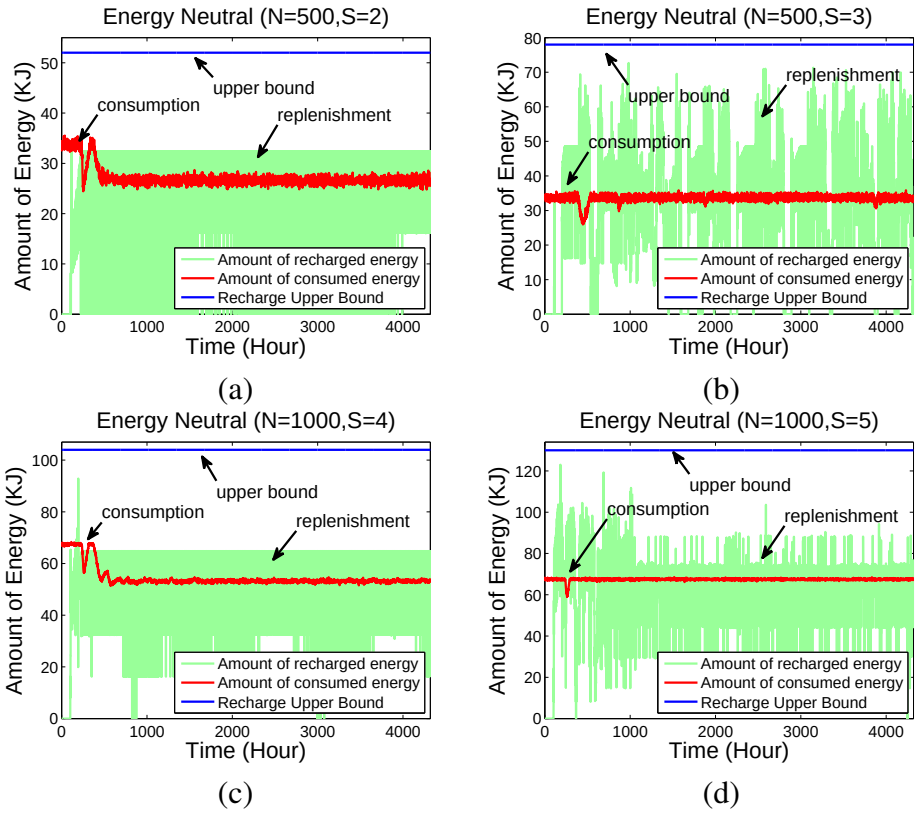


Figure 2.6: Evolution of energy consumption vs. energy replenishment in 6 months time. (a) $N = 500$, $S = 2$. (b) $N = 500$, $S = 3$. (c) $N = 1000$, $S = 4$. (d) $N = 1000$, $S = 5$.

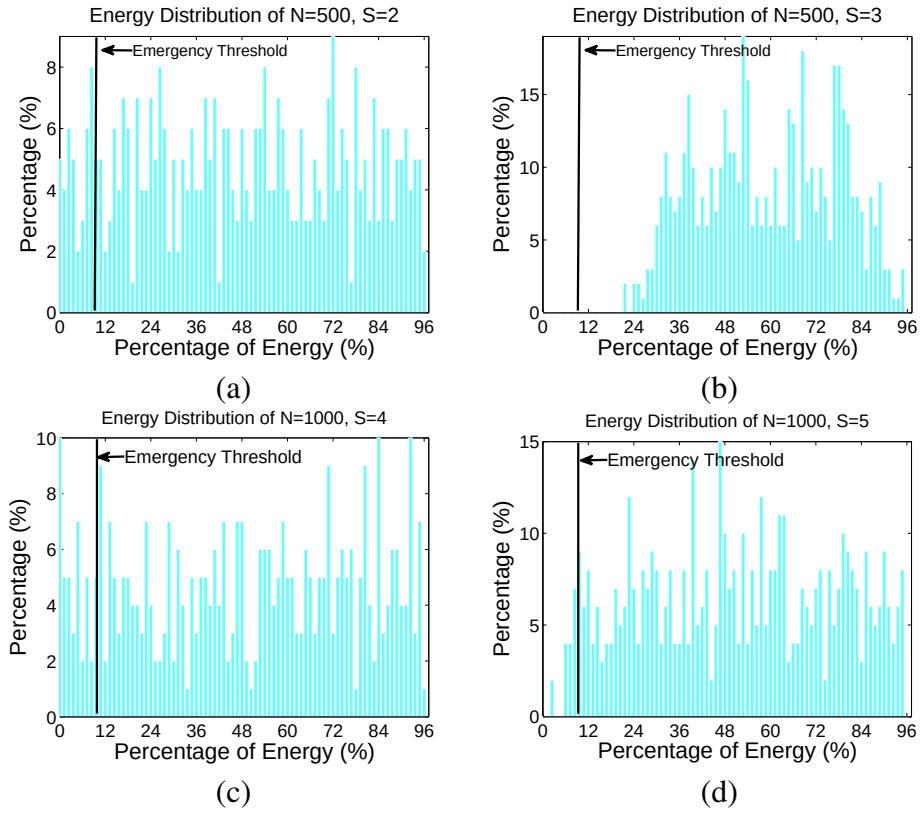


Figure 2.7: Energy distribution at equilibrium. (a)$N = 500$, $S = 2$. (b) $N = 500$, $S = 3$. (c) $N = 1000$, $S = 4$. (d) $N = 1000$, $S = 5$.

2.5.3 Network Performance

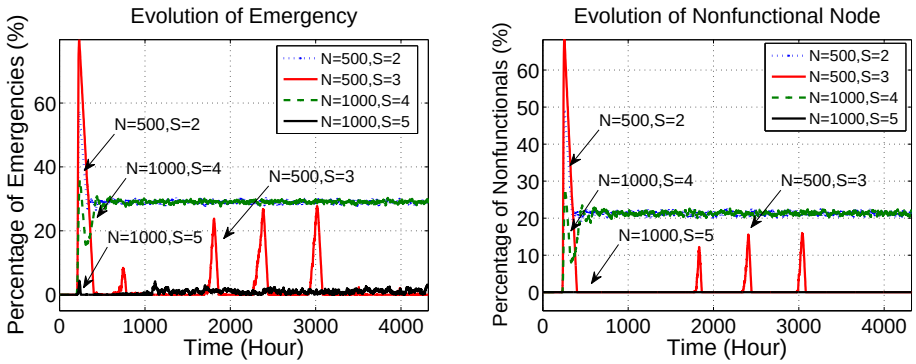
In this subsection, we evaluate the energy evolution of the network, the number of emergency and nonfunctional (i.e., energy depleted) nodes of the network, and the maintenance cost of the framework.

Energy Evolution

First, we show the energy evolution in the networks with 500 nodes and 1000 nodes served by different numbers of MCs, compared to the upper bounds of total recharge capabilities. In Fig. 2.4, the maximum recharging rate is achieved in the 17-th minute to 45-th minute duration. It replenishes 75% of the battery energy, equivalent to 433 J/min. The upper bound is calculated by assuming that all the MCs are performing recharge at maximum recharging rates all the time. In Fig. 2.6, the amount of energy consumed, replenished and recharge upper bound in every one-hour time slot is plotted as functions of the simulation time.

In Fig. 2.6(a) and (c), we can see that the consumed energy "steps down" to a lower level around 400 hours and then enters equilibrium. This is because a portion of sensor nodes deplete their energy and do not get recharged. In these two scenarios the energy neutral condition has been violated, simply because the number of MCs is not enough. Fig. 2.6(b) and (d) show the energy evolution when the number of MCs is increased by 1, both of which satisfy the energy neutral condition at the equilibrium and there is no such "step-down" effect in energy consumption. The gap between the recharge upper bound and energy replenished is due to that there are traveling time and idling time between two consecutive recharges in simulations. In addition, the MCs may perform normal recharge in which the recharging rates are much lower than the maximum recharging rates used for calculating the upper bound.

The energy distribution among nodes also carries valuable information about the health of the network. Higher average energy distribution is more robust to unexpected surges in energy consumption. Fig. 2.7 shows the energy distribution of $N = 500$, $S = 2, 3$ and $N = 800$, $S = 4, 5$. To see the benefits of more MCs, compare Fig. 2.7(a) to Fig. 2.7(b). The latter has energy distribution that concentrates around a higher average value. In Fig. 2.7(d) for a network size of 1000 sensors, the number of nodes with energy below the emergency threshold is



(a) (b)

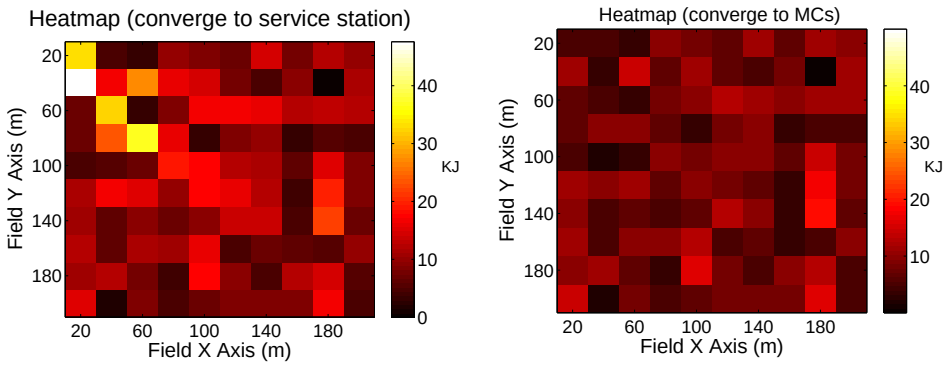
Figure 2.8: Number of emergent and nonfunctional nodes (a) number of emergent nodes (b) number of nonfunctional nodes.

significantly lower than that in Fig. 2.7(c).

Number of Emergencies

Fig. 2.8 compares the percentage of nodes in emergency and nonfunctional (i.e., energy at zero) status for networks of 500 and 1000 nodes with different numbers of MCs. First, we can see that there are surges in the numbers of emergency and nonfunctional nodes during the first 200 hours. This is due to the fact that the MCs only responds to requests when the node energy is below the normal recharge threshold. When such requests swarm into the job queues on the MCs at the beginning of 200 hours, we can see that the MCs' capacity has been temporarily exceeded. As the energy of sensors is restored, the numbers of emergency and nonfunctional nodes decrease sharply.

To illustrate the consequences of insufficient number of MCs, we vary the number of MCs S over a range including the minimum number needed for energy neutral. Fig. 2.8 (a) and (b) show the number of emergency and nonfunctional nodes over time. For cases $N = 500, S = 2$ and $N = 1000, S = 4$ when the number of MCs is insufficient for energy neutral, we can see that about 30% nodes are in constant emergency and 20% nodes are in nonfunctional status after the network achieves equilibrium. For $N = 500, S = 3$, there are occasional nonfunctional nodes but they are soon recharged by the MCs. For a majority of the time, the number of nonfunctional nodes stays at zero. For $N = 1000, S = 5$, the number



(a) (b)

Figure 2.9: Comparison of energy information convergence schemes (a) converge to service station (b) converge to MCs.

of nonfunctional nodes stays at zero at equilibrium with only a small number of emergencies.

2.5.4 Cost Evaluation

Comparison of Energy Information Collection Schemes

We compare energy consumption for different energy information collection schemes. Rather than collecting energy information at the MCs, another method is to route it through multi-hop transmission to the service station. The service station computes recharge schedules and disseminates decisions to MCs via long range radio. A challenge to this alternative scheme is that more energy is consumed on nodes near the service station. For demonstration purposes, we draw the heat map of energy consumed in a one-hour interval after the network enters equilibrium in Fig. 2.9. First, we observe that more energy is consumed if the information is routed back to the service station, i.e., 3-4 times of that to route it to MCs. Second, more energy is consumed on nodes near the service station, which is shown as bright spot in Fig. 2.9(a). Since a rechargeable battery has a limited number of recharging cycles, higher loads on these nodes result in more frequent recharge and faster battery expiration.

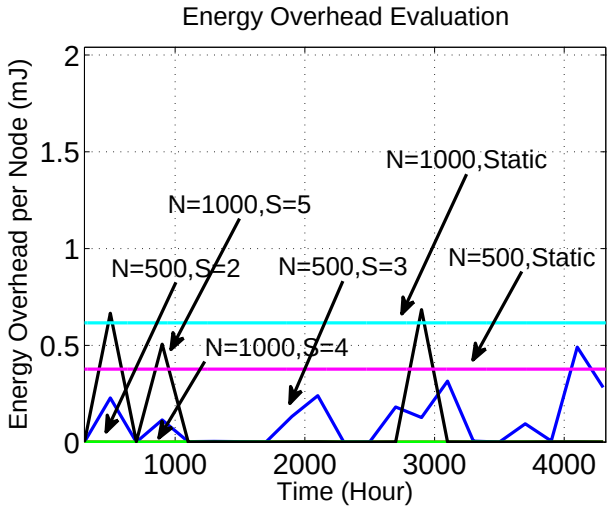


Figure 2.10: Average energy overhead for each sensor node per hour.

Evaluation of Protocol Overhead

We evaluate the energy overhead incurred during transmission of all types of messages sent by sensor nodes or MCs, and compare with that of the static scheme in [41], where energy information is routed to the service station every 6 hours. Fig. 2.10 shows the average energy overhead for each node per hour in a 6 month period for different scenarios. Due to head selection, our protocol has certain amount of overhead (around 6-8 mJ/h per node) at the beginning. We plot the energy overhead after the networks enter equilibrium.

First, we can see that the average energy overhead is from 0.1 to 1 mJ. Compared to the average energy consumption of 135 J/h (i.e., average energy consumption per slot 37.5 mJ times 3600s) due to sensing activities for each node, the overhead is negligible. Second, we can see that the average energy overhead is similar to that the static scheme. Intuitively, our protocol could incur more energy overhead because energy information is collected more frequently. However, the static scheme requires the energy information routed back to the service station. Thus unbalanced energy consumption on nodes near the service station is inevitable. The scalability of the scheme degrades when the size of the network increases. In contrast, delivering the energy information to multiple mobile MCs alleviates the unbalance in our protocol.

The energy overhead to gather emergency energy information is not significant

in our protocol. Both the emergency interests from MCs and emergency reports from nodes are sent directly to proxies (i.e., top-level heads) without propagating through the hierarchy, leading to less energy overhead. More importantly, when the number of MCs is sufficient, most of the time the network has very small fraction of nodes in emergency.

Normal energy information gathers causes more overhead. Upon receiving *energy interests*, the heads need to poll their descendants in a top-down manner which finally results in the broadcast of *energy interest* message in subareas at the bottom-level. Such broadcast, as well as the reply from each node, leads to the increase of the number of messages transmitted and the overhead. When the number of MCs is sufficient as $N = 500, S = 3$ and $N = 1000, S = 5$, more energy overhead is observed.

Evaluation of Load Balance and Mileage on MCs

We monitor the energy replenished by each MC and compare their workloads. The workload is measured by the amount of energy replenished to sensor nodes during the entire simulation period. Table 2.6 shows that the workloads are well balanced in all four scenarios due to the effective coordination in our framework. The MCs share the work evenly and no MC is overloaded. On the other hand, we use the mileages MCs travel to evaluate the cost (e.g., the energy consumed) for MCs to move around. Fig. 2.11 shows the accumulated mileages in 6 months. For both network sizes, the networks with fewer MCs (500 nodes and 2 MCs, 1000 nodes and 4 MCs) have lower mileage compared with the same network with more MCs (500 nodes and 3 MCs, 1000 nodes and 5 MCs), respectively. This is due to the presence of nonfunctional nodes. According to the calculation of weight for emergency selection (Eq. (2.20)), decision is made based on the residual lifetime of the nodes and the traveling time from the MCs to the nodes. For the networks with fewer MCs, there are always approximate 20% nonfunctional nodes after the networks enter equilibrium. The weights are dominated by the traveling time which is proportional to the distances from the MCs to these nodes. Thus the MCs always choose the nearest nodes for recharge. For the network with more MCs, however, the traveling time is not the dominating factor, thus the MCs may choose a farther node with shorter residual lifetime for recharge to avoid battery depletion. This

Table 2.6: Balance of load on MCs

MC	1	2	3	4	5
$N = 500$, $S = 2$	51%	49%	-	-	-
$N = 500$, $S = 3$	35%	34%	31%	-	-
$N = 1000$, $S = 4$	25%	25%	25%	25%	-
$N = 1000$, $S = 5$	22%	20%	20%	19%	19%

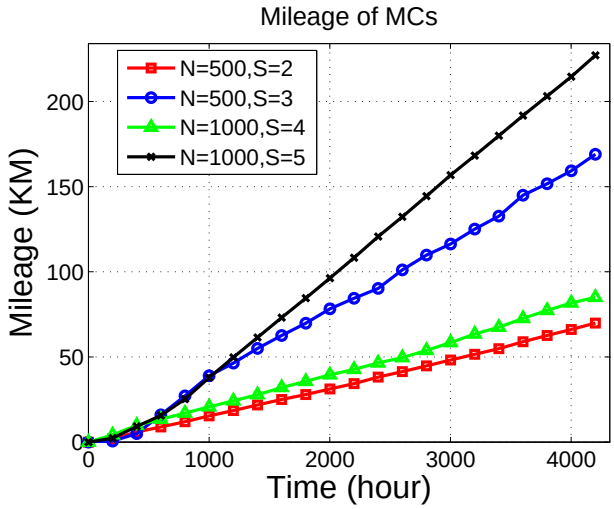


Figure 2.11: Mileages MCs have traveled in 6 months.

causes the increase of MC mileage.

2.6 Conclusion

In this chapter, we study how to coordinate multiple MCs to recharge sensors and propose a distributed real-time communication protocol for gathering energy information. The protocols can adapt to unpredictable network conditions and satisfy the needs for both normal and emergency recharging. Then we discussed several recharge scheduling algorithms for different scenarios. In case of emergency recharge, an MC needs to resolve multiple emergencies at different locations. The problem is formulated into the classic Orienteering Problem that aims to maximize the total amount of recharged energy in a given time period. Based on the fact that recharging time is much larger than traveling time, the problem can be simplified into a Knapsack problem solved by dynamic programming with high accuracy. In

the meanwhile, a weighted sum algorithm that considers sensor's lifetime and MC's traveling time is proposed for normal recharge operation. Performance evaluations have demonstrated the proposed algorithms can maintain perpetual operations of the network effectively.

Chapter 3

Mobile Data Gathering and Recharge Scheduling with Mobile Charger's Movement Costs and Capacity Constraints

3.1 Motivations

In the previous chapter, we have proposed a novel framework that powers sensor nodes via wireless energy. We can see that as long as the number of MCs are enough and the MCs are scheduled appropriately, we can extend the lifetime if WRSN to infinitely long time for *perpetual operations*. However, two important questions are still left unanswered. First, how can we determine the number of MCs for a network plan ? Second, what if the MC's own energy consumption and battery capacity are considered ? Are there any impacts on recharge scheduling ?

To answer the first question, we need to analyze the relations between energy consumption and replenishment. This is because, for WRSNs, energy replenishment cannot be considered separately from energy consumption patterns, which rely on how data is gathered in the network. Previous works in [42, 71] simply utilize a static data sink to gather packets over multi-hops. It is subject to the infamous energy hole problem [14] where nodes near the base station consume energy and deplete batteries much faster, causing service interruptions. A single vehicle that

gathers data and charges nodes simultaneously[41] can mitigate the problem. However, it causes high data collection latency due to the non-negligible battery recharge time. A battery requires nontrivial recharge time (e.g., 30 to 90 min) whereas gathering data takes only a few minutes (e.g., 1.6 min for transmitting 3 MBytes at 250 kbps). Thus the waiting time for completing recharge increases dramatically when more nodes need recharge. The gathered data would inevitably experience long latency and may be of little value when delivered to the base station for some time-sensitive applications.

Second, ignoring MC's own energy consumption during movement and recharge may lead to serious problems in reality. First, it may cause impractical schedules where the MCs deplete their energy, become stranded and unable to return to the base station. The network would eventually use up energy and stop operation completely. Second, they tend to overestimate the MC's recharge capability and nodes' lifetimes since the MCs have limited battery capacity. They have to spend time returning to the base station for battery replacement and cannot keep recharging nodes continuously. Third, they may result in inefficient recharge scheduling and node selection. They may choose nodes faraway simply because they have lower energy levels, and subsequently MCs travel back-and-forth over long distances, wasting significant amounts of energy.

To answer the above two problems, we propose a comprehensive framework that solves both data collection and recharge scheduling problems. To eliminate the entanglement between recharging and latency, we employ a separate, dedicated *data gathering vehicle*. Thus the data latency only depends on the mobility pattern (e.g., dispatching frequency, number of stops, speed) of this vehicle. This avoids long latency caused by slow recharging processes [41]. To prevent stopping at every node thus prolonging the tour length and latency, we let nodes form clusters and forward data to cluster heads. Thus only stops at these cluster heads are needed. A series of interesting questions arise in this new scheme. First, what should be the appropriate cluster size such that all nodes are covered while there are not too many clusters causing long latency? Second, what is the minimum number of MCs to cover all the nodes given a bounded cluster size? To answer these questions, we establish a mathematical model for the energy neutral condition to characterize the tradeoff between data collection latency and the number of MCs, both related to the cluster size. A small cluster size leads to more stops, thus higher latency. In the

extreme case of single-hop clusters, the vehicle has to traverse through every other node to obtain all the data. A large cluster size reduces latency, but incurs more relaying traffic and more energy consumption. Our model successfully quantifies such trade-offs.

Next, we consider MC's limited battery capacity and their moving energy consumptions in recharge scheduling. We maximize *recharge profit* (i.e., the recharged energy less the traveling cost), while meeting nodes' battery deadlines and MC's capacity constraints. These constraints bring us new challenges. On one hand, recharging nearby nodes reduces an MC's moving cost. On the other hand, faraway nodes, not just nearby ones, need recharge once in a while. We have to balance between the need to recharge the whole network and the desire to minimize the traveling cost. In particular, we need to answer the following questions: How to schedule MCs so they will not waste energy traveling back and forth over long distances? Which nodes an MC should select to ensure it has enough energy to return, and in what orders so as to meet nodes' battery deadlines? We formulate the recharge scheduling problem into an optimization of *Profitable Traveling Salesmen Problem with Capacity and Battery Deadline Constraints*, which was studied before but has only computationally intensive solutions.

We propose two efficient algorithms. The first is a simple *Greedy Algorithm* that maximizes an MC's profit at each step. However, it may lead to long traveling distances. We further propose a three-step *Adaptive Algorithm*. After collecting recharge requests, it partitions the network into several regions using the K-means algorithm [66]. Each MC is assigned a region and its movements are confined within the region, so long-distance travels are avoided. Then each MC works independently to construct *Capacitated Minimum Spanning Trees* in its designated region where edges in the tree have the minimum traveling cost. This ensures that the MC's capacity is not exceeded so it can return to its starting position. Finally, the algorithm performs route improvements to meet nodes' battery deadlines. It categorizes nodes according to their lifetimes. An initial route containing nodes that do not need prioritized recharge is first constructed using Traveling Salesmen Problem algorithms. Then it inserts nodes that need prioritized recharge into the route while ensuring each insertion retains time feasibility of the whole recharge sequence.

The rest of the paper is organized as follows. Section 3.2 outlines the network

model and assumptions. Section 3.3 describes the main design of low latency mobile data collection. A mathematical model with a set of theoretical results are derived in Section 3.4. Section 3.5 formalizes the recharge optimization problem and proposes two algorithms. Finally, Section 3.6 provides the evaluation results and Section 3.7 concludes this chapter.

3.2 Network Model and Assumptions

Fig. 3.1 gives a pictorial illustration of the network. Sensory data is generated at *normal nodes* and aggregated at *anchor points* (i.e., cluster heads) in a multi-hop fashion. A *data gathering vehicle* traverses the sensing field periodically and stops at *anchor points* to collect data. It uploads the collected data to the *base station* at the end of each data collection tour. The base station also provides basic maintenance of the network by offering battery replacement. It can be commanded by network administrators remotely to perform computations such as network partitioning in the Adaptive Algorithm proposed later.

Meanwhile, a fleet of *Mobile Chargers* query the network for energy information using the mechanism introduced in [45]. The MCs send those queries periodically, make recharge decisions (i.e., which nodes to recharge, in which order) and recharge nodes accordingly. Once an MC fulfills all requests, it sends out a query to see whether there is new energy request. Both types of vehicles return to the base station and have their own batteries replaced when their energy is low.

We assume a number of N_s sensor nodes are uniformly randomly scattered in a square sensing field with side length L. Node density of the network is $\rho = \frac{N_s}{L^2}$. In this paper, we focus on event-driven sensing applications and assume events occur at every location with equal probability, spatially and temporally independent of each other. Thus, the data generation process can be modeled as a Poisson process with average rate λ[74]. All sensors transmit at the same power level with fixed transmission range d_r. The energy consumed for transmitting/receiving a packet of length l, denoted by e_t, e_r, is modeled as in [70], i.e., $e_t = (e_1 d_r^{\alpha} + e_0)l$, where e_1 is the loss coefficient per bit, α is the path loss exponent (usually from 2 to 4) and e_0 is energy consumed on sensing, coding, modulations. In this paper, we use $e_0 = 50 \times 10^{-8}$ J/bit, $e_1 = 10 \times 10^{-8}$ J/bit, $\alpha = 4$.

The network is split into a number c clusters. A cluster is formed in a way such

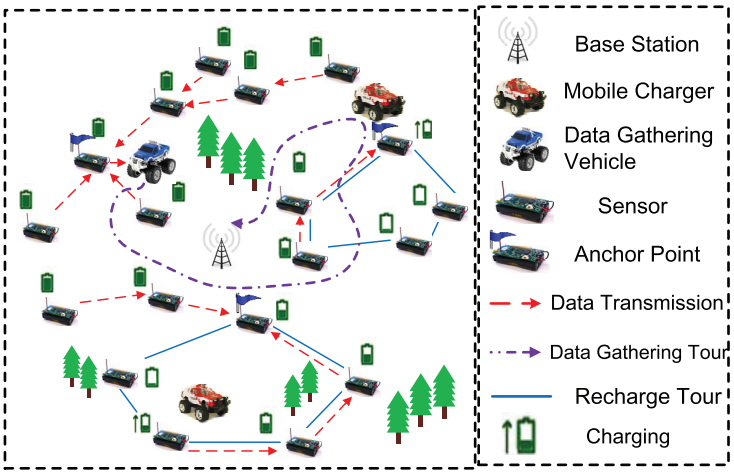


Figure 3.1: Illustration of the network architecture and components.

that the maximum hop count from a node to the *anchor point* (cluster head) is k. When a node falls within k-hops of multiple anchor points, it will join the cluster with the least number of hops. A *data gathering vehicle* starts from the base station every T_c time period, stops at anchor point location i for time t_i to gather all sensed data and returns to the base station after all anchor points have been visited. The dispatch interval T_c is greater than the duration of the data gathering tour. The data gathering vehicle visits anchor point locations directly to minimize transmission energy consumption on these nodes. The transmission bit rate is B.

There are also m *mobile chargers* working together to replenish sensor batteries. A number of nodes are selected for an MC to form its recharge set. If a node cannot survive the time needed to recharge all the other nodes in the set, it needs *prioritized recharge* (i.e., it should be charged earlier in the recharge sequence). MCs can bring sensor batteries from zero to full capacity C_s in T_r time which is governed by battery characteristics (e.g., for a Panasonic Ni-MH AAA battery [46] of battery capacity C_s = 780 mAh and T_r = 78 min.). All the vehicles are equipped with high-capacity batteries of C_h capacity and consume at e_c J/m while moving at speed v m/s. In addition, we have made the following assumptions: 1) we assume the energy consumption during transmission and reception of a packet is equivalent ($e_r \approx e_t$); 2) the MCs have positioning systems and know their locations; 3) the locations of all the sensor nodes are known to the MCs (e.g., through a one-time effort during network initialization). Finally, important notations used in this paper are summarized in Table 3.1.

3.3 Low Latency Mobile Data Collection in WRSNs

We now study how to minimize data collection latency given k-hop clusters. To minimize delay, it is desirable to have the data gathering vehicle stop at fewer anchor points. To ensure all data can be collected, the k-hop coverage areas of these anchor points should collectively cover the entire network. The delay mainly depends on three variables: sum of stopping time at anchor points, traveling time through all anchor points and data uploading time to the base station.

The stopping time at each anchor point depends on the amount of data generated during two consecutive visits of the data gathering vehicle. The traveling time depends on the number of anchor points and vehicle's speed. Hence, let us first

Table 3.1: List of Notations

Notation	Definition
N_s	Number of sensor nodes
L	Side length of squared sensing field
c	Number of clusters in the network
k	Cluster size in terms of communication hop count
m	Number of MCs
d_r	Transmission range of sensor nodes
e_t, e_r	Energy consumptions for transmitting and receiving a packet
e_c	Energy consumption of MC while moving
λ	Average packet rate of Poisson distributed traffic
T_c	Data collection period
B	Data uploading bit rate
C_s	Battery capacity of sensor nodes
C_h	Battery capacity of MCs
T_r	Recharge time of sensor's battery
v	Moving speed of vehicles

determine the number of anchor points that can cover the entire sensing field in k hops. As studied in [14], a k-hop cluster can be closely approximated by a circle with radius $r = kd_r$ with k coronas as shown in Fig. 3.2. Then, finding the minimum number of anchor points is equivalent to finding a complete coverage of the sensing field with minimum number of circles of radius r. The problem is closely related to the circle covering problem studied by Kershner [73], which gives the minimum number of circles needed to cover a rectangular region in the following lemma.

Lemma 1: The number of circles c to cover a sensing field with area L^2(L is the side length of the field) has the lower bound of ([73])

$$c > \frac{2\pi\sqrt{3}(L^2 - 2\pi r^2)}{9\pi r^2} \tag{3.1}$$

Although the exact placement pattern to achieve this lower bound was not given in [73], it has been proved in [20] that the maximum coverage is achieved when we tessellate the sensing area with equilateral triangles of side length $\sqrt{3}kd_r$ and place the centers of circles at the vertices of triangles. However, how to place these clusters in a square sensing field considering the effects of boundaries was

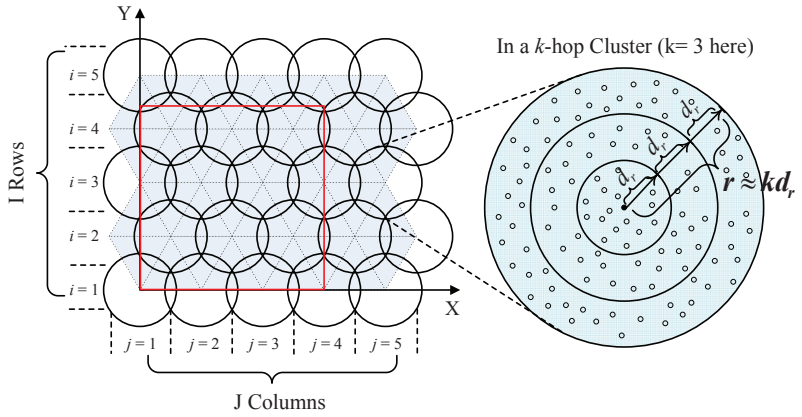


Figure 3.2: Example of equilateral triangular tessellation of clusters covering a sensing field with hop count $k = 3$.

not discussed in [20] so we introduce a placement pattern first. For a square sensing field with the origin (0,0) at the left bottom, we place I circles parallel to the x-axis and J circles parallel to the y-axis, then the cartesian coordinates of centers of circles at the i-th row, j-th column are

$$[X_{ij}, Y_{ij}] = \begin{cases} [\sqrt{3}(j-1)r, \frac{3}{2}(i-1)r] \\ i = \{2u+1; \forall u \in \mathbb{Z}\} \\ [\frac{\sqrt{3}}{2}r + \sqrt{3}(j-1)r, \frac{3}{2}(i-1)r] \\ i = \{2u; \forall u \in \mathbb{Z}\} \end{cases} \tag{3.2}$$

After the deployment pattern has been determined, the number of circles I to cover each row can be calculated as

$$I = \begin{cases} \lfloor \frac{L}{\frac{3}{2}r} \rfloor + 1, \frac{L}{\frac{3}{2}r} - \lfloor \frac{L}{\frac{3}{2}r} \rfloor \leq \frac{1}{2} \\ \lfloor \frac{L}{\frac{3}{2}r} \rfloor + 2, \text{otherwise} \end{cases} \tag{3.3}$$

The number of circles J to cover each column with an odd index $i = \{2u+1; \forall u \in \mathbb{Z}\}$ is

$$J = \begin{cases} \lfloor \frac{L}{\sqrt{3}r} \rfloor + 1, \frac{L}{\sqrt{3}r} - \lfloor \frac{L}{\sqrt{3}r} \rfloor \leq \frac{1}{2} \\ \lfloor \frac{L}{\sqrt{3}r} \rfloor + 2, \text{otherwise} \end{cases} \tag{3.4}$$

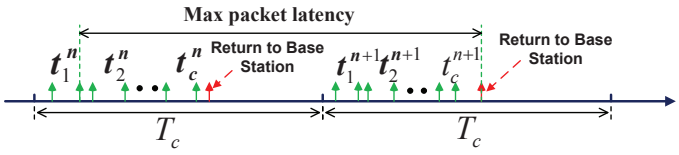


Figure 3.3: A timing diagram of two consecutive mobile data gathering tours.

The number of circles J to cover each column with an even index $i = \{2u; \forall u \in \mathbb{Z}\}$ is

$$J = \begin{cases} \lfloor \frac{L}{\sqrt{3}r} \rfloor, \frac{L}{\sqrt{3}r} - \lfloor \frac{L}{\sqrt{3}r} \rfloor = 0 \\ \lfloor \frac{L}{\sqrt{3}r} \rfloor + 1, \text{otherwise} \end{cases} \tag{3.5}$$

Fig. 3.2 shows an example of equilateral triangular tessellation of 14 clusters covering a square sensing field with $k = 3, L = 3\sqrt{3}r$. Compared to the lower bound of $c > 7.97$ obtained from Eq. (3.1), an additional 6 clusters are needed to cover the boundaries of the field. Given a field length L, the number of clusters c (number of anchor points), coverage of the entire sensing field can be obtained from Eqs. (3.3), (3.4) and (3.5). Then we derive an upper bound of mobile data gathering latency in the following lemma.

Lemma 2: The mobile data gathering latency is bounded by

$$T_d \leq T_c + (c-1)T_s + (\sqrt{2(c-3)}L + 4L)/v \tag{3.6}$$

where T_d is the data latency, $T_s = F_\lambda^{-1}(\epsilon)\rho r^2 \pi l T_c / B$, $F_\lambda^{-1}(x)$ is the inverse CDF of Poisson distribution with average rate λ, ϵ is a value close to 1 but not equal to 1 (e.g. $\epsilon = 0.99$), v is the vehicle's speed.

Proof. Fig. 3.3 shows a timing diagram of mobile data gathering. t_i^n is the stopping time at the i-th anchor point during the n-th round of data gathering. We observe that the maximum latency occurs when a packet arrives at the first anchor point in the visiting sequence after the data gathering vehicle has left. Then the packet has to be buffered and wait for another collection period after time T_c, plus sum of stopping time at subsequent anchor points, traveling time to the base station through the rest of anchor points. The maximum stopping time T_s at an anchor point occurs when each node generates at maximum data rate $F_\lambda^{-1}(\epsilon)$. Note that ϵ is a value

very close to 1 but not equal to 1 (e.g., $\epsilon = 0.99$) because $F_\lambda^{-1}(1) \to \infty$. For each cluster with a number of $\rho r^2\pi$ sensors, $T_s = F_\lambda^{-1}(\epsilon)\rho r^2\pi/B$. Therefore, the sum of stopping time at subsequent anchor points is bounded by $(c-1)T_s$.

To traverse $(c-1)$ nodes, a deterministic upper bound on the shortest tour length was given in [68]. That is, for n points in a rectangle with size $a \times b$, the shortest tour length $s < \sqrt{2(n-2)ab} + 2(a+b)$. Here, $a = b = L$, $n = c-1$, so the upper bound of traveling time is $(\sqrt{2(c-3)}L + 4L)/v$. By summing by this result with maximum stopping time at subsequent anchor points $(c-1)T_s$ and T_c, we have derived an upper bound of mobile data gathering latency. □

From *Lemma 2*, we can compare the data gathering latency with the combined approach in [41] numerically. For MCs of battery capacity 12Ah of 5V ($C_h = 216KJ$), a recharge tour would take around $\frac{C_h T_r}{C_s} = 32$ hours to finish. This amounts to at least 32 hours waiting time for the data to be delivered to the base station till the vehicle returns to the base station for battery replacement. For our approach, we set $N_s = 500$, $T_c = 60$ mins, $r = 45$ m, $c = 14$, $L = 160$m, $B = 250$ Kbps, $l = 10$ bytes, $\lambda = 3$ and after plug into Eq. (3.6), we have $T_d \leq 1.65$ hours which is significantly less than the combined approach about an order of magnitude. For further improvement of latency, we can dispatch the data gathering vehicle more frequently by using a small T_c. We will use different T_c to see their average latencies and corresponding upper bounds in the simulations.

3.4 Number of Mobile Chargers for k-hop WRSN

Having discussed k-hop cluster formation and data latency in our framework, we now analyze the minimum number of MCs needed to fulfill all energy requests given the number of clusters c obtained from Eqs. (3.3), (3.4), (3.5).

3.4.1 Number of MCs

For the perpetual operation of the network, the energy neutral condition must hold in a long time period ,

$$E(T) \leq R(T) + E_0 \tag{3.7}$$

in which T is a large time, $E(T)$ is the total energy consumption of the network up to T, $R(T)$ is the total energy replenished into the network by the MCs up to T and E_0 is the initial energy of all the sensor nodes. The energy neutral condition states that the energy consumption of all the sensor nodes must be less than or equal to the total energy available in long term. Otherwise, sensor nodes would eventually deplete energy. Note that for the network to function, it is not necessary for the condition to hold at every single moment. In practice, a small fraction of the network may consume more energy in a short time window due to external activities, leading to temporary unbalance between energy consumption and replenishment. As long as there are enough MCs, these nodes will be recharged, and such unbalance is transient, not permanent.

Our objective is to obtain the minimum number of MCs m needed for Eq. (3.7) to hold. First, we estimate $R(T)$ which is the amount of energy that can be replenished into the network. The maximum recharge capacity of an MC is achieved when it recharges sensor nodes continuously without any idling time. The longest recharging time for a sensor occurs when a node's energy is brought from zero energy to full capacity which takes T_r time plus the longest moving time between two consecutive sensors in the recharge sequence (moving on the diagonal of the square sensing field). Therefore, in the worst scenario, it takes $\sqrt{2}L/v + T_r$ time to recharge each sensor. Then we can estimate the energy replenished into the network in T time by m MCs,

$$R(T) = \frac{mC_bT}{\sqrt{2}L/v + T_r}. \tag{3.8}$$

Next, we need to derive $E(T)$ on the left hand side of Eq. (3.7) which is a random variable. Given the structure of the cluster of radius $r = kd_r$, each corona carries traffic loads from all outer coronas. The number of nodes in the i-th corona, is $N_i = (2i-1)d_r^2\pi\rho$ for $0 < i \leq k$. Since the outmost k-th corona only needs to send out its own data and data is generated independently, the mean of energy consumption at the k-th corona μ_k in time period T is,

$$\mu_k = N_i\lambda T e_t = (2k-1)d_r^2\pi\rho\lambda T e_t \tag{3.9}$$

For the i-th corona ($0 < i < k$), it carries all the traffic from the outer coronas so

the mean energy consumption is,

$$\begin{aligned}\mu_i &= N_i\lambda T e_t + \sum_{j=i+1}^{k} N_j\lambda T(e_t+e_r) \\ &= d_r^2\pi\rho\lambda T\big((k^2-i^2)(e_t+e_r)+(2i-1)e_t\big)\end{aligned} \tag{3.10}$$

Then we can compute the mean of network energy consumptions $\overline{E(T)}$,

$$\begin{aligned}\overline{E(T)} &= (\sum_{i=1}^{k-1}\big((k^2-i^2)(e_t+e_r)+(2i-1)e_t\big) \\ &+ (2k-1)e_t + k^2(e_t+e_r))d_r^2\pi\rho\lambda Tc \\ &= \left((\frac{2}{3}k^3-\frac{1}{2}k^2-\frac{1}{6}k)(e_t+e_r)+k^2e_t\right)d_r^2\pi\rho\lambda Tc\end{aligned} \tag{3.11}$$

Based on the energy neutral condition, by combining $R(T)$ in Eq. (3.8) and $\overline{E(T)}$ in Eq. (3.11), we have the following lemma. *Lemma 3:* The probability for the energy neutral condition to hold is

$$P_{op} = \Phi\left(\frac{R(T)+E_0-\overline{E(T)}}{\sqrt{\overline{E(T)}}}\right) \tag{3.12}$$

where $R(T)$ and $\overline{E(T)}$ are obtained in Eq. (3.8) and Eq. (3.11), respectively. $\Phi(\cdot)$ denotes the Cumulative Distribution Function of the Normal distribution.

Proof. Energy consumption of a cluster can be described by the sum of independent Poisson variables over T. When T is observed over a long time period, we can use the *Central Limit Theorem* to approximate Poisson distribution by a Normal distribution $\mathcal{N}(\overline{E(T)},\overline{E(T)})$(the mean and variance of a Poisson distribution is the same) [72]. □

From Lemma 3, we immediately get the following Proposition.

Proposition 1: The minimum number of MCs required to achieve perpetual

operation is

$$m = \left\lceil \frac{(\Phi^{-1}(\epsilon)\sqrt{\overline{E(T)}} + \overline{E(T)} - E_0)(\sqrt{2}L/v + T_r)}{C_s T} \right\rceil \tag{3.13}$$

where $\Phi^{-1}(\epsilon)$ is the inverse cumulative distribution function of Normal distribution, ϵ is a value very close to 1 but not equal to 1.

Proof. Since $\Phi^{-1}(1) \to \infty$, we consider the network achieves perpetual operation with very high probability approaches 1 but not equal to 1, e.g. $\epsilon = 0.99, \Phi^{-1}(0.99) \approx 2.33$. From Eq. (3.12), we have

$$\frac{\frac{mC_bT}{\sqrt{2}L/v+T_r} + E_0 - \overline{E(T)}}{\sqrt{\overline{E(T)}}} \geq 2.33,$$

after some manipulations we can obtain the minimum number of MCs m needed to satisfy the energy neutral condition. □

Based on the results from *Proposition 1* and *Lemma 2*, we demonstrate the trade-off between number of MCs and data latency. For $L = 400$ m, we change the number of cluster hop count k and plot the corresponding number of MCs needed as well as upper bound of data latency in Fig. 3.4. We can see a trade-off point around $k = 3$. It means when $k = 3$, we can minimize the number of MCs without sacrificing too much from the data collection latency.

3.4.2 Estimate Node Lifetime

To devise effective recharging schedules, we need to know how long a sensor node can survive after it has requested for recharge. Such information is vital in making recharge decisions in the next section. Since a node's energy consumption rate is a random variable and depends on traffic patterns, it is important for each node to know its traffic burden which is determined by the number of hops from base station. This information can be obtained by message propagation from the base station in various routing protocols and adjusted accordingly during operation.

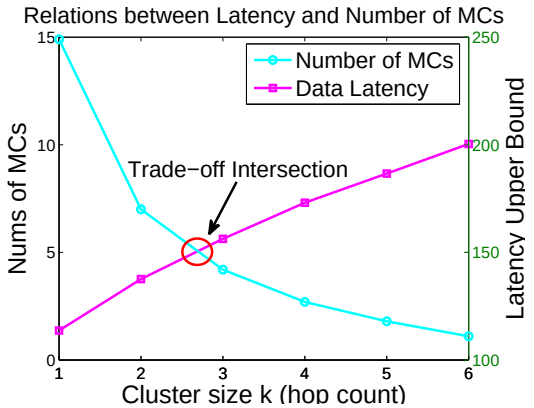


Figure 3.4: Trade-off between number of MCs and data collection latency.

From Eqs. (3.9) and (3.10), we know the average traffic rate of a node in the j-th corona ($1 \leq j \leq k$) is, $\lambda_j = \lambda(1 + (k^2 - j^2)/(2j - 1))$. Given residual energy E_r, the maximum number of packets the node can transmit is $n = \lfloor \frac{E_r}{(e_t + e_r)} \rfloor$.

Lemma 4: Given a recharge sequence of N nodes in which a node at the j-th corona waiting to be recharged, it will survive time t with probability (lifetime $L_j > t$),

$$P(L_j > t) = 1 - \frac{\gamma(N, \lambda_j t)}{\Gamma(N)}, \tag{3.14}$$

where $\gamma(\cdot, \cdot)$ and $\Gamma(\cdot)$ are the lower incomplete gamma function and complete gamma function[72], respectively.

Proof. Since sensor nodes are randomly deployed in the field, and the data generation process is independent of each other, the summation of packet interarrival times until the sensor node can no longer transmit packets is the lifetime of the sensor node. Because data generation is Poisson distributed with rate λ_j, the interarrival time of packets is exponentially distributed. It is known that the sum of independently identically distributed exponential variables results a *Gamma distribution* with probability density function

$$f_{L_j}(x) = \lambda_j e^{-\lambda_j x} \frac{(\lambda_j x)^{N-1}}{(N-1)!}, x \geq 0 \tag{3.15}$$

and the Cumulative Distribution Function of Gamma distribution is

$$P(x < t) = \int_0^t \lambda_j e^{-\lambda_j x} \frac{(\lambda_j x)^{N-1}}{(N-1)!} dx = \frac{\gamma(N, \lambda_j t)}{\Gamma(N)} \tag{3.16}$$

□

Proposition 2: For the recharge sequence of N nodes, if a node at the j-th corona has probability $\frac{\gamma(N,\lambda_j T_l)}{\Gamma(N)} \approx 0$, $T_l = (N-1)(T_r + \sqrt{2}L/v)$, no matter where the node is placed in the recharge sequence, it will not deplete battery energy before its recharging starts.

Proof. The worst case occurs when the node is placed at the end of the recharge sequence. The longest waiting time to get recharged is $T_l = NT_r + (N-1)\sqrt{2}L/v$ since there are $N-1$ nodes ahead with $\sqrt{2}L/v$ maximum traveling time between two sensor nodes and $\sqrt{2}L$ is the diagonal of the square field. Once $\frac{\gamma(N,\lambda_j T_l)}{\Gamma(N)} \approx 0$, $P(L_j > T_l)$ approaches probability 1 so it is guaranteed to recharge the node before it depletes battery energy. □

Based on *Proposition 2*, given a recharge sequence, we can calculate the possibility that a node can survive the entire recharging process. This lays the theoretical foundations to solve the recharge scheduling problem in the next section.

3.4.3 Adaptive Recharge Threshold

We observe that the difference of energy consumptions between nodes at different locations is mainly caused by data communications. Although the hop count for clusters k should not be too large to avoid the energy hole problem on anchor points, it is inevitable to have higher data traffic in the inner coronas. If all the nodes follows a universally same recharge threshold, it may result some nodes closed to the anchor point nodes to deplete energy very soon and lead to unfair service for nodes with higher consumption rates. To this end, the recharge thresholds should be made adaptive and proportional to energy consumption rates at different coronas. In other words, nodes closer to the anchor points should request recharge more frequently than others.

Let $\tau_i (0 < \tau_j < 1)$ denote the recharge thresholds for nodes at the j-th corona. We make the ratio between the recharge thresholds of corona i and j equal to that

between their energy consumptions due to data transmission. Assume we have set the recharge threshold of the first corona to be τ_1. The thresholds for other coronas are,

$$\tau_i = \frac{(k^2 - i^2)(e_t + e_r) + e_t(2i-1)}{(k^2-1)(e_t+e_r)+e_t}\tau_1 \approx \frac{2k^2 - (i-1)^2 - i^2}{2k^2-1}, \tag{3.17}$$

where $0 < i < k$. The approximation is taken under the assumption that $e_t \approx e_r$. To illustrate Eq. (3.17), e.g. $k = 5$, after τ_1 is set, we obtain $\tau_2 = \frac{45}{49}\tau_1$, $\tau_3 = \frac{37}{49}\tau_1$, $\tau_4 = \frac{25}{49}\tau_1$ and $\tau_5 = \frac{9}{49}\tau_1$.

3.5 Capacitated Recharge Problem with Battery Deadlines

During operation, the MCs query sensors for recharge and they usually engage in multiple recharge tasks at different locations. In this section, we study a Capacitated Multi-Vehicle Recharge Problem with Battery Deadlines (CaMP-BaD) and consider practical constraints from real sensing applications. The first challenge is the constant changes (i.e., decrease) of charging vehicles' energy due to moving and recharging sensor nodes. The recharge route should be planned carefully to reflect MC's current energy status and traveling costs to nodes' locations. The second challenge is the nonuniform energy consumption due to data transmissions. Some nodes consume energy at higher rates and should be taken care of more frequently than others to maintain the functionality of the network. The recharge routes should reflect all aforementioned concerns. The difficulty of the problem lies in achieving conflicting goals - the need to keep the whole network running pushes the charging vehicles to recharge as many sensor nodes as possible while the desire to reduce cost means that charging vehicles should minimize traveling distances to save energy cost. Therefore, an ideal solution should achieve a good balance between the two without sacrificing either.

Next we show the recharge problem can be formulated into a *Profitable Traveling Salesmen Problem with Capacity and Battery Deadline constraints* (PTSP). In the Profitable Traveling Salesmen Problem [51], a reward is collected by visiting a city while the objective is to maximize the profit which is defined as the reward

minus cost. In our problem, the reward represents the amount of energy that can be replenished into a sensor node and the cost measures the energy cost in traveling to that node's location.

To tackle the problem, we first present a straightforward *Greedy Algorithm* (GA). After realizing that the greedy algorithm might incur extra movements of charging vehicles, we further propose a three-step *Adaptive Algorithm* (AA) through 1) adaptive network partition using K-means algorithm, 2) Capacitated Minimum Spanning Tree (CMST) formation and 3) route improvements using node insertions. By partitioning the network, the MCs are confined in their own regions so traveling back and forth through the entire field is avoided. Then we form CMST for each MC. The trees indicate which subset of sensor nodes the MC should select to minimize traveling cost and ensure the total weight of the tree is within the MC's recharge capacity. After that, we perform route improvements on nodes in CMST to capture sensor nodes' dynamic battery deadlines. Finally, we analyze the complexity of the proposed algorithms.

3.5.1 Problem Formulation

The recharge optimization problem can be defined as follows. Given a set of MCs $\mathcal{S} = \{1, 2, \ldots, m\}$ and a set of recharge node list $\mathcal{N} = \{1, 2, \ldots, n\}$, we formulate the CaMP-BaD problem into a PTSP problem. Consider a graph $G = (V, E)$, where V_i ($i \in \mathcal{N}$) is the location of sensor node i to be visited, and E is the set of edges. We add a vertex V_0^a as the starting position of MC a. Each edge E_{ij} is associated with a traveling energy cost c_{ij}, which is proportional to the distance between nodes i and j, c_{0i}^a represents the cost from initial position V_0^a of MC a to node i. An MC a has recharge capacity C_a ($\leq C_h$) that determines the maximum number of nodes it can recharge before it goes back to the base station for its own battery replacement. Different MCs might have different recharge capacities during the run. Each sensor node i has lifetime L_i and demand (reward) for energy recharge r_i (demand equals the total battery capacity of a sensor node minus its residual energy). A_i specifies the arrival time for an MC at sensor node i.

We introduce two decision variables x_{ij}^a for edge E_{ij} and y_{ia} for vertex V_i. The decision variable x_{ij}^a is 1 if an edge is visited by vehicle a, otherwise it is 0. The decision variable y_{ia} is 1 if and only if node i is served by vehicle a, otherwise it is

0. u_i is the position of vertex i in the path. Our objective is to maximize the total amount of energy recharged minus total traveling energy cost of the MCs while ensuring their recharge capacities are not exceeded and no sensor node depletes battery energy.

$$\mathbf{P1}: \quad \max \left\{ \sum_{a=1}^{m} \sum_{i=1}^{n} r_i y_{ia} - \sum_{a=1}^{m} \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}^{a} - \sum_{a=1}^{m} \sum_{i=1}^{n} c_{0i}^{a} x_{0i}^{a} \right\} \tag{3.18}$$

Subject to

$$\sum_{j=1}^{n} x_{0j}^{a} = 1; a \in \mathcal{S} \tag{3.19}$$

$$\sum_{i=1}^{n} x_{ik} = \sum_{j=1}^{n} x_{kj} = 1; k \in \mathcal{N} \tag{3.20}$$

$$\sum_{i=1}^{n} r_i y_{ia} + \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}^{a} + \sum_{i=1}^{n} c_{0i}^{a} x_{0i}^{a} \leq C_a; a \in \mathcal{S} \tag{3.21}$$

$$\sum_{a=1}^{m} y_{ia} = 1; i \in \mathcal{N} \tag{3.22}$$

$$A_i \leq L_i; i \in \mathcal{N} \tag{3.23}$$

$$x_{ij}^{a} \in \{0, 1\}; i, j \in \mathcal{N}, a \in \mathcal{S} \tag{3.24}$$

$$y_{ia} \in \{0, 1\}; i \in \mathcal{N}, a \in \mathcal{S} \tag{3.25}$$

$$1 \leq u_i \leq n_r; i \in \mathcal{N} \tag{3.26}$$

$$u_i - u_j + (n_r - m) x_{ij} \leq n_r - m - 1; i, j \in \mathcal{N}, i \neq j \tag{3.27}$$

In the above formulation, constraint (3.19) states that the recharge tour for each MC starts at initial position 0. Constraint (3.20) ensures the connectivity of the path and every vertex is visited at most once. Constraints (3.21) and (3.22) guarantee the MC's capacity is not violated and each vertex is visited by only one MC. Constraint (3.23) guarantees the arrival time of the MC is within each sensor's residual lifetime. Constraints (3.24) and (3.25) impose x_{ij} and y_{ia} to be 0-1 valued. Constraints (3.26) and (3.27) eliminate the subtour in the planned route, which is formulated

according to [57]. The classic TSP with Profits can be considered as a special case of CaMP-BaD with unlimited capacity and unspecified deadlines. Since TSP with Profits is well known to be NP-hard [51], CaMP-BaD is also NP-hard.

A direct solution to the CaMP-BaD is difficult to obtain and rare in existing literature. Hence, we review some literature that has partially solved the problem. In [75], a survey of different approaches to TSP with profits is presented. Lagrangian decomposition method and approximated algorithms developed based on existing solutions can provide solutions very closed to optimality. However, adding capacity (Eq. (3.21)) and time (Eq. (3.23)) constraints makes the problem more complicated. A great deal of research efforts on these two constraints are devoted in the context of Vehicle Routing Problem, in which a number of vehicles start from a depot to visit client locations and the objective is to minimize the total traveling cost of the vehicles. In [60], the Capacitated Vehicle Routing Problem where each vehicle has a fixed capacity is considered. Time constraints are studied in Vehicle Routing Problem with Time Windows[62]. A local search algorithm is proposed in [62] based on the k-exchange concept, and reduction of the computation for checking feasibility constraint is also studied. A theoretical approach to obtaining $3 \log n$-approximation algorithm is sought in [63] (n is the number of nodes). However, subroutines from existing solutions visit the node with the smallest deadline last, which contradicts to our problem where such nodes should be serviced earlier.

Due to the nature of our problem, it is not realistic to use standard optimization techniques[60, 75] because these methods deal with datasets of static inputs and the optimization is usually done offline through a one-time effort. In contrast, energy consumption in our framework is probabilistic in nature. A charging vehicle's recharge capacity declines after recharging sensor nodes, so the input to our problem is more dynamic than that most existing solutions have considered. Furthermore, existing algorithms require high computation power that may not be available on charging vehicles. Therefore, we need to design algorithms suitable to our problem context. Next, we present two such algorithms.

3.5.2 Greedy Algorithm (GA)

The simplest approach is a greedy algorithm which selects the node with the maximal recharge profit (i.e., recharge reward less traveling cost) for each node selection.

After an MC finishes recharging a node, it picks the next available node with the maximal profit. When the its energy falls below a threshold χ, it returns to the base station for battery replacement and then resumes recharge in the same fashion.

Despite of its simplicity, GA may have some problems in practice. The first problem is that the MC might move back and forth over long distances, thereby increasing the traveling energy cost. This happens when the node with the maximum profit lies faraway, and the energy efficiency of MCs can deteriorate. Second, because the only consideration is profit, it may not fulfill a recharge request in a fixed time. These observations offer us room for further improvements. To prevent charging vehicles from traveling long distances, we can confine the scope of movements by partitioning the network into several regions adaptively and assigning each charging vehicle to one of the regions. Second, a more sophisticated scheduling method should be developed to capture MCs' capacity as well as sensors' battery deadline constraints. In the next subsection, we will introduce an Adaptive Algorithm (AA) to address the limitations in GA.

3.5.3 Recharge by Adaptive Algorithm (AA)

Adaptive Network Partitioning

In the first step, the base station requests sensor nodes for energy information periodically using the method discussed in the previous chapter. Then it adaptively partitions the network into m *regions* according to the originating locations of requests. The result of partitions is disseminated to the MCs using long range radio. We utilize the well-known K-means algorithm to perform the partition[66]. Using the K-means algorithm would allow the MCs to adaptively select a subset of nodes with their square sum of distance minimized regarding to the centroid of each region so the MC would only move in a confined scope, and most likely with less distances. For each region, our objective is to minimize the intra-region square sum of inter-node distance.

$$S = \sum_{j=1}^{m} \sum_{i=1}^{n_r} \|n_i^{(j)} - \mu^{(j)}\|^2 \tag{3.28}$$

in which $\|n_i^{(j)} - \mu^{(j)}\|^2$ is the square distance between a recharge node n_i of region j to the region's centroid $\mu^{(j)}$ (computed by taking the mean of x, y coordinates of

all the nodes in the region). Now we briefly explain the partitioning process.

Initially, we select a number of m sensor nodes with the minimum lifetime from $\mathcal{N}$ to be the centroid of regions. We assign each node to the closest centroid. After all the nodes have been associated with a centroid, we re-calculate centroid positions taking the average value of x and y coordinates of nodes in the region. This process is repeated until the centroids no longer change. After the partition, the centroid of each region represents a virtual position that has the minimal sum of distances to all the nodes in its region. This position can be used as the starting position for the MC to recharge nodes in its region.

Generating Capacitated Minimum Spanning Tree

In the first step, m regions are generated and each charging vehicle only needs to take care of the nodes in its region. To decide the route to recharge sensor nodes, we need to ensure each MC's recharge capacity is not exceeded (Eq. (3.21)). At the same time, we also want to minimize the traveling energy cost for the MC. These requirements lead to finding Capacitated Minimum Spanning Tree (CMST)[67] where the total sum of demands from nodes does not exceed the MC's capacity and the minimum traveling energy cost can be found by constructing the minimum spanning tree. In this way, we can ensure sensor nodes closed to each other are placed in the same tree and later covered by the same recharge route.

The exact solution to CMST requires us to go over all possible tree setups and pick the one with the lowest cost, which involves exponential computation. Fortunately, an efficient algorithm by Esau-Williams(EW) can find a suboptimal solution very close to the exact solution in polynomial time [67]. The EW algorithm merges any two subtrees when there is a "saving" in the total cost of two trees.

Nevertheless, there are some limitations of the original EW algorithm when applied for our problem. First, when determining whether two subtrees can be merged, only the demands from sensor nodes are counted whereas the traveling costs on edges are not considered. Second, multiple such trees can be generated. How does the MC decide which tree to pick? To overcome these limitations, we extend the original EW algorithm. As mentioned earlier, a deterministic upper bound on the shortest tour length is developed as $\sqrt{2(n-2)ab} + 2(a+b)$ for a rectangle of side length a, b and n nodes. For the square sensing field with L side

length and subtree of n_b nodes, we have a loose upper bound on the traveling cost, $(\sqrt{2(n_b-2)}+2)Le_c$. Second, when multiple trees are generated, we select a tree that maximizes the ratio of total energy demand to traveling cost. In this way, we can exploit limited resources on MCs better and improve energy efficiency of the network.

Next, we explain our extension to the EW algorithm in detail. Each MC computes CMST independently by iteratively updating a distance matrix. The distance matrix facilitates the computation process by maintaining costs of tree nodes. Let us denote recharge set $\mathcal{N}_a$ with n_a nodes for MC a ($\bigcup_{a=1}^{m}\mathcal{N}_a = \mathcal{N}_r$). We define trade-off function t_i for each node in its recharge set $\mathcal{N}_a$, $t_i = \min(c_{ij}^{(a)}) - c_{0i}^{(a)}$ and $j \in P_i$, where P_i is the neighboring set of node i, $\min(c_{ij}^{(a)})$ finds the minimum cost from node i to its neighbor j in P_i and $c_{0i}^{(a)}$ is the cost from node i to MC's starting position (i.e., the root)[1]. The trade-off function evaluates whether it is beneficial to merge subtrees of nodes i and j. A positive t_i indicates that it incurs smaller cost for the MC to directly travel from the root to node i so merging subtrees of nodes i and j is not preferred. A negative t_i indicates how much it can be saved by connecting subtrees of i and j. Thus the most negative t_i results in the most savings in an iteration.

After t_i has been computed, we search through all trade-offs $t_i(\forall i = 1, \ldots, n_a)$, looking for the minimum trade-off (i.e., the most negative value). Assume t_k is the most negative trade-off and j is k's minimum cost neighbor. To capture MC's capacity constraint in Eq. (3.21), if the sum of total demands from the subtrees of k and j plus upper bound of their traveling cost is less than the recharge capacity (which means we can cover the subtrees of k and j under the current recharge capacity), we merge the subtrees of k and j. Since the action of merging k and j has resulted in a lower total traveling cost to k, direct traveling from the root to reach k has higher cost and should be avoid. So we remove the edge from node k to the root by setting $c_{0k}^{(a)}$ in the distance matrix to ∞.

At this point, two subtrees satisfying the recharge capacity with minimum sum of cost have been merged, and we need to update the minimum cost of the newly merged tree to the root. It is done by updating the minimum cost in the distance matrix from the tree to the root by setting the value to $\min(c_{0i}^{(a)})$, where i is the node

[1]In order to reduce intra-region traveling cost, we set the centroid output from K-means algorithm to be the root.

in the newly merged tree.

On the other hand, if merging subtrees of k and j violates MC's recharge capacity, we need to restrict any further actions to merge j to k because these two trees cannot be covered by the MC in a single run. Then we recompute the trade-off function t_k to search for the next neighboring node that results in minimum trade-off until the next valid neighboring node j is found and merged to the existing trees. The iteration continues until all the trade-offs become nonnegative, in other words, no more saving can be made.

After the CMST has been generated, the MC selects a tree with the maximal ratio of recharge demand to sum of tree's edge cost and utilize the route improvement algorithm to form the final recharge sequence among the tree nodes. After the MC finishes recharging nodes in a tree, it checks whether its energy falls below a threshold. If so, it returns to the base station for battery replacement. Table 3.2 shows the pseudo-code of our extended EW algorithm.

Insertion Algorithm for Route Improvement

After the CMST has been obtained, next we want to produce a recharge sequence for nodes such that for each node the MC arrives before its battery deadline. Let us denote the result from CMST to be a recharge node set $\mathcal{N}_r^{(a)}$ ($\mathcal{N}_r^{(a)} \subseteq \mathcal{N}_a$). Recall that if the condition in Proposition 2 is satisfied, a node can be placed anywhere in the recharge sequence. We call such a set of nodes a *feasible node set* $\mathcal{N}_f^{(a)}$. Otherwise, a node may need prioritized treatment to meet its battery deadline. We denote such a set of nodes as a prioritized set $\mathcal{N}_p^{(a)}$ ($\mathcal{N}_f^{(a)} \cup \mathcal{N}_p^{(a)} = \mathcal{N}_r^{(a)}$).

Intuitively, we first use a Traveling Salesman Problem algorithm (e.g., the $\mathcal{O}(n^2)$ nearest neighbor heuristic algorithm[69], where n is the number of nodes) to find a feasible solution as the initial sequence Ψ for nodes in the feasible set $\mathcal{N}_f^{(a)}$. Then we insert nodes from the prioritized set $\mathcal{N}_p^{(a)}$ into Ψ while ensuring the battery deadline in Eq. (5.8) for all nodes are still met. To this end, we sort the nodes in $\mathcal{N}_p^{(a)}$ in a descending order of residual lifetimes and denote the sorted sequence as Ω. We insert these nodes starting from the first node Ω_1. Let A_i denote the arrival time of the MC at the i-th node in the shortest path Ψ, $i = \{1, 2, \ldots, n_f^{(a)}\}$.

To insert the j-th node Ω_j from Ω into Ψ, we first find position m_t in Ψ such that $A_{m_t} \leq l_{\Omega_j}$ and $A_{m_t+1} > l_{\Omega_j}$ where l_{Ω_j} is Ω_j's lifetime. We call m_t the *temporary*

Table 3.2: Extended Esau-Williams Algorithm

input: recharging node set $\mathcal{N}_r$, distance matrix $D^{(a)}$,
recharge capacity C_a, demand of nodes d_i, $i \in \mathcal{N}_a$.
output: CMST nodes need to recharge.
Initialize $t^{(a)} < 0$, weight of tree, $C^{(a)} = 0$.
while ($t^{(a)} < 0$)
Find neighbor m_i of i results min cost, $\min_{m_i} D^{(a)}(i, m_i)$.
Compute trade-off value list $t_i^{(a)} = D^{(a)}(i, m_i) - D^{(a)}(1, i)$.
Find k and j resulting most negative trade-off value,
$k \leftarrow \min_i(t^{(a)}), j \leftarrow m_k$.
do
Add new nodes $N_{new} \leftarrow k + j$ if not exist in current trees
if weight of merging subtree of $N_{new} < C_a$
Add N_{new} to corresponding tree i
Update cumulative weight of corresponding tree i, $C_i^{(a)}$.
Declare N_{new} is accepted.
else
Update $D^{(a)}(k, j) \leftarrow \infty$
Search for next min cost neighbor for k,
$m_k \leftarrow \min_{m_k} D^{(a)}(k, m_k)$.
Recompute trade-off for k, $t_k^{(a)} = D^{(a)}(k, m_k) - D^{(a)}(1, k)$.
Declare N_{new} rejected.
until (N_{new} is accepted) or (all $t_i^{(a)} \geq 0$)
end while
Select a tree results maximum energy efficiency.

maximum position to insert Ω_j. It indicates the maximum number of nodes in Ψ that can be served before node Ω_j depletes its battery. To accommodate the remaining $|\Omega| - j$ nodes, we need to find a position such that even all the remaining nodes are inserted before Ω_j, Ω_j can still meet its battery deadline. We find the maximum position m such that $A_m \leq A_{m_t} - \sum_{i=j+1}^{n_p^a} t_i$ and $A_{m+1} > A_{m_t} - \sum_{i=j+1}^{n_p^a} t_i$, where t_i is the recharge time of Ω_j. Now, the maximum position m represents the rightmost position Ω_j can be inserted if all remaining nodes are later inserted before Ω_j.

For each of the m possible positions that Ω_j can be inserted, a total traveling cost is computed and the one that minimizes the traveling cost is selected as the final insertion position for Ω_j. Then we obtain a new sequence Ψ and remove Ω_j from Ω. The iteration continues until we exhaust Ω or an infeasible situation is encountered. Table 3.3 shows the pseudo-code of the insertion algorithm.

We briefly illustrate how the insertion algorithm works in Fig. 3.5. We consider two nodes Ω_1, Ω_2 with lifetime 104 mins and 90 mins that need to be inserted into a feasible recharge sequence. We find the position k to insert Ω_1 is between node 6 and 7 since $A_6 < l_{\Omega_1} < A_7$. To ensure Ω_1 can survive when Ω_2 is later inserted before Ω_1, k' can only be between node 3 and 4 (since $A_3 < A_6 - T_{\Omega_1} < A_4$). Then we search all the 4 possible locations (before node 1, 2, 3, 4) and find that the position before node 3 minimizes the traveling cost. Thus Ω_1 is inserted between node 2 and 3. We repeat the procedure for Ω_2. Since it is the last node, we can directly calculate the rightmost insertion position k' and find the minimum cost among possible inserting positions.

3.5.4 Complexity Analysis

We now analyze the complexity of our algorithms. The complexity of the greedy algorithm is $\mathcal{O}(n)$ because it only selects the maximum profitable node at each step. For the adaptive algorithm, the base station has abundant resources and it performs the k-means algorithm. So we focus on the computing burdens on MCs for calculating CMST and route improvements. In the worst case, there is only one MC to recharge n nodes. For the extended EW algorithm, finding the minimum trade-off value requires $n^2 + 2n$ iterations at the outer loop. In the inner loop, the worst case is that for a node with the minimum trade-off value, every minimum-cost neighbor is rejected due to capacity violations. So n iterations are required. In sum,

Table 3.3: Insertion Algorithm

input: CMST $\mathcal{N}_r^{(a)}$, lifetime l_i and recharge time t_i, $i \in \mathcal{N}_r^{(a)}$, distance matrix $D^{(a)}$, feasible set $\mathcal{N}_f^{(a)}$ satisfying Proposition 2.
output: resultant recharge sequence Ψ.
Compute shortest path in the feasible set, $\Psi \leftarrow$ TSP($\mathcal{N}_f^{(a)}$)
Sort $\mathcal{N}_p^{(a)}$ in a descending order of lifetime as Ω
Initialize $i \leftarrow 1$, last step node position $k \leftarrow \infty$.
while $\Omega \neq \emptyset$
 Find temporary max position m_t in Ψ such that
 $A_{m_t} \leq l_{\Omega_i}$ and $A_{m_t+1} > l_{\Omega_i}$
 Find the max insertion position m such that
 $A_m \leq A_{m_t} - \sum_{k=i+1}^{n_r^p} t_k$ and $A_{m+1} > A_{m_t} - \sum_{k=i+1}^{n_r^p} t_k$.
 if Cannot find $m \geq 0$. **break**,return infeasible and report.**end if**
 Set minimum cost $c_{min} \leftarrow \infty$.
 for x from 0 to m
  Insert node Ω_i into Ψ, get temporary sequence Ψ_t
  Calculate cost $c \leftarrow \sum_{j=1}^{|\Psi_t|-1} D^a(j, j+1)$.
  if $c < c_{min}$, $\Psi \leftarrow \Psi_t$, $c_{min} \leftarrow c$, $k \leftarrow x$. **end if**
 end for
 $i \leftarrow i + 1$, update $\Omega \leftarrow \Omega - i$
end while
Return recharge sequence Ψ, minimum cost c_{min}.

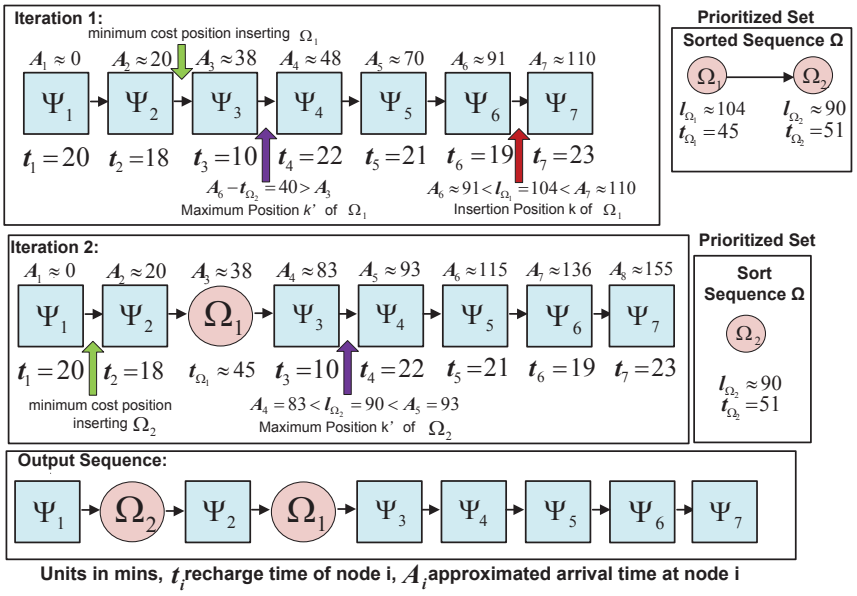


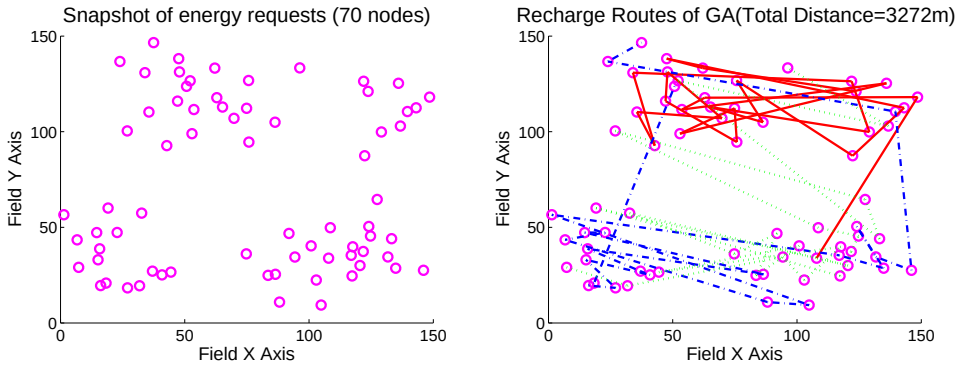
Figure 3.5: Illustration of insertion algorithm.

its time complexity is $\mathcal{O}(n^3)$.

For the route improvement algorithm, running a TSP algorithm requires $\mathcal{O}(n^2)$ time. Sorting nodes' lifetimes requires $\mathcal{O}(n \log n)$ time. Then, insertion requires $\mathcal{O}(n^2)$ time. Hence, the total time complexity of route improvement algorithm is $\mathcal{O}(n^2)$ and the adaptive algorithm takes $\mathcal{O}(n^3)$ time. Note that although the proposed algorithms are centralized, they run on the MCs that usually have much higher computation and energy resources than common sensor nodes. It is not difficult for them to handle computations for large networks.

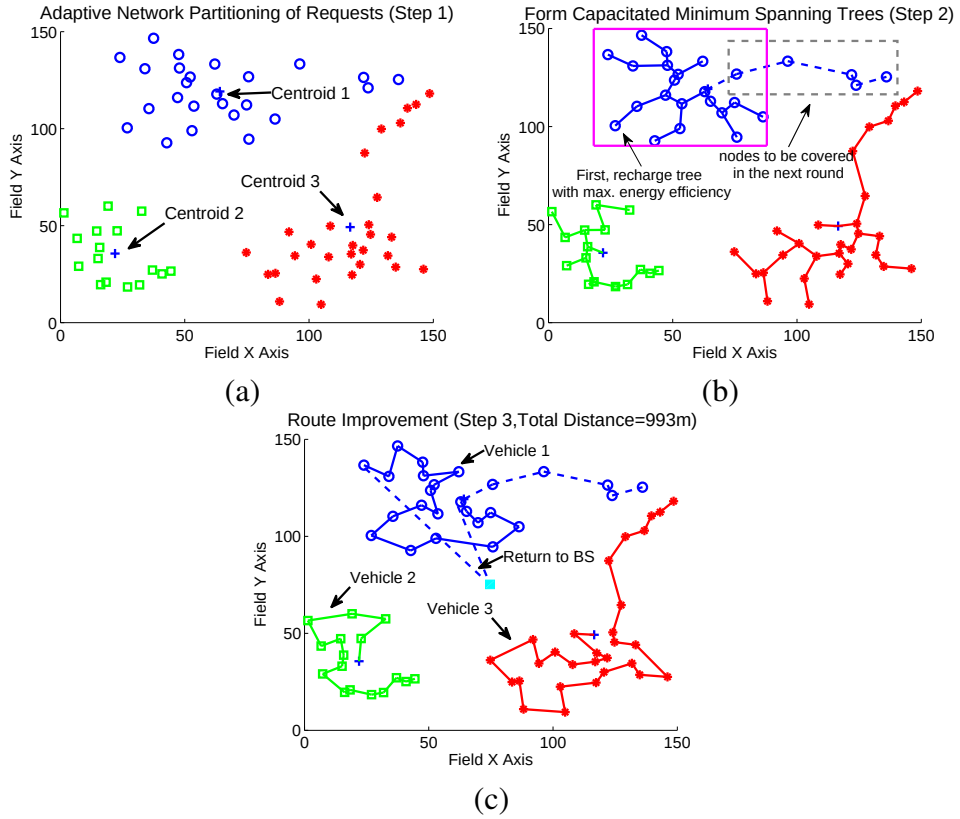
3.5.5 An Example of Algorithms

To illustrate operations of the algorithms, we show an example in Fig. 3.6-Fig. 3.7. A snapshot of 70 recharge requests from sensors during the operation is presented in Fig. 3.6(a) when three MCs cooperate to recharge these nodes. Fig. 3.6(b) demonstrates the recharge routes using the Greedy Algorithm with a total distance of 3272 m. We can see the MCs travel long distances to take care of energy requests in the field, which matches our analysis in Section 3.5.2. Fig. 3.7(a) shows an adaptive network partitioning of the recharge requests into three regions. Then the MCs compute the CMST in parallel fashion in Fig. 3.7(b). Note that two trees are



(a) (b)

Figure 3.6: An example of the Greedy Algorithm (a) a snapshot of recharge request. (b) recharge routes from the Greedy Algorithm.



(a) (b) (c)

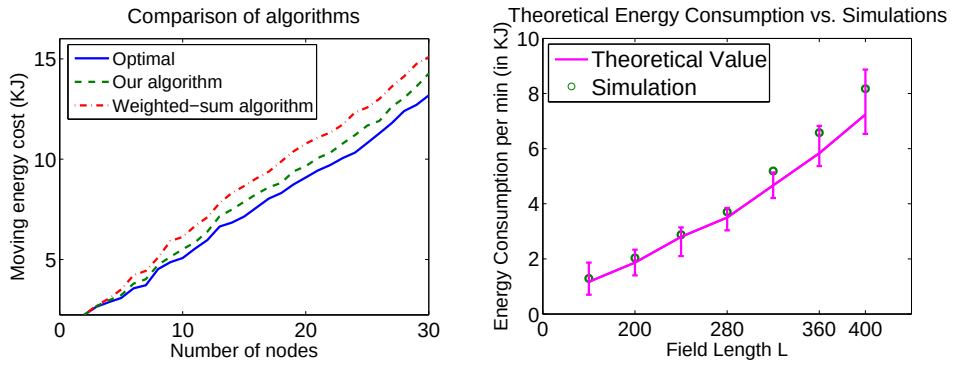
Figure 3.7: An example of the Adaptive Algorithm (a) adaptive network partitioning regarding recharge request. (b) establish CMST (c) improve recharge route.

generated for MC 1 due to limited recharge capacity. The tree with higher ratio between energy demands and sum of edge costs is chosen first. The uncovered nodes will be recharged in the next round after the MC has replenished its own battery at the base station. Next, each MC calculates an improved recharge route on the selected tree shown in Fig. 3.7(c). MC 1 has to return to base station for battery replacement before recharging nodes on the second tree (edges shown as dashed line). In contrast to the Greedy Algorithm, MCs only travel a distance of 993 m which suggests great potentials of the Adaptive Algorithm to reduce system cost.

3.6 Performance Evaluations

We have developed a discrete event-driven simulator using POSIX multi-thread programming in C language. In our simulator, packet transmissions between nodes are modeled by inter-thread communications and each MC also calculates the recharge decisions by exchanging information. To model WRSNs with high accuracy, the simulator takes real parameters such as battery recharge times.

A number of $N = 500$ sensor nodes are uniformly randomly deployed over a square sensing field with side length $L = 160$ m. All sensors transmit at the same power level with fixed transmission range $d_r = 15$ m. The choice of maximum cluster hop-count k will have a direct impact on energy consumption and data gathering latency. On one hand, a large k would result in large intra-cluster energy consumptions due to more traffic relays, especially on anchor points which aggregate all the packets. This would potentially increase the load on MCs. On the other hand, a small k will generate more clusters. To cover all the nodes, the data collection tour would be elongated and cause higher latency. Through trials we find that when $k = 3$, $c \approx 5$ clusters are needed to cover the entire field, and the intra-cluster energy consumptions are not too large. Thus we set $k = 3$. Dijkstra's shortest path algorithm is used to route packets from sensors to their corresponding anchor points at an average rate of $\lambda = 3$ pkt/min and 30 bits per packet following a Poisson process. Each time slot is 1 min. The bit rate is 250 Kbps. Since a higher initial energy takes longer time for the network to achieve equilibrium, we set all sensors to start from 50% battery initially to make the network enter equilibrium faster. The MCs collect energy information every 12 hours and each time it finishes fulfilling all the energy requests.



(a) (b)

Figure 3.8: Evaluation of algorithms and validation of theoretical model (a) comparison of different algorithms (b) validation of energy consumption model.

Sensor nodes have adaptive recharge thresholds regarding their communication hop counts to anchor points following Eq. (3.17). Given $\tau_1 = 0.75$, we can calculate $\tau_2, \tau_3 = 0.57, 0.22$, respectively. The battery's recharge time is modeled from [46]. We assume MCs are electric-powered vehicles carrying computing, communication modules and high density battery packs (e.g., 12A, 5V standard ones). The MC can weight tens of pounds and we assume it is 20 lbs. Using the method in [76], we estimate that the MC consumes energy at a rate of 5.59 J/m. To evaluate how the number of MCs affects system performance as well as validate theoretical results in Proposition 1, we vary the number of charging vehicles m from 1 to 5 and set the simulation time to 4 months.

3.6.1 Evaluation of Algorithm and Energy Consumption Model

We first evaluate the performance of the adaptive algorithm by comparing with the optimal solution and *weighted-sum algorithm* proposed in [45]. The weighted-sum algorithm finds the shortest recharge sequence based on traveling time and residual lifetime of sensor nodes through a weighted parameter. It tries different weighted parameters and chooses the best solution among all the trials. Both the weighted-sum and adaptive algorithms aim to capture the battery deadline constraints.

Due to the NP-hardness of our problem, it is very difficult to obtain optimal solutions using brute force for large networks. To provide a baseline for comparison, we have managed to obtain optimal solutions for networks up to 30 nodes by prun-

ing solutions that lead to infeasibility or sub-optimality. We set the residual energy of sensor nodes uniformly randomly distributed from zero to 20% and compare different approaches that form recharge routes through all the nodes. The simulation results are averaged over 100 datasets. Fig. 3.8(a) shows the moving energy consumption of charging vehicles. We can see that for a small network size (1-5 nodes), the gaps between our adaptive algorithm and the optimal solution is small. This is because the number of different possible schedules is small. Our algorithm may find the optimal schedule, or one very close. What is interesting is that the ratio remains almost the same as we increase the number of nodes. The maximum ratio of 1.10 appears when the number of nodes is 14. On average, the ratio is 1.065 to the optimal solution, which offers a good approximation. This shows our algorithm can still find schedules very close to optimal even when the search space has grown dramatically. For the weighted-sum algorithm, the maximum ratio is 1.22 when we have 8 nodes, and the average approximation ratio is 1.16. The results indicate that the adaptive algorithm saves an additional 8% energy cost compared to the weighted-sum algorithm. Besides, the selection of weight parameter in [45] may not be easy in real applications. The adaptive algorithm utilizes an existing solution from the TSP problem without the complexity to examine various weight values.

We also evaluate the correctness and accuracy of the energy consumption model shown in Fig. 3.8(b). To examine our model over different network field sizes, we first set $N = 500$, $L = 160$m (node density $\rho = 0.019$ nodes/m^2) and increase L from $160 - 400$ m while keeping node density the same. The theoretical results show the average energy consumptions with variations along the curve. That is, if we use the lower bound of Eq. (3.1) to calculate the number of anchor points, we have a lower limit for the energy consumption. On the other hand, if we count anchor points according to actual layouts governed by Eqs. (3.3), (3.4) and (3.5), an upper bound on energy consumption is derived (it overestimates partial clusters on the boundaries). It is observed that our energy consumption model can achieve very high accuracy (falls within theoretical variations). For $L = 160 - 280$, the simulation results almost match our theoretical model and for $L = 320 - 400$m, the simulation results are within 15% of the average theoretical numbers. The inaccuracies are due to an increasing number of clusters on the field boundaries, which are not complete circles causing overestimates. Next, we will validate the entire

theoretical model on the minimum number of MCs.

3.6.2 Evaluation of Network Performance

In this subsection, we evaluate the performance of proposed algorithms in terms of the number of nonfunctional nodes, energy consumption vs. replenishment, recharge fairness, duration of nonfunctional nodes, data collection latency and operating energy cost.

Nonfunctional Nodes

First, we examine the evolution of the number of nonfunctional nodes. When a sensor node depletes its battery energy, it becomes nonfunctional until recharged. Fig. 3.9 presents the results of nonfunctional nodes by proposed algorithms.

For the Greedy algorithm (GA), when $m = 1$, the number of nonfunctional nodes surges dramatically around 18 days to over 80% until it slowly decreases and stabilizes at 55% around 37 days. Similar phenomenons are observed for $m = 2, 3$. This is because the MCs favor nodes closer to the anchor points with more recharge profits. Thus they do not serve nodes in the outmost corona of clusters fast enough after their requests. MCs only cover them when their batteries nearly deplete. By then, their recharge capacity ($m = 2$) is temporarily exceeded, which causes the big spike. Although $m = 1 - 3$ MCs can gradually resolve most nonfunctional nodes, it is observed that there is persistently more than 50%, 20% and 10% nonfunctional nodes for $m = 1, 2, 3$, respectively. In contrast, the Adaptive Algorithm (AA) provides more stability. When $m = 2, 3$, there is no such huge spike. For $m \geq 3$, nonfunctional nodes are within 10% at network equilibrium. This is because AA captures the sensor battery deadlines. When $m = 5$, AA can reduce the nonfunctional nodes to zero.

We observe that $m = 5$ is likely to be a threshold since 4 MCs still result in sporadic 5% nonfunctional nodes. From *Proposition 1*, after plugging in the experimental parameters, we obtain $m = \lceil 4.72 \rceil = 5$. Thus $m = 4$ can barely satisfy the energy neutral condition. This calculation matches our observations in Fig. 3.9(b), validating the correctness of our theoretical results.

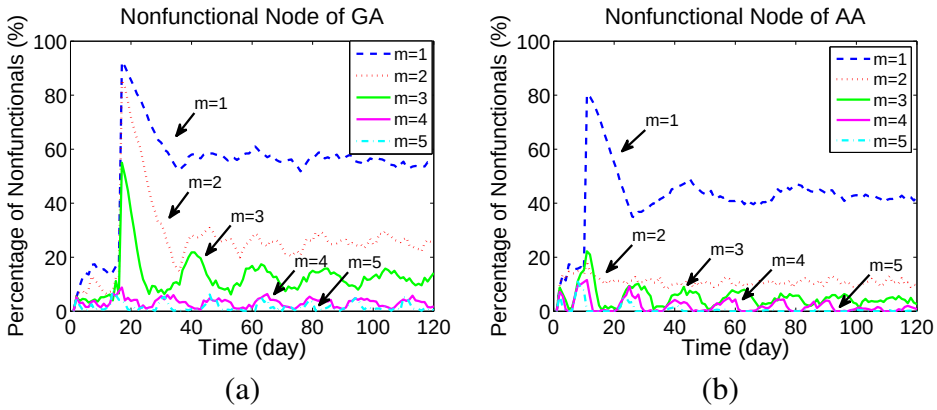
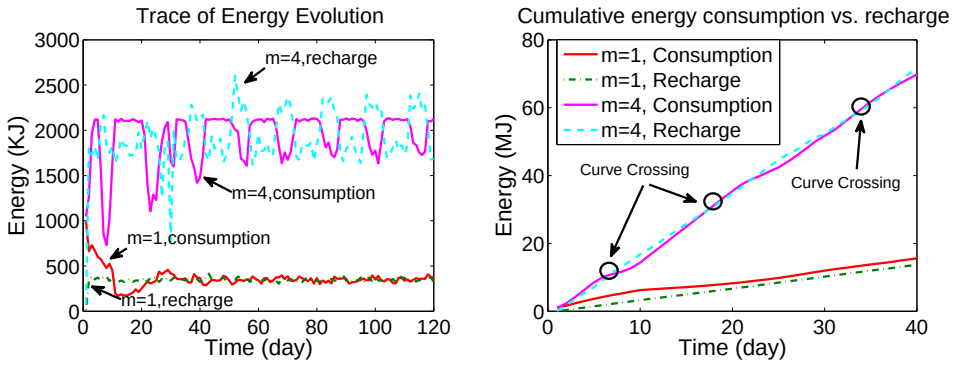


Figure 3.9: Evolution of nonfunctional nodes. (a) GA. (b) AA.

Energy Consumption vs. Recharge

In this subsection, we demonstrate the evolution of energy consumption vs. replenishment. Since GA and AA have similar curve shapes, we illustrate the energy changes of AA only. In Fig. 3.10(a), we trace the evolution of consumed and replenished energy when $m = 1, 4$. For $m = 1$, it is definitely not enough to sustain network operations. Thus nodes continuously deplete battery and no longer consume energy, which causes the drop in energy consumption at the very beginning. Since the recharge capacity of one vehicle puts an upper limit on the energy consumption, the two curves reach an equilibrium and converge after 30 days. For $m = 4$, about 4 times the energy is replenished compared to $m = 1$, thus the large gap in between. We also observe that when there is a drop in energy consumption, the energy replenishment correspondingly jumps up, which represents four vehicles acting in response to to battery depletions.

To illustrate energy balance in the network, we also show the cumulative energy evolution in Fig. 3.10(b). For clarity and better observing the gaps and intersections between curves, we plot 40 days' simulation time. If the energy replenishment curve is above the consumption curve, more energy has been refilled into the network than consumed, and vice versa. For $m = 1$, the energy consumption curve is above the energy replenishment curve. A larger gap is observed at the first 10 days, indicating energy replenishment can barely keep up with consumptions. In contrast, with $m = 4$, the energy consumption curve stays above replenishment until the two curves first cross each other around 6 days. This is because from the very begin-



(a) (b)

Figure 3.10: Energy consumption vs. replenishment of AA. (a) trace of energy evolution. (b) cumulative energy consumption vs. replenishment (40 days).

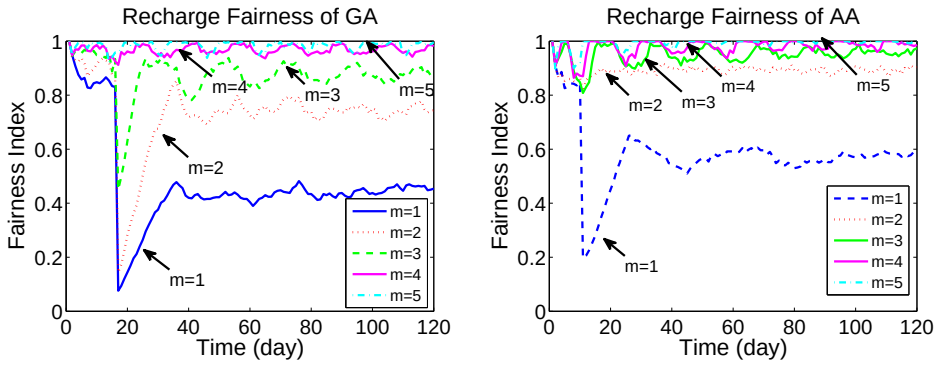
ning, more energy is consumed than replenished. Around 6 days, a few nodes have depleted energy and stopped consuming more, which brings down the consumption curve. The replenishment curve stays above the consumption curve until the next crossing around 20 days. Therefore, the evolution of network energy also validates $m = 4$ is a threshold case since sporadic battery depletions are observed.

Recharge Fairness

Recharge fairness indicates whether charging vehicles recharge nodes commensurate to their workloads. Those having more workload (e.g., nodes near the base station) should be recharged more frequently. This is reflected from the functional time of sensor nodes. To quantify recharge fairness, we leverage the fairness index from [77],

$$F = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} (x_i)^2}, \tag{3.29}$$

in which x_i is a normalized indicator whether a node is functional in a time slot. x_i equals $1/N_s$ if i is functional in a time slot, otherwise, it is zero. The fairness index F ranges from 0 (worst case if all nodes are nonfunctional) to 1 (best case if all the nodes are functional). In Fig.3.11(a), when nodes in the outmost ring become nonfunctional, the fairness of GA algorithm severely degrades as vehicles only recharge nodes with maximum profits. We can see from Fig.3.11(b) that AA can distribute energy resources fairly among the nodes especially when $m = 4, 5$



(a) (b)

Figure 3.11: Comparison of recharge fairness. (a) GA. (b) AA.

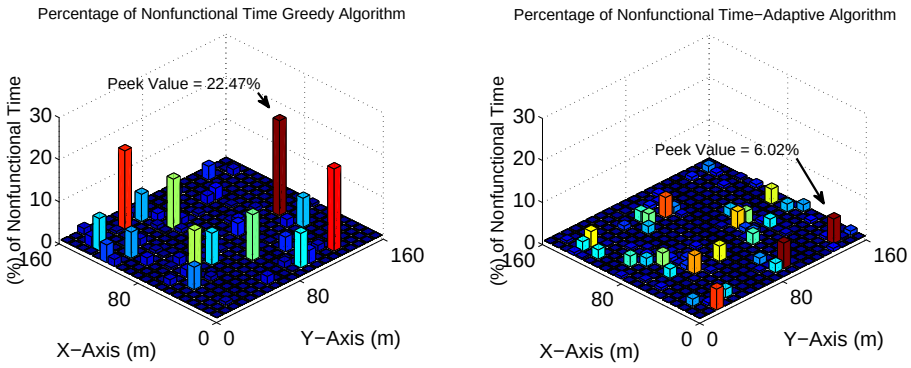
($F = 1$).

Nodes' Nonfunctional Periods

Fig. 3.12 plots the percentage of nonfunctional durations of nodes as a function of their locations. Using GA, nodes near the anchor points have a maximum of 22.47% time in nonfunctional states whereas AA is only 6.02%. Further, AA spreads nonfunctional durations across the field while the spikes of GA are highly concentrated around anchor point locations. This is because nodes close to anchor points consume energy faster and are more prone to become nonfunctional. GA considers profit only and has no measure for battery deadlines. In contrast, AA considers both profit and battery deadlines. Therefore, the duration of nonfunctional nodes with AA is significantly less than that of GA.

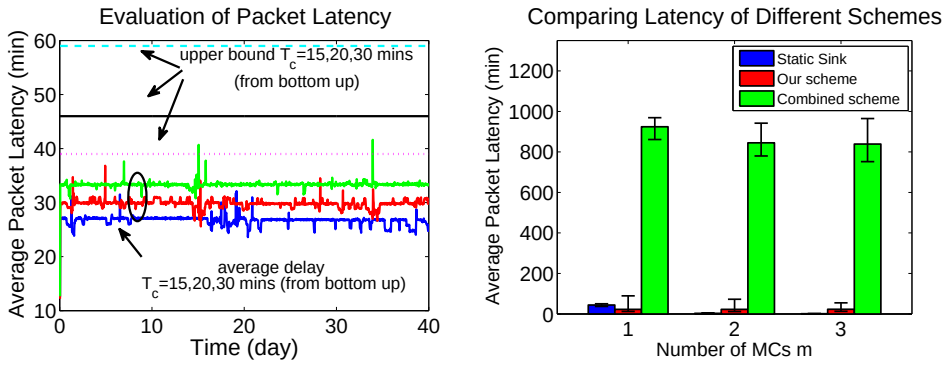
Data Collection Latency

Data collection latency mainly depends on two variables: dispatching time interval T_c and availability of routing paths. The former is a system parameter determining how often to dispatch the data gathering vehicle; the later relies on the number and locations of nonfunctional nodes. To transmit packets to anchor points timely, all nodes should be functional on a routing path. We assume shortest routing paths by Dijsktra's algorithm are used. If a node depletes energy and there is no alternate route available, all pending messages are buffered at senders until the path is restored.



(a) (b)

Figure 3.12: Comparison of durations for nonfunctional nodes when $m = 4$. (a) GA. (b) AA.



(a) (b)

Figure 3.13: Evaluation of data collection latency (a) Latency using different T_c vs. upper bound. (b) Comparison of latency between different data collection schemes.

To see the fluctuations of curves more clearly, we trace the evolution of data collection latency for the first 40 days in Fig. 3.13(a). We vary $T_c = 15, 20, 30$ mins, resulting in average packet latencies of 27,30,34 mins respectively. The small spikes are caused by temporary unavailability of routing paths when packets are buffered for longer time. We have also plotted corresponding data collection upper bounds calculated in *Lemma 2* and we observe that the actual packet latencies are well within these bounds. It is interesting to see when $T_c = 15$ mins, the latency is 27 mins whereas when $T_c = 30$ mins, the latency only increases slightly to 34 mins. This is because data transmission time and MC's moving time dominate when $T_c = 15$ mins. This indicates that sending out the data gathering vehicle too frequently may not help reduce packet latency too much compared to the extra operating costs incurred.

In addition, we have also compared packet latency between different data collection schemes. A static data sink is used in [42, 71] to gather all the packets and we denote it as "Static". A combination of data sink and wireless charging on a single vehicle is proposed in [41] and we denote it as "Combined". Fig. 3.13(a) compares the average packet latency when we increase the MCs from 1 to 3. First, we can see both the static and our schemes have about two orders of magnitude less latency than the combined scheme. The large latency of the combined scheme is caused by the inevitable gap between battery recharge time and data transmission time. The delivery of gathered data has to wait for at least 10 hours until the MC returns to the base station for battery replacement. On the contrary, our scheme employs a dedicated vehicle without any waiting for recharge. Second, although the static scheme is expected to yield less latency than our scheme (when $m = 2, 3$), it has a higher latency when $m = 1$. Since using a static sink results in more traffic relays, thus higher energy consumption. When there are not enough MCs, nonfunctional nodes lead to unavailability of routing paths and longer latencies.

Operating Energy Cost

In this subsection, we evaluate the traveling energy cost of MCs. Fig. 3.14(a) compares the average traveling cost per vehicle for GA and AA. When $m = 1 - 3$, more energy cost is observed with AA. This is because the AA takes care of nodes in the outmost corona preemptively before they deplete energy, thus more energy

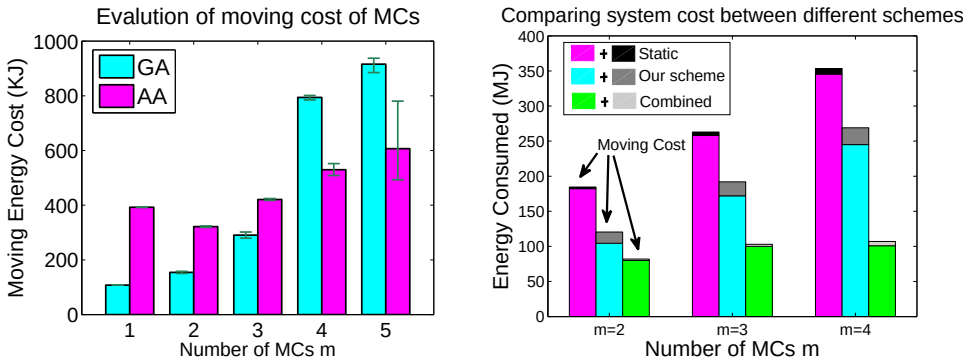


Figure 3.14: Evaluation of operating energy cost. (a) Comparing MC's moving cost between GA and AA. (b) Comparing total system cost between different data collection schemes.

is used in traveling. With GA, charging vehicles travel to the outmost corona only when recharge profits there are larger, but by then those nodes nearly deplete energy and many become nonfunctional. Although GA has lower traveling cost when $m = 1 - 3$, the network performance deteriorates greatly. When $m = 4 - 5$, we can partition the network into more regions with smaller sizes so the movements of MCs can be confined in smaller regions. This brings down the movement energy for MCs. However, as GA does not partition the network, long distance travels are inevitable. So AA incurs less energy with more MCs.

Fig. 3.14(b) shows the total system cost on MCs for different data collection schemes. For fair comparison, we set the communication hop count $k = 3$ in both our scheme and the combined scheme in [41]. The main body of the bar charts are energy used for recharging sensor nodes and the dark portion on top represents the total moving energy cost on vehicles. First, we can see the static scheme used in [71] consumes most energy since multi-hop forwarding to the base station requires more hops of traffic relays. Although introducing a dedicated data gathering vehicle increases the moving cost, the total system cost is still 30% less than the static scheme. This is because we have smaller clusters and thus less energy for traffic relay on intermediate nodes. The combined scheme seems to have the least system cost. However, it has prohibitive network latency as illustrated in Fig. 3.13(b). Further, since the data collection hop count $k = 3$, it is possible that some nodes are not covered in simulation time. So their packets have to be buffered until the

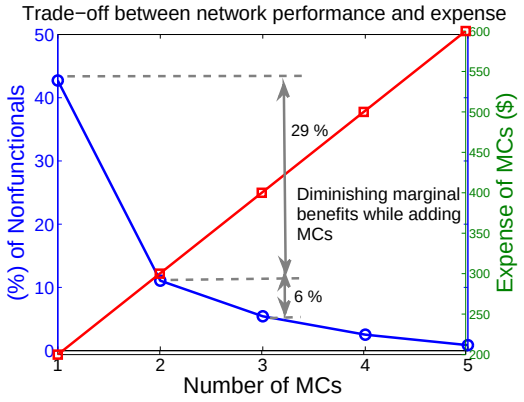


Figure 3.15: Trade-off between network performance and expense.

MC moves into multi-hop communication range. It lowers the energy consumption at the cost of dramatically exacerbating packet latency.

Trade-off between Network Performance and Expense

Finally, we evaluate the trade-off between network performance and monetary costs of the MCs. We assume one MC may not cost too much (e.g., $100) when manufactured at large scale. Fig. 3.15 shows the average percentage of nonfunctional nodes versus the total costs of MCs. Initially there are one MC and one data collecting vehicles, resulting in nearly 42% nonfunctional nodes. Adding one more MC reduces this number to 12%. As we keep adding, the marginal benefits decrease whereas the expense grows linearly. This shows that when the number of nonfunctional nodes is already very small (e.g. below 10%), adding more MCs may not be cost-effective. Therefore, considering such trade-offs, a good strategy is to select a minimum number of MCs that can maintain very low levels (e.g. around 5%) of nonfunctional nodes.

3.7 Conclusions

In this chapter, we consider several important factors overlooked in the previous chapter, including the MC's energy consumption, capacity limits, energy efficiency and data latency. We first propose a low latency mobile data gathering scheme that can collect packets from all nodes and provide theoretical results on latency.

Then we establish a mathematical model to calculate the minimum number of MCs needed, nodes' lifetimes and adaptive recharge thresholds. We formulate recharge optimization problem into a Profitable Traveling Salesmen Problem with Capacity and Battery Deadline constraints, which is NP-hard. We propose two low complexity algorithms. The greedy algorithm maximizes the recharge profit in each step. A three-step adaptive algorithm systematically captures the recharge capacity and nodes' battery deadline constraints while minimizing traveling costs. We evaluate and compare the proposed algorithms by extensive simulations. They show that the adaptive algorithm can provide better stability by reducing the number of nonfunctional nodes and their durations. We also validate the theoretical results through simulations. The comparison with other schemes show that the adaptive algorithm achieves both low latency and high energy efficiency.

Chapter 4

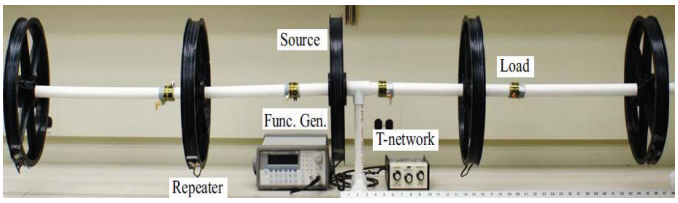
Multi-hop Wireless Charging via Resonant Repeaters

4.1 Introduction

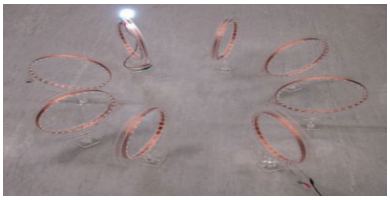
In the previous chapters, the MC can only recharge one sensor node at a time due to physical limitations from wireless charging range. This is because the charging efficiency decays as an inverse cube of distance - hence, most of the previous works only considered "short-range" charging where an MC needs to approach nodes in very close proximity and can only recharge the nodes one by one. This may lead to extremely long recharging latency: if a rechargeable battery takes 1-4 hours to fully recharge, a network of hundreds of nodes can take days or weeks. During such long latencies some nodes may exhaust energy and cause service interruption.

Inspired by the latest advances in mid-range wireless charging (where mid-range refers to energy transmitting distances much larger than the diameter of coils) that can relay energy over several hops to simultaneously replenish multiple nodes, in this paper, we explore how to leverage this technology to solve the above problems and enhance network scalability and performance.

One of the most cost-effective means to relay energy is to use *resonant repeaters*. Resonant repeaters can be easily manufactured from copper coils at low costs. In [80], significant improvements (10%-46%) in efficiency are reported by adding resonant repeaters between the source and receiving coils. In [81], distributing 15mW energy over a distance of 2m to 6 different loads through 4 resonant repeaters has



(a) Distribute 15mW energy to 6 loads by 4 repeaters over 2m. Repeater coils are twisted on the black wheels with loads separated in between (courtesy of [81]).



(b) Power a 14W lamp by organizing repeaters into domino form (courtesy of [78]).

Figure 4.1: Experimental prototypes of multi-hop wireless charging using resonant repeaters[78, 81].

been demonstrated (Fig. 4.1(a)). In [78], experiments have shown that resonant repeaters can be organized into a domino form to power a 14W lamp (Fig. 4.1(b)). Their theoretical results indicate up to 50-70% charging efficiency even after 5-6 hops of relays.

For WRSNs, only very few works have considered recharging nodes in multi-hops [82, 84]. Although pioneering first steps, these works do not consider the physics laws governing wireless charging efficiency. In practice, the efficiency is not only impacted by the distance and MC's position, but also by a series of phenomenons such as cross-coupling where complex interactions between neighboring resonant repeaters cannot be simply ignored. Further, unlike data flows whose rates can be continuously adjusted, an energy flow can be turned on/off but there is no easy means to alter its rate over links [78]. Thus these works would deviate from real network operating conditions.

To tackle these limitations, we propose a new multi-hop wireless charging framework to improve charging capability and scalability. With a low-cost repeating circuit installed, sensor nodes can relay energy to their neighbors. Since previous single-node recharge scheduling algorithms do not consider such energy relaying,

we provide a new recharge scheduling algorithm for this fundamentally different charging model. The new framework raises several interesting questions. First, how to quantify the improvements from charging capability compared to the single-node recharge in terms of the number of nodes a MC can cover, and the number of MCs needed? Second, given time-varying recharge requests, where MCs should stop to recharge surrounding nodes such that multi-hop wireless charging cost is minimized and how to schedule the MCs to minimize the moving cost? Third, are there any relationships between the two types of costs and is there a way to minimize the total system cost? Finally, what is the tradeoff among energy efficiency, network scalability and packet latency compared to the single-node recharge scheme?

To answer these questions, we first show how to accurately calculate wireless charging efficiency based on well-established methods in physics and electronics [78], so as to estimate energy charging cost during multi-hop relay. Then we theoretically analyze the energy consumptions under the hybrid data gathering model and estimate the improvements of using multi-hop charging. Based on the mathematical model, we can derive the number of MCs needed to cover a network. Further, to minimize both charging and moving costs, we formulate recharge scheduling into a problem in the category of location-routing problems [85] with two objectives. Since the problem is NP-hard, we propose a two-step approximation algorithm that guarantees all energy demands are satisfied while minimizing the costs. In the first step, we identify a set of representative sensor locations (called "anchors") where MCs stop and recharge nearby nodes such that overall charging cost is minimized. Our algorithm achieves a bounded approximation ratio of $\log n$ to the optimal solution (where n is number of nodes). In the second step, we first utilize an approximation algorithm for the Traveling Salesmen Problem to compute a complete shortest recharge tour through anchors. Then we assign recharge routes for different MCs by dividing the complete tour according to MCs' recharge capacity, energy demands and multi-hop charging cost. Given the selection of anchors, our algorithm generates recharge tours with the moving cost on MCs bounded by $(\frac{5}{2} - \frac{1}{2k})$ ratio to the optimal result (where k is number of tours). Finally, upon discovering more room exists to optimize the system cost (charging cost plus moving cost), we propose a post-optimization algorithm that iteratively changes nodes with low charging efficiency into anchors and inserts them back into the established routes to further reduce the overall system cost.

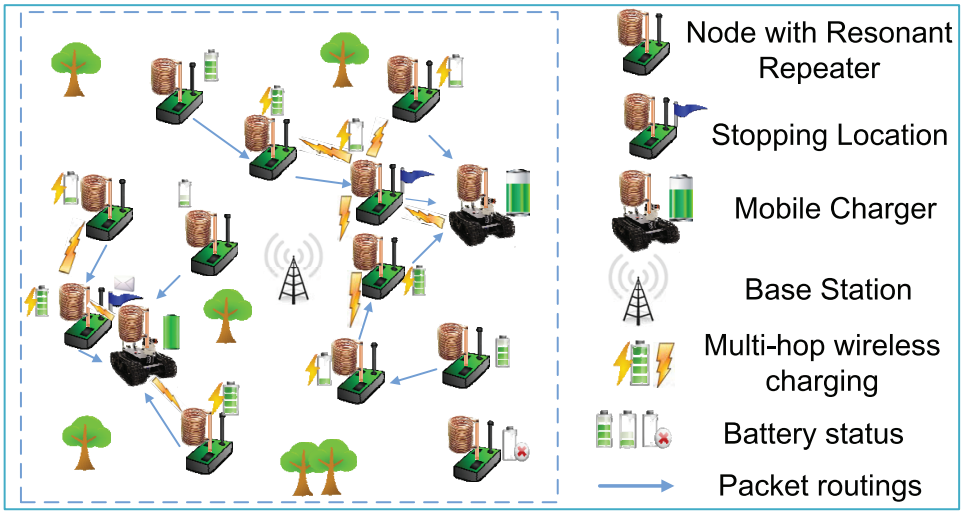


Figure 4.2: Multi-hop wireless charging based on resonant repeaters.

The rest of this chapter is organized as follows. Section 4.2 outlines the network model, and briefly describes how to compute charging efficiencies. Section 4.3 formalizes multi-hop recharge scheduling into a bi-objective optimization problem and proposes a two-step approximation algorithm with a post-optimization algorithm given in Section 4.4. Section 4.5 provides simulation results. Finally, Section 4.6 gives some discussions and Section 4.7 concludes this chapter.

4.2 Preliminaries

In this section, we introduce the network model and briefly describe the procedures to calculate multi-hop charging efficiency while taking comprehensive factors such as mutual inductance and cross-coupling into consideration.

4.2.1 Network Model

Network Components

Fig. 4.2 shows the basic components in our framework. We assume N static sensor nodes are uniformly randomly distributed in a circular field. An embedded resonant repeater is added into the charging circuitry on both MCs and sensors. It can be manufactured at low costs using copper wires/coils.

In contrast to previous works in [82, 84], which do not provide any model of energy relay or charging efficiency, our framework establishes on physical models and provides concrete details by considering mutual inductance and cross-coupling effects between neighboring sensor coils. For simplicity, we assume all the nodes and MCs have identical coils with n_t rounds and r_s radius. To successfully relay energy, nodes need to tune their resonant frequencies to the same frequency as MCs and these nodes form a *charging set* around the stopping location of an MC. In practice, this is done by having different resonant frequency bands for neighboring charging sets. The band between different frequencies is wide enough to avoid any interference.

To provide an effective charge that can stimulate enough currents on sensors' reception circuits, the charging efficiency η should be greater than a threshold τ, e.g., $\tau = 30\%$; otherwise, the node cannot be properly charged and it stops relaying wireless energy. We assume a charge controller is built into the circuit. It regulates the charging current to be a constant and stable value in order to protect the battery and elongate its lifespan.

If a node's battery level falls below threshold β, a recharge request is triggered and sent to MCs. m MCs respond to recharge requests cooperatively. They stop at selected sensor locations (called *anchors*) to recharge nodes that have also requested for recharge with multi-hop energy relay.

To maintain perpetual operation of the network, the MCs need to make every effort to recharge nodes before their battery energy depletes. For a recharge schedule, we denote the time instance when the MC begins to recharge sensor i (via multi-hops) by A_i. Then for the node with lifetime L_i, the MC should arrive before the battery depletes, $A_i \leq L_i$. $L_i = E_i/\mu_i$ where E_i is the residual battery energy and μ_i is the average traffic rates including the traffic relayed by i. In practice, it is common that the energy requests come in the form of bursts and the MCs cannot handle all the requests at once. Some nodes that cannot be recharged on time will deplete energy and become nonfunctional temporarily. To this end, we introduce a term of *recharge delay*, q_i, to measure how long an MC misses the battery deadline of a node (late arrival). q_i takes the maximum value of $A_i - L_i$ and 0. That is, if $A_i > L_i$, a late arrival occurs and $q_i = A_i - L_i$; otherwise, $q_i = 0$. The recharge delay is also a measure of the time duration while a node is in nonfunctional status.

In addition, we make the following assumptions: 1) We assume the network

is connected so messages can be exchanged among nodes. 2) Because nodes are static, network topology can be obtained at the initialization stage by a one-time effort. 3) To increase life cycles of batteries, only nodes in the charging range with energy below a threshold β will be recharged. Otherwise, they serve as energy relays for other nodes by switching on the resonant repeating circuit. 4) When the MC is about to deplete its battery, it goes back to the base station for a quick battery replacement.

4.2.2 Multi-hop Wireless Charging Efficiency

Calculating multi-hop wireless charging efficiency is the key in our framework. In this subsection, we describe an approach to estimate efficiency η_n after n relays. In principle, efficiency is governed by *mutual inductance*. Let L_{ij} denote the mutual inductance between repeaters on nodes i and j. From [79] we have

$$L_{ij} = \kappa_{ij}(n_t L_s)^2 \approx \frac{r_s^3}{2d_{ij}^3}(n_t L_s)^2 \tag{4.1}$$

where r_s is the coil radius, n_t is the number of rounds of coil wires, κ_{ij} is the magnetic *coupling coefficient* between nodes i and j ($0 \leq \kappa_{ij} \leq 1$), and L_s is the self-inductance of coils. $L_s = \mu_0 r_s(\ln \frac{8r_s}{r_d} - 2)$, r_d is the wire radius and μ_0 is the permeability constant equal to $4\pi \times 10^{-7} H \cdot \text{m}^{-1}$ (Henry per meter). The approximation is taken when wireless charging distance d_{ij} between i and j is much larger than the dimensions of coil radius r_s. Based on Kirchoff's Voltage Law, an established method in [78] can be used to calculate charging efficiency. The input voltage from MC's transmitting coil induces currents I_2-I_n on all sensor coils oscillating at frequency w and these values can be obtained by solving n linear equations as shown below (where $X = wL_s - \frac{1}{wC}$ and C is the capacitance).

$$\begin{pmatrix} R+jX & \cdots & jwL_{1n} \\ jwL_{12} & \cdots & jwL_{2n} \\ \vdots & \ddots & \vdots \\ jwL_{1(n-1)} & \cdots & jwL_{(n-1)n} \\ jwL_{1n} & \cdots & R+jX \end{pmatrix} \begin{pmatrix} I_1 \\ I_2 \\ \vdots \\ I_{n-1} \\ I_n \end{pmatrix} = \begin{pmatrix} V_{sc} \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

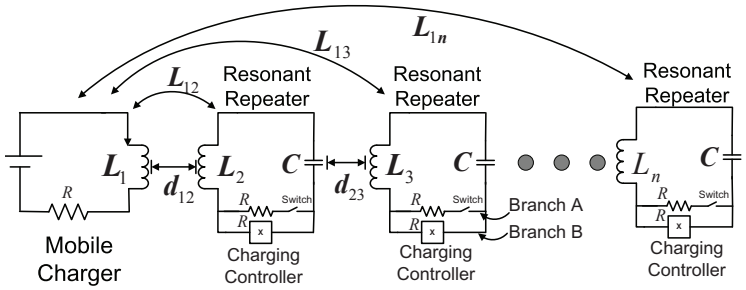


Figure 4.3: A schematic of multi-hop wireless charging circuitries.

The above computation ensures that mutual inductance and cross coupling effects are accounted in our model. To maximize the utility of resonant repeaters, nodes finish charging their batteries should still act as energy relays. Denote the resistance of the charging circuit of branch B by R in Fig. 4.3. A resistor of the same R is added to match its resistance with the charging circuit. When a node is charging, switch at branch A is open and the output load is R. Once charging is finished, the battery stops charging and makes branch B open. Then we close the switch at branch A so the output load is still R. In this way, the charging efficiencies can stay the same despite some nodes might finish recharging earlier.

Although there are some energy cost in the repeating circuity, it can be justified from the following two aspects. First, since battery has a low internal resistance[46], the energy dissipated on the resistor for the given charging current is very small. Our model will successfully capture this factor into the calculations of charging efficiency next. Second, since nodes within MC's charging range share similar amounts of traffic load, the standard deviation of recharge times is small. Nodes within the same charging set usually finish charging at around the same time, thereby reducing the energy costs during these time gaps.

While relaying energy, power consumption of the resonant repeater is the energy dissipated at the resistor R (i.e., $I_k^2 R$, where I_k is the constant current at branch of the k-th relay). The efficiency at the n-th repeater output is

$$\eta_n = (RI_n^2)/(R\sum_{k=1}^{n} I_k^2) = I_n^2/\sum_{k=1}^{n} I_k^2. \tag{4.2}$$

As an example, in Table 4.1, we calculate the charging efficiency for up to

Table 4.1: Charging efficiency vs. relay hops

hops	1	2	3	4
$d = 0.25$ m	0.93	0.88	0.85	0.81
$d = 0.5$ m	0.89	0.68	0.54	0.39
$d = 0.75$ m	0.82	0.48	0.43	0.11
$d = 1$ m	0.78	0.33	0.27	0.03
$d = 1.25$ m	0.53	0.21	0.11	0.01
$d = 1.5$ m	0.35	0.08	0	0

4 hops with $n_t = 300$ rounds and $r_s = 10$ cm coil radius while changing the hop-to-hop distance d from 0.25 m to 1.5 m. First, we can see wireless charging efficiency decreases with more hops. This matches the intuition that energy relay attenuates rapidly from the source. Second, we observe that the efficiency decreases sharply when d is larger. This is because that the mutual inductance declines as an inverse cube of distance. For instance, when $d = 0.25$ m, charging efficiency after 4 hop relay (η_4) is still 81%. When $d = 1.5$ m, η_2 has reduced to 8% and hardly provides any effective charge for sensor's battery. Thus, the efficiency depends on the number of intermediate nodes that are relaying energy as well as the distance between them. Based on this method, each node can calculate energy cost during multi-hop charging by acting as a source where the MC might be residing at. Since the charging range is usually much less than the communication range, nodes can propagate requesting packets to know the positions of their neighbors and use this information to calculate charging efficiency.

4.3 Scheduling MCs for Multi-hop Charging

In this section, we discuss how to schedule m MCs for multi-hop wireless charging to respond to sensors' energy requests. A variety of practical factors, e.g., location-dependent charging efficiencies, energy charging cost, MC's recharge capacity, and energy consumption in movements, are brought into our problem formulation.

Our objectives are two-folds: on one hand, we aim to minimize the energy cost via multi-hop charging. It requires MCs to select advantageous locations (anchors) for stopping so that overall charging efficiency is maximized. On the other hand, we want to minimize moving energy consumption for MCs within their recharge

capacities. In principle, our problem resembles the location-routing problem (LRP) [85]. LRP finds the optimal warehouse locations for minimum accessing and distributing costs of traversal routes over demand locations that start and end at warehouses. It encompasses two NP-hard problems, i.e., location and routing problems, and seeks to provide an integrated solution to optimize the overall system cost. However, instead of MCs directly visiting each warehouse location in the original LRP, our problem involves an additional level of cover problem. That is, the anchors have to ensure that all sensors are "covered", i.e., be charged either directly or via multi-hops. Based on the energy requests at different times, MCs need to calculate anchors and fulfill all requests from sensors adaptively.

Thus we formulate our problem in the context of LRP with two objectives that minimize both MCs' charging cost and moving cost. Due to the NP-hardness nature of our problem, we propose a two-step approximation algorithm. In the first step, a ratio of $\log n$ to the optimal charging cost is achieved, where n is the total number of recharge requests. In the second step, given the selection of anchors, the maximum touring cost is bounded by a ratio of $(\frac{5}{2} - \frac{1}{2k})$ to the optimal solution, where k is the number of scheduled tours (normally, $k = m$). Finally, based on the results from the algorithm, we study the relationships between the two objectives and combine them into a single-objective problem using the weighted method [90]. A post-optimization algorithm is proposed to further reduce the total system cost by inserting anchors into the established routes.

4.3.1 Problem Formulation

Given the set of MCs, $\mathcal{M}$, the set of sensor nodes requesting recharge, $\mathcal{N}$, the set of potential anchors where MCs can stop, $\mathcal{A}(\mathcal{A} \subseteq \mathcal{N})$, and the set of starting locations of MCs, $\mathcal{I}$, we formulate the problem as follows.

Consider a graph $G = (V, E)$, where V_i $(i \in \mathcal{N} \bigcup \mathcal{I})$ is the location of sensor node i, and E are edges connecting sensor nodes. The weight of an edge E_{ij} is the energy cost c_{ij} traveling on the edge, which is proportional to the distance between nodes i and j. Each MC has recharge capacity C_h corresponding to the maximum number of nodes and distance it can travel in each tour. A node i has energy demand d_i (which equals full capacity minus its residual energy). Each anchor a covers a set of nodes $\mathcal{S}_a$ and the entire covered set of all the anchors achieves $\mathcal{N}$ $(\bigcup \mathcal{S}_{a \in \mathcal{A}} = \mathcal{N})$.

Recharging $\mathcal{S}_a$ requires t_a time which is usually determined by the node with the longest recharge time. For a node i, η_{ia} denotes the recharge efficiency when an MC resides at anchor a. Several decision variables are introduced in the formulation. x_{ijk} is 1 if anchor $i \in \mathcal{A}$ immediately precedes $j \in \mathcal{A}$ for MC k; otherwise, it is 0. For $i \in \mathcal{N}, k \in \mathcal{M}, a \in A$, y_{ia} is 1 if node i can be recharged when an MC resides at $a \in A$. z_{ik} is 1 if node i is recharged by MC k. u_a is 1 if an anchor a is chosen; otherwise, it is 0. v_{ik} is the position of anchor i in the path of MC k. Our objective is to minimize the charging cost in multi-hop energy relays, F_c, and MCs' moving cost, F_m.

$$\mathbf{P1}: \quad \min F = (F_c, F_m) \tag{4.3}$$

where,

$$F_c = \sum_{i \in \mathcal{N}} \sum_{a \in \mathcal{A}} \frac{1 - \eta_{ia}}{\eta_{ia}} d_i y_{ia} \tag{4.4}$$

$$F_m = \sum_{i \in \mathcal{A}} \sum_{j \in \mathcal{A}} \sum_{k \in \mathcal{M}} c_{ij} x_{ijk} + \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{A}} \sum_{k \in \mathcal{M}} c_{ij} x_{ijk} \tag{4.5}$$

Subject to

$$\sum_{i \in \mathcal{A}} x_{ijk} = z_{jk}, j \in \mathcal{A}, k \in \mathcal{M} \tag{4.6}$$

$$\sum_{j \in \mathcal{A}} x_{ijk} = z_{ik}, i \in \mathcal{A}, k \in \mathcal{M} \tag{4.7}$$

$$\sum_{a \in \mathcal{A}} y_{ia} = 1, i \in \mathcal{N} \tag{4.8}$$

$$\eta_{ia} y_{ia} > \tau, i \in \mathcal{N}, a \in \mathcal{A} \tag{4.9}$$

$$y_{ia} \leq u_a, i \in \mathcal{N}, a \in \mathcal{A} \tag{4.10}$$

$$\sum_{i \in \mathcal{N}} z_{ik} (\sum_{a \in \mathcal{A}} d_i y_{ia} / \eta_{ia}) + \sum_{i \in \mathcal{A}} \sum_{j \in \mathcal{A}} c_{ij} x_{ijk} + \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{A}} c_{ij} x_{ijk} \leq C_h, k \in \mathcal{M} \tag{4.11}$$

$$\sum_{k \in \mathcal{M}} z_{ak} = u_a, a \in \mathcal{A} \tag{4.12}$$

$$2 \leq v_{ik} \leq |\mathcal{N}|, i \in \mathcal{A}, k \in \mathcal{M} \tag{4.13}$$

$$v_{ik} - v_{jk} + (|\mathcal{A}| - |\mathcal{M}|) x_{ijk} \leq |\mathcal{A}| - |\mathcal{M}| - 1$$
$$i, j \in \mathcal{A}, k \in \mathcal{M} \tag{4.14}$$

$$x_{ijk}, y_{ia}, z_{ik} \in \{0, 1\}, i, j \in \mathcal{N}, a \in \mathcal{A}, k \in \mathcal{M} \tag{4.15}$$

In the above formulation, constraint (4.6) and constraint (4.7) stipulate the connectivity of the path that an MC stopping at an anchor also leaves it. Constraint (4.8) imposes that all the nodes request recharge are covered by anchors. Constraint (4.9) ensures that the recharge efficiency for a node from its anchor should be larger than the efficiency threshold. Constraint (4.10) guarantees that a node is assigned to one of the anchors. Constraint (4.11) mandates that the sum of total demands serviced by an MC plus its moving energy consumptions should not exceed its recharge capacity. Constraint (4.12) enforces that each anchor is visited by only one MC. Constraints (4.13) and (4.14) are formed according to [57] to prevent subtours of MCs. Constraint (4.15) forces x_{ijk}, y_{ia} and z_{ik} to be 0-1 valued.

Remarks: This formulation reflects recharge schedules at time t based on N energy requests (N is an input). For executions at different times, the optimization problem takes corresponding inputs and generates different results (anchors,

MC schedules, etc). Although we do not formulate node lifetime strictly into the formulation, it will be considered by our algorithm in Section 4.3.2 and Section 4.4.3.

The above problem is NP-hard because the location routing problem is known to be NP-hard[85]. Although standard optimization procedures can yield optimal solutions [85], it is prohibitive to run them on MCs due to enormous computation overhead. The base station has computational resources. However, the communication overhead to maintain updated energy requests and disseminate recharge decisions for MCs could be high in a long run. Moreover, the existing optimization methods are usually designed to handle static inputs and lack the flexibility to deal with constant variations in sensor networks such as battery energy and MC movements. Therefore, a polynomial-time approximation algorithm with an acceptable bounded ratio is more desirable in practice. To design the approximation algorithm, we follow a natural approach to tackling the objectives sequentially and finally examine the relationships between them. Next, we propose a two-step approximation algorithm which first selects the anchors that minimize energy charging cost, and then finds the minimum recharge routes for MCs.

4.3.2 Approximation Algorithms

In this subsection, we explain the details of the algorithm. We first define a *charging set* $\mathcal{S}_i$ of node i as its nearby nodes with charging efficiencies larger than τ when an MC stops at node i. At the network initialization phase, each node performs the procedures in Section 4.2.2 to compute its charging set in a distributed manner. For node i, its neighbor j is included in $\mathcal{S}_i$ only if j's charging efficiency is larger than threshold τ and the corresponding efficiency is denoted as $\eta_{j,i}$ ($j \in \mathcal{S}_i$). The algorithm starts with finding the set of anchors based on the energy requests.

Adaptive Anchor Selection

We define the weight of each set $\mathcal{S}_i$ as the total energy needed to satisfy the recharge demands of these nodes, $w_i = \sum_{j \in \mathcal{S}_i}(1 - \eta_{j,i})d_j/\eta_{j,i}$. It is not difficult to observe that our objectives in Eq. (4.4) is equivalent to minimizing the sum of weights of the selected sets. In general, this problem belongs to the category of *Set Cover Problem* (SCP) with one difference: While the original SCP allows the results to share

Table 4.2: Adaptive Anchor Selection Algorithm

Input: Recharged node set $\mathcal{N}$, charging set $\mathcal{S}_i$, energy demand d_i, charging efficiency of node j $\eta_{j,i}$ when an MC stops at i, $i \in \mathcal{N}$. Empty sets $\mathcal{A}$, $\mathcal{B}$.
Output: Set of anchors $\mathcal{A}$ and resultant subsets $\mathcal{B}$
While $\mathcal{B} \neq \mathcal{N}$
Calculate $\overline{w_i} = \sum_{j \in \mathcal{S}_i} \frac{(1-\eta_{j,i})d_j}{\eta_{j,i}} / |\mathcal{S}_i|$.
Find minimum weight $k = \arg\min_i \overline{w_i}$, $i \in \mathcal{N}$.
$\mathcal{A} \leftarrow \mathcal{A} \bigcup k$, $\mathcal{B} \leftarrow \mathcal{B} \bigcup \mathcal{S}_k$.
$\mathcal{S}_i \leftarrow \mathcal{S}_i - B, \forall i \backslash k \in \mathcal{N}$.
End While

the same nodes and thus resultant sets are not necessarily disjoint, our formulation restricts a node to be recharged by only one MC (Eq. (4.12)), since if a node can be recharged by more than one MCs in different recharge routes, it is always preferred to assign the node to a charging set with higher charging efficiency. Hence in our problem, the resultant sets should be disjoint. Next, we modify the classic greedy approach to fit into the context of our problem.

Initially, we define sets $\mathcal{A}$ and $\mathcal{B}$ to record anchors and their covered node sets respectively and both sets are initialized to empty. First, for each node $i \in \mathcal{N}$, we compute its average weight, $\overline{w_i} = \sum_{j \in \mathcal{S}_i} \frac{(1-\eta_{j,i})d_j}{\eta_{j,i}} / |\mathcal{S}_i|$ and search for the set with the minimum $\overline{w_i}$. Assume node k's subset has the least average weight so k becomes an anchor. Then, it is added into $\mathcal{A}$ and $\mathcal{S}_k$ is put into $\mathcal{B}$ to be marked as "covered." In practice, this is done by tuning all the nodes in $\mathcal{S}_k$ to have the same resonant frequency (described in the next subsection). Since those nodes might be also covered by other sets, we need to remove them from the remaining sets. Their elements are updated accordingly, $\mathcal{S}_i = \mathcal{S}_i - B, \forall i \backslash k \in \mathcal{N}$. At this time, if $\mathcal{B}$ contains all the nodes in $\mathcal{N}$, the algorithm terminates. Otherwise, it continues to find the next set among the remaining nodes with minimum average weight until all the nodes are covered ($\mathcal{B} = \mathcal{N}$). Algorithm 4.2 shows the pseudo-code for the adaptive anchor selection algorithm.

Resonant Frequency Assignment

After the anchors have been found in the first step, we need to assign resonant frequencies in order to distinguish charging sets and avoid potential interference. By tuning to a proper frequency, nodes can "join" or "leave" a set very easily. Given an available resonating frequency range, we divide it into numerous frequency bands and each band should be reused as long as there is no interference between the neighboring charging sets, i.e., the frequency assignment for each charging set and its neighbors are different. This problem is equivalent to the classic vertex coloring problem [56] which tries to color nodes in a graph with as small number of colors as possible such that no two adjacent nodes have the same color. Here, the vertice are anchors and edges are connections represented by energy relays between anchors if the distance between any two elements in their charging sets is less than the maximum charging range $r_{\max}$. Unfortunately, vertex coloring is a well-known NP-hard problem and it even turns out that approximation within $n^{1-\epsilon}$ is NP-hard ($0 < \epsilon < 1$, $n = |\mathcal{A}|$)[86]. For a reasonable balance between computation complexity and optimality, we propose an algorithm that uses at most $\max_{1 \le i \le |\mathcal{A}|} (\Delta_i + 1)$ frequency bands, where Δ_i is the degree of anchor i. A set of frequency bands is denoted by $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$.

After the anchors are determined in Section 4.3.2, the algorithm starts from an arbitrary anchor in $\mathcal{A}$ and uses f_1 as its resonant frequency. Then it proceeds to the next anchor and uses available frequency band with the lowest f_i if it is not used by any of its neighboring anchors. The algorithm terminates when all the anchors in $\mathcal{A}$ are assigned proper frequency bands. At this point, the anchors, charging sets and their resonant frequencies are determined and these decisions are disseminated to the anchors. Anchors also send out packets carrying their corresponding frequency information within the boundary of their charging sets. Since the maximum charging range $r_{\max}$ is usually less than transmission distance d_r, the construction of charging sets is done easily by one-hop transmission. Thus, the message overhead is $O(|\mathcal{N} - \mathcal{A}|)$. The algorithm is summarized in Algorithm 4.3.

Note that the upper bound of $\max_{1 \le i \le |\mathcal{A}|} (\Delta_i + 1)$ holds because an anchor i has at most Δ_i neighboring anchors and occupies at most Δ_i frequency bands (some of the neighboring anchors may have already been assigned frequencies). By the same token, for the anchor with the maximum degree, at most the same amount

Table 4.3: Resonant Frequency Assignment Algorithm

Input: Set of anchors $\mathcal{A}$, set of frequency bands $\mathcal{F}$.
Output: Frequency assignment $f_a, \forall a \in \mathcal{A}$.
Establish connections among anchors based on $r_{\max}$.
While $\mathcal{A} \neq \emptyset$
 Check the frequency of anchor a's neighbors, denoted by $\mathcal{F}'$.
 Find available frequency bands, $\mathcal{F} \leftarrow \mathcal{F} - \mathcal{F}'$.
 Assign frequency $\min(f_k)$ $k \in \mathcal{F}$ to anchor a.
 Set frequency of nodes in charging set $\mathcal{S}_a$ to f_k.
 $\mathcal{A} \leftarrow \mathcal{A} - a$.
End While

of frequency bands are needed for its neighbors. Thus, it is not difficult to see the upper bound holds at the maximum degree of anchors.

Schedule Recharge Routes

After the set of anchors $\mathcal{A}$ has been found, we assign the recharge routes for m MCs while considering MCs' capacities along with their moving cost and multi-hop charging cost. Based on [87], we propose an approximation algorithm to bound MCs' moving energy cost given the anchors. Our approach first utilizes a Traveling Salesman Problem (TSP) algorithm to compute a complete route on $\mathcal{A}$, e.g., 1.5-approximation Christofides algorithm [88]. In this way, we can ensure that anchors close to each other are placed on the same MC's recharge route. To facilitate our analysis, we assume that the complete tour starts at the base station and ends at the last node for recharge. In fact, the starting positions of MCs are the ending positions from the last tour and MCs traverse through the base station to upload data packets. The recharge sequence can be expressed as $r = (b, 1, 2, \ldots, i, \ldots, n)$, where anchor $i \in \mathcal{A}, n = |\mathcal{A}|$ and b is the base station. To reflect MC's starting position, an extra edge with cost $c_{i,b}, i \in \mathcal{I}$, can be added to represent the energy cost from MC's starting location $i \in \mathcal{I}$ to the base station b. Let $c_{\max}$ denote the maximum energy cost from any node on the path to the base station, $c_{\max} = \max_{i \in \mathcal{A} \bigcup \mathcal{I}} c_{b,i}$. The TSP algorithm yields a complete route r that incurs c_r energy cost using one MC.

Next, r is split into k tours. For partitioning, we start with an arbitrary direction along r. For each route $j, 1 \leq j \leq k$, we find the last anchor along the complete tour

r that ensures the traveling energy cost is no greater than $\frac{j}{k}(c_r - c_{\max}) + 2c_{\max}$. Here, the term $2c_{\max}$ is the maximum energy cost from MC's starting position to the base station plus the cost from the base station to the first anchor on the recharge path. Then r will be split into k tours. Let a_i^j and a_l^j represent the i-th and the last nodes in the j-th tour, respectively. The j-th tour is then obtained as $(\mathcal{I}_j, b, a_1^j, a_2^j, \ldots, a_l^j)$.

k depends on MCs' recharge capacity (constraint in Eq. (4.11)). We check whether an equal division of m MCs from the total energy costis less than MC's capacity. Depending on the results, there are two cases:

Case 1: If an equal division of m among the total cost is less than MC's capacity C_h, $k = m$. In this case, m MCs are sufficient to cover all the nodes in one shot.

Case 2: Otherwise, $k > m$ and,

$$k = \lceil (\sum_{i \in \mathcal{A}} \sum_{j \in \mathcal{S}_i} \frac{(1 - \eta_{j,i}) d_j}{\eta_{j,i}} + c_r - c_{\max}) / (C_h - 2c_{\max}) \rceil. \tag{4.16}$$

This case usually occurs when the temporary energy demands overwhelmingly exceed MCs' recharge capacity so that they have to take $\lceil \frac{k}{m} \rceil$ rounds to cover all the routes. In each round, at most m routes can be selected from k, thus late recharge for some nodes is inevitable. Therefore, our objective is to reduce the recharge delay as much as possible. Let us denote the recharge time for node i by t_i and traveling time between nodes i and $i+1$ by $t_{i,i+1}$ in the recharge sequence. For multi-hop wireless charging, the MC leaves an anchor after it has fulfilled all requests in a charging set, so the recharge time of $\mathcal{S}_a$ is $t_a = \max_{i \in \mathcal{S}_a}(t_i)$. The total time duration for a route j is $T_j = \sum_{a=1}^{l_j} t_a + \sum_{a=1}^{l_j - 1} t_{a,a+1}$. The longest route takes the maximum time among T_j to finish. For route j, if it is selected by an MC in the current round, the recharge delay of all the nodes is

$$P_j = \sum_{i \in \bigcup \mathcal{S}_k^j, k \in \mathcal{A}^j} q_i = \sum_{i \in \bigcup \mathcal{S}_k^j, k \in \mathcal{A}^j} \max(A_i - L_i, 0) \tag{4.17}$$

However, if route j is not selected, in the next round, the worst case occurs when it has to wait for the longest route to finish. The recharge delay is

$$P_j' = \sum_{i \in \bigcup \mathcal{S}_k^j, k \in \mathcal{A}^j} \max(A_i + T_{\max} - L_i, 0) \tag{4.18}$$

An increment

$$\Delta P_j = \sum_{i \in \bigcup \mathcal{S}_k^j, k \in \mathcal{A}^j} \big(\max(A_i + T_{\max} - L_i, 0) - \max(A_i - L_i, 0) \big) \tag{4.19}$$

is observed. To keep recharge delay at minimal, we sort ΔP_j and select the m routes with the largest increment in each round so that those routes that would incur longer delay can be recharged in the current round. The pseudo-code of the algorithm is presented in Algorithm 4.4.

4.3.3 Approximation Bounds and Complexity

We now analyze the approximation bounds for the proposed algorithm. For $n = |\mathcal{N}|$ recharge requests, our algorithm gives a $\log n$ approximation of the energy cost during multi-hop wireless charging and a $(\frac{5}{2} - \frac{1}{2k})$ ratio for the traveling cost given the selected anchors, where k is the number of tours depending on energy demands and recharge capacity C_h. In the extended greedy algorithm of the Set Cover Problem, we assume the optimal energy cost is w^*. During computation, when there are i nodes left to be covered, it incurs at most $\frac{w^*}{i}$ energy cost per node. The bound of the extended greedy algorithm is thus $\sum_{i=1}^{n} \frac{w^*}{i} = w^* \log n$. The equality holds because the summation $\sum_{i=1}^{n} \frac{1}{i} = \log n$ is the n-th harmonic number.

Remarks: Although the $\log n$ bound for energy charging cost seems quite large, it is essentially one of the best polynomial-time approximation algorithms: it has been proved in [89] that the Set Cover Problem cannot be approximated in polynomial time within a ratio of $c \log n$, for $c < \frac{1}{4}$, under general complexity assumptions. A tighter bound might not be necessary given the increased complexity and transient nature of energy requests.

Next, we show that the traveling energy cost has an approximation ratio of $(\frac{5}{2} - \frac{1}{2k})$ respect to k tours. Here, when $k > m$, the $k-m$ tours are traversed by MCs after they have replaced batteries in the base station. Nevertheless, the total cost would still be the same. For the complete tour, the energy cost is c_r with the optimal value c_r^*. Use Christofide's minimum spanning tree approximation to the TSP, $\frac{c_r}{c_r^*} \leq 1.5$ [88]. Assume that tour j has the maximum energy cost c_j among k tours and its optimal value is c_j^*. The energy cost for tour j is at most $\frac{1}{k}(c_r - c_{\max})$ (excluding the

Table 4.4: Route Scheduling Algorithm

Input: Set of anchors $\mathcal{A}$, MCs $\mathcal{M}$, energy demand d_i of node i, charging efficiency of node j, $\eta_{j,i}$ when charger is at i. Set of MCs' initial locations $\mathcal{I}$, capacity C_h, base station b, max energy cost traveling on an edge $c_{\max}$.
Output: Recharge sequence r_j for MC j's tour.
Compute complete TSP recharge path on $\mathcal{A}$ starting from b.
Record the TSP sequence $r = (b, 1, 2, \ldots, i, \ldots, n)$ with cost c_r.
If $(\sum_{i\in\mathcal{A}}\sum_{j\in\mathcal{S}_i}\frac{(1-\eta_{j,i})d_j}{\eta_{j,i}} + c_r - c_{\max})/m + 2c_{\max} < C_h$, $k = m$,
Else $k = \lceil(\sum_{i\in\mathcal{A}}\sum_{j\in\mathcal{S}_i}\frac{(1-\eta_{j,i})d_j}{\eta_{j,i}} + c_r - c_{\max})/(C_h - 2c_{\max})\rceil$.
End If
Start with an arbitrary direction on r, $j = 1$
While r is not exhausted
For tour j, search for the last node a_l^j along r
Satisfying $c_j \le \frac{j}{k}(c_r - c_{\max}) + 2c_{\max}$.
Obtain the j-th tour, $(\mathcal{I}_j, b, a_1^j, a_2^j, \ldots, a_{l_j}^j), 1 \le j \le k$.
Exclude nodes in j-th tour from r. $j \leftarrow j + 1$.
End While
If $k = m$, assign each MC to a recharge route
Else Find the route with max time duration,
$T_{\max} = \max_{j=(1,2,\ldots,k)}(\sum_{a=1}^{l_j} t_a + \sum_{a=1}^{l_j-1} t_{a,a+1})$, calculate recharge delay.
$\Delta P_j = \sum_{i\in \bigcup_{k\in\mathcal{A}^j}\mathcal{S}_k^j} \big(\max(A_i + T_{\max} - L_i, 0) - \max(A_i - L_i, 0)\big)$.
Sort ΔP_j and select the m largest for recharge each round.
End If

edge leaving the base station in the complete tour r) plus $2c_{\max}$ for the two edges connecting the base station to MC's starting position and the first anchor in each tour. Therefore, $c_j \leq \frac{1}{k}(c_r - c_{\max}) + 2c_{\max} = \frac{1}{k}c_r + (2 - \frac{1}{k})c_{\max}$. We divide both sides by c_j^* and have

$$\frac{c_j}{c_j^*} = \frac{1}{k}\frac{c_r}{c_j^*} + \left(2 - \frac{1}{k}\right)\frac{c_{\max}}{c_j^*} \leq \frac{1}{k}k\frac{c_r}{c_r^*} + \left(2 - \frac{1}{k}\right)\frac{1}{2} \leq \frac{5}{2} - \frac{1}{2k} \tag{4.20}$$

The inequality holds because for each tour, an edge is added to connect the first sensor node to the base station, $c_r^* \leq \sum_{i=1}^{k} c_i^*$. If we divide both sides by k and use the fact that $\max_{1\leq i\leq k}(c_i^*) = c_j^*$, we have $\frac{c_r^*}{k} \leq c_j^*$. We take the approximation $c_{\max} \leq \frac{1}{2}c_j^*$. The equality holds when the tour has only one node.

Let us denote the number of energy requests by N and the number of anchors by A. The time complexity of the anchor selection algorithm is $\mathcal{O}(N \log N)$ because if we first sort nodes according to their weights, $\mathcal{O}(N \log N)$ is required. In each step, we select the node with minimum weight and the number of iterations is bounded by N. To assign proper frequencies for anchors, the frequency assignment algorithm needs to go through all A anchors so its time complexity is $\mathcal{O}(A)$. For the route scheduling algorithm, if $k = m$, the time complexity is $\mathcal{O}(A^3 + N)$, i.e., Christofides $\mathcal{O}(A^3)$ algorithm [88] plus splitting demands over N. If $k > m$, the time complexity is $\mathcal{O}(A^3 + N + k + k \log k)$ which consists of a series computations in linear time and sorting operations. When A^3 is much larger than N and k, both cases have time complexity $\mathcal{O}(A^3)$ dominated by the Christofides algorithm.

4.4 Post-optimization by Inserting Anchors

When node's battery deadline is not exceeded, there could be further room to optimize the results of the two-step algorithm. In this subsection, we propose a post-optimization algorithm. Since both objectives in Eq. (4.4) and Eq. (4.5) are the energy outputs from the MC's own battery, we can combine them into a single objective using the weighted method in [90], $F = w_1F_c + w_2F_m$.

The weights w_1 and w_2 are assigned by network administrators to measure the importance of energy charging cost compared to moving cost. If $w_2 > w_1$, it means that the administrator cares more about MC's moving cost over energy charging

cost. For example, if $w_2/w_1 = 2$, for total cost F, reducing the moving cost by 1 J is equivalent to saving energy charging cost of 2 J on MCs. In practice, we would expect $w_2 > w_1$ in most cases as the administrators want to minimize the recharge time by covering more nodes with anchors so a slight increase of energy cost due to multi-hop charging is acceptable.

4.4.1 Inserting Anchors

It is critical to observe that the optimal system cost F achieves a good compromise between F_c and F_m. In fact, any solution that can minimize F is said to be *Pareto optimal* when $w_1, w_2 \neq 0$[91]. In multi-objective optimization, Pareto optimality describes a state that we cannot further increase the profit of one objective without reducing the profit of another objective. For our problem, it means that we cannot further reduce charging cost without increasing the moving cost on MCs. On one hand, introducing more anchors would potentially increase MCs' moving cost F_m; on the other hand, more anchors means fewer energy relays thus less energy charging cost F_c. Based on this observation, we propose a post-optimization algorithm that evaluates whether inserting an anchor into the established *charging sets* leads to lower system cost. However, since such insertion splits the original charging set, it would elongate the total recharge time of the route. To this end, the algorithm should also ensure anchor insertions do not cause battery depletion on subsequent nodes in the route. To keep it simple and effective in a dynamic network environment, we need to avoid computationally intensive algorithms.

The basic procedures is illustrated below. Initially, for each anchor a_i, a node with the maximum charging cost is selected in its charging set $\mathcal{S}_{a_i}$. Then these selected nodes are sorted in a descending order according to their charging costs.

The MC starts from the first node j in the list which has the maximum charging cost on the entire route. Tentatively designate node j as a new anchor because by charging j directly, a great amount of energy cost can be reduced. We denote node j as a new anchor a'_j. Next, an important step is to see whether we can further reduce energy charging cost by moving some of the elements from $\mathcal{S}_{a_i}$ to $\mathcal{S}_{a'_j}$.

This is because a node k in $\mathcal{S}_{a_i}$ may be more efficiently recharged via the new

anchor. For each node k in $\mathcal{S}_{a_i}$, we compare if,

$$(1-\eta_{k,a_i})/\eta_{k,a_i} > (1-\eta_{k,a'_j})/\eta_{k,a'_j} \tag{4.21}$$

If yes, we move node k to be covered in $\mathcal{S}_{a'_j}$ and denote the old a_i by a'_i after this operation. The new anchor will be assigned a new frequency band that is not being used by its neighbors. For k to join the new charging set, its resonant frequency is tuned to be the same as a'_j. All elements in $\mathcal{S}_{a_i}$ are examined to see whether it is beneficial to be included under the new anchor a'_j or remain with old anchor a_i. At this point, a new anchor a'_j is introduced to partition the original charging set whereas their joint coverage still remains the same.

4.4.2 Optimize Total Cost

The next step is to calculate whether there would be a reduction in the total cost F. Denote the changes of moving cost after introducing a'_j by ∂f_m and changes of charging cost by ∂f_c. We assume the new sequence $(a_1, a_2, \ldots, a'_i, a'_j, \ldots, a_{l_s})$ has the lowest moving cost so

$$\partial f_m = (c_{a_{i-1},a'_i} + c_{a'_i,a'_j} + c_{a'_j,a_{i+1}}) - (c_{a_{i-1},a_i} + c_{a_i,a_{i+1}}) \tag{4.22}$$

and

$$\partial f_c = \sum_{a\in\{a'_i,a'_j\}} \sum_{k\in\mathcal{S}_a} \frac{(1-\eta_{k,a})d_k}{\eta_{k,a}} - \sum_{k\in\mathcal{S}_{a_i}} \frac{(1-\eta_{k,a_i})d_k}{\eta_{k,a_i}} \tag{4.23}$$

Then we see whether $\triangle F = w_1\partial f_c + w_2\partial f_m$ is less than zero. If yes, it means a reduction of F is accomplished.

4.4.3 Preserve Battery Deadline

Before the new anchor can be successfully added into the recharge route, the algorithm should check whether the insertion preserves time feasibility of the entire sequence. For the new sequence $(a_1, a_2, \ldots, a'_i, a'_j, \ldots, a_{l_s})$, a node with the minimum value of MC's arrival time minus lifetime is selected for each charging set ($\arg\max_{k\in\mathcal{S}_{a_i}}(A_k - L_k)$ for $A_k - L_k < 0$). The lifetime of this node represents the latest time for an MC to reach its superior anchor and the difference between A_k and

L_k indicates the tightness of the battery deadline. The closer A_k approaches L_k, the less chance a new anchor can be inserted prior to this node without violating the battery deadline. Recall from Section 4.3.2 that the recharge time of a'_j's charging set $\mathcal{S}_{a'_j}$ is governed by the node with the maximum recharge time ($t_{a'_j} = \max_{i \in \mathcal{S}_{a'_j}} t_i$). Thus, the new insertion introduces an additional $\Delta T = t_{a'_j} + \partial f_m / v$ waiting time to all subsequent nodes after a'_j in the sequence. For anchor a_i from a'_j to a_{l_s}, the algorithm computes whether $A_{a_i} + \Delta T - L_{a_i} > 0$. If yes, it indicates the new anchor would potentially cause battery depletion in a_i's charging set and the insertion should be avoided. Otherwise, the new anchor can be successfully added into the recharge route and assigned an appropriate resonant frequency.

To speed up the optimization process, whenever a new anchor insertion causes battery depletion at anchor a_i, a_i is marked, which means new anchors can only be inserted after this location in the sequence. In the subsequent iterations, while a maximum charging cost node is being considered as a candidate anchor, the algorithm first checks its location with the previous mark. If its location is before the mark, the algorithm skips this node and proceeds to the next one. This operation saves a considerable amount of time by avoiding unnecessary computations that would lead to battery depletion ultimately. The algorithm terminates when a new anchor cannot be added into the recharge sequence, i.e., no more improvement on the system cost. The pseudo-code for the post-optimization algorithm is shown in Algorithm 4.5.

4.4.4 Time Complexity

We now analyze the time complexity of the algorithm. Since A anchors are generated from the two-step approximation algorithm, we need to check at most A charging sets. Suppose the size of maximum charging set is S_m. Initially, finding nodes with maximum charging cost for A anchors requires AS_m time and the sorting takes $A \log A$ time. In the worst case, the algorithm iterates through all A anchors and each iteration requires S_m for new anchor re-assignments and AS_m time for checking possible battery deadline violations. In sum, the post-optimization algorithm takes $\mathcal{O}(AS_m + A \log A + A(S_m + S_m A)) = \mathcal{O}(A^2 S_m + A(S_m + \log A))$.

Table 4.5: Post-optimization Algorithm on MC s

Input: Recharge sequence $a_1, a_2, \ldots, a_{l_s}$ for MC s,
Set of anchors $\mathcal{A}_s$, energy demand d_i of node i, charging efficiency of j, $\eta_{j,i}$ if MC is at i, moving cost $c_{i,j}$ on edge (i, j), time feasibility mark at anchor $x \leftarrow 0$ objective weights w_1, w_2, charging set $\mathcal{S}_a$ for all anchors.
Output: A new recharge sequence consists of anchors.
Find nodes with max charging costs for each $\mathcal{S}_a$, $\max(\frac{1-\eta_{i,a}}{\eta_{i,a}} d_i)$, $i \in \mathcal{S}_a$. Sort these nodes in descending order list $\mathcal{I}$. $j \leftarrow 1$
While $x \neq a_{l_s}$ AND $\mathcal{I} \neq \emptyset$
For $j > x$, consider j as a candidate anchor a'_j and $\forall k \in \mathcal{S}_{a_i}$.
If $\frac{(1-\eta_{k,a_i})}{\eta_{k,a_i}} > \frac{(1-\eta_{k,a'_j})}{\eta_{k,a'_j}}$, $\mathcal{S}_{a'_i} \leftarrow \mathcal{S}_{a_i} - k$, $\mathcal{S}_{a'_j} \leftarrow \mathcal{S}_{a'_j} + k$.
$\partial f_m \leftarrow (c_{a_{i-1},a'_i} + c_{a'_i,a'_j} + c_{a'_j,a_{i+1}}) - (c_{a_{i-1},a_i} + c_{a_i,a_{i+1}})$.
$\partial f_c \leftarrow \sum_{a \in \{a'_i, a'_j\}} \sum_{k \in \mathcal{S}_a} \frac{(1-\eta_{k,a})d_k}{\eta_{k,a}} - \sum_{k \in \mathcal{S}_{a_i}} \frac{(1-\eta_{k,a_i})d_k}{\eta_{k,a_i}}$.
$\triangle F \leftarrow w_1 \partial f_c + w_2 \partial f_m$, new sequence $(a_1, \ldots, a'_i, a'_j, \ldots, a_{l_s})$,
If $\triangle F < 0$
For $A_k - L_k < 0$ in each charging set, find
$k = \arg\max_{k \in \mathcal{S}_{a_i}} (A_k - L_k)$, $\Delta T = \max_{i \in \mathcal{S}_{a'_j}} t_i + \partial f_m / v$
For anchor a_i from a'_j to a_{l_s},
If $A_{a_i} + \Delta T - L_{a_i} > 0$.
When $a_i > x$, update mark $x \leftarrow a_i$,
Declare time infeasible, **Break**.
End If
End For
Insertion of a'_j is successful, $\mathcal{I} \leftarrow \mathcal{I} - j$, $j \leftarrow j + 1$.
Else Consider next node j, $\mathcal{I} \leftarrow \mathcal{I} - j$, $j \leftarrow j + 1$.
End If
End While

4.4.5 A Complete Example

To see the entire operation of the algorithm more clearly, we show an example in Fig. 4.4. Fig. 4.4(a) demonstrates a snapshot during the operation of 3 MCs ready to resolve 80 recharge requests of nodes with energy demands from 200-1500 J. The first step is to find anchors that can offer entire coverage of all energy requests with the minimal charging cost. Fig. 4.4(b) shows the results of anchor selection algorithm. 23 anchors are selected and the largest charging set includes 9 nodes. For clarity, we only plot the charging set in Fig. 4.4(b). In Fig. 4.4(c), a complete recharge route is found through all the anchors starting from the base station using the Christofides algorithm [88]. In Fig. 4.4(d), the complete recharge path is split into 3 different routes and each MC is assigned a route. Up to this point, MCs can fulfill all the energy requests by stopping at anchor locations and charge nodes in multi-hops.

To further reduce the system cost, we conduct post-optimization procedures for each MC. For demonstration purposes, we use weights $w_1 = 1, w_2 = 3$ to evaluate the improvement by inserting an anchor and perform an iteration for all 3 MCs. An anchor with maximum charging cost is selected in each route. We calculate the value of $\triangle F$ to see whether there is further saving in the system cost. Our algorithm yields $\triangle F_1 = -496$ J for MC 1, $\triangle F_2 = -490$ J for MC 2 and $\triangle F_3 = 130$ J for MC 3. The insertions would elongate durations of the three recharge routes by 68, 62 and 41 mins, respectively, which still satisfies the minimum battery deadline of the subsequent nodes. Since $\triangle F_1$, $\triangle F_2$ for MCs 1 and 2 are less than zero, inserting anchors at the locations shown in Fig. 4.4(e) has further reductions in system cost. On the other hand, since $\triangle F_3$ for MC 3 is larger than zero, there would be a slight increase of the total cost so we should not insert the anchor at the picked set. For clarity, we have shown two successful cases of anchor insertion in Fig. 4.4(f) for MCs 1 and 2. The post-optimization process ends after each MC has examined all its charging sets for further improvement or a late recharge occurs.

4.5 Performance Evaluations

In this section, we evaluate the performance of multi-hop wireless charging (denoted as "MH"). Since the works in [82, 84] do not provide concrete models of multi-

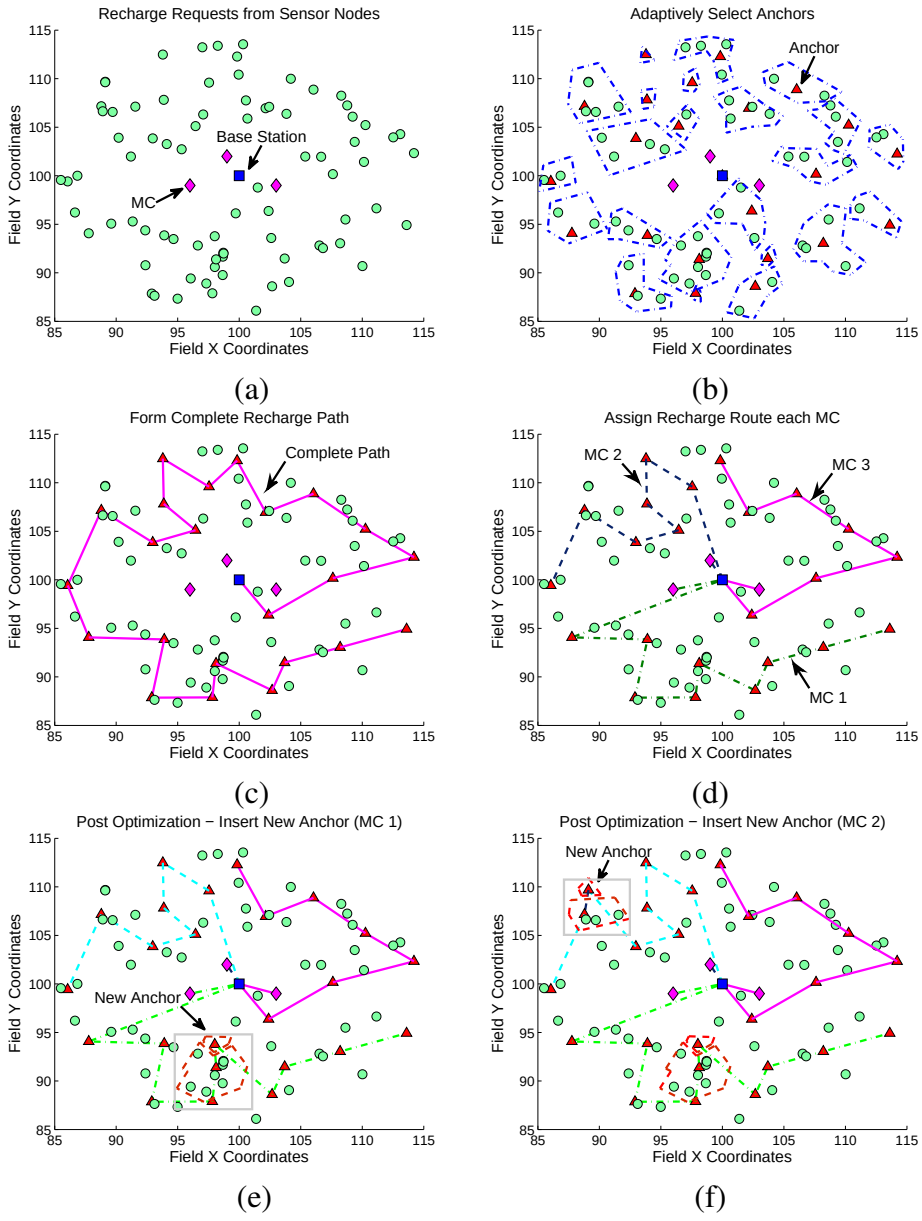


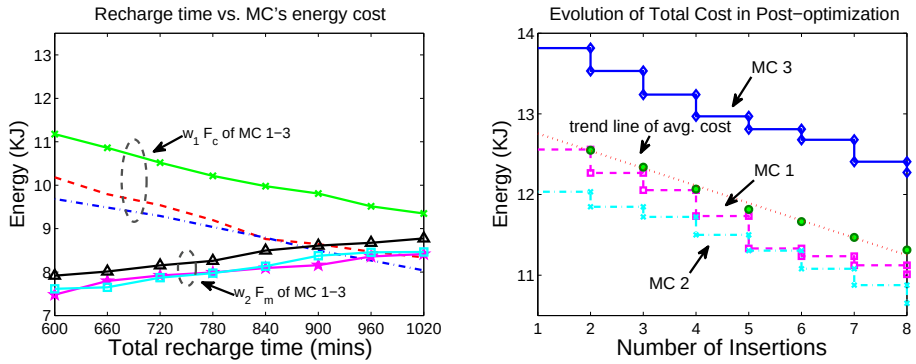
Figure 4.4: A complete example of the algorithm. (a) MCs receive a number of energy requests. (b) Find anchors among nodes. (c) Form a complete recharge path through anchors. (d) Assign recharge route to each MC. (e) Inserting an anchor in MC 1's route. (f) Inserting an anchor in MC 2's route.

hop wireless charging, it is very difficult to compare the performance with theirs. Actually, even the performance and cost of MH over the conventional single node wireless charging (denoted as "SN") is unknown. To this end, we decide to compare our framework with SN in [33, 36, 42, 83]. We distribute 500 sensor nodes uniformly randomly in a circular field with radius of 25 m. The transmission distance and sensing range are 5 m. Sensors' energy consumptions are modeled according to [70]. By using some typical values of $e_0 = 50 \times 10^{-6}$ J/bit, $e_1 = 10 \times 10^{-7}$ J/bit, $\alpha = 4$ and $l_p = 32$ bits, e_c is 21 mJ for transmitting/receiving a packet. We use Dijkstra's shortest path routing algorithm to direct packets to their destinations.

Recharge threshold β is critical to the overall performance. On one hand, if β is large, e.g. 90%, MC's recharge capacity may be easily overwhelmed upon receiving too many energy requests; on the other hand, if β is set to be very small, e.g. 10%, nodes might not have enough residual lifetime before the MC arrives, thereby causing large numbers of energy depletions. Therefore, we set β at 50% of the total battery capacity. We use an AAA NiMH battery of 780 mAh capacity working at 1.5 V. Recharge time is modeled from [46] with a maximum at 78 mins. The MH charging efficiency threshold is $\tau = 0.3$; any node with smaller charging efficiency will not receive any energy. All the MCs and sensors have identical coils with $n_t = 300$ rounds and $r_s = 10$ cm. Wireless charging efficiencies are calculated using the procedures in Section 4.2.2. Each MC is equipped with a 12V battery. At the speed of 1 m/s, the current draws from the battery is 4Ah. Thus, the moving energy consumption is $e_s = 48$ J/m. The simulation is set to run for 4 months' time.

4.5.1 Evaluation of Post-optimization

First, we validate the designs of post-optimization algorithm. We evaluate the evolution of cost during the simulation when the energy requests are within the range of $[10, 120]$ with 3 MCs. Fig. 4.5(a) shows the relation between recharge time and MC's energy cost. As we keep adding new anchors into the recharge route, the total recharge time increases from 600 to 1020 mins and the (weighted) moving costs F_m of MCs also increase. On the other hand, energy charging cost F_c declines as more anchors are introduced into the routes. The evolution of MCs' moving and charging costs validates that adding new anchors can reduce charging costs, elongate the recharge time span and increase MCs' moving costs.



(a) (b)

Figure 4.5: Evaluation of algorithm design. (a) Relationships between energy cost and recharge time. (b) Effectiveness of post-optimization algorithm.

To visualize the progress of post-optimization more clearly, we trace the evolution of total energy cost on different MCs and plot a trend line of their combined average cost in Fig. 4.5(b). The x-axis represents the number of iterations before the algorithm terminates. We observe from the trend line that the post-optimization algorithm can effectively reduce the total energy cost by 12%. During simulations, once the algorithm detects an increase of total system cost after adding an anchor ($\triangle F > 0$), it removes the anchor from the route. New anchors are added when $\triangle F < 0$ and we observe that, on average, the post-optimization algorithm can effectively reduce total cost in each iteration in Fig. 4.5(b). Thus the above results validate that the post-optimization algorithm further improves solutions.

4.5.2 Number of Nonfunctional Nodes

We now demonstrate the advantage of MH by comparing the number of nonfunctional nodes with SN. Once a node depletes its battery and no MC has arrived yet, it is nonfunctional until being recharged. Fig. 4.6(a) compares the number of nonfunctional nodes when $N = 500$. To keep nonfunctional nodes within 5%, at least 5 MCs are needed for SN. In contrast, for MH, only 1 MC is needed and 2 MCs can almost eliminate the chances of battery depletion over the entire operations. The surge of nonfunctional nodes around 10-15 days for SN is because the recharge requests have temporarily exceeded MCs' capability. As the network reaches equilibrium, the curves decline gradually. However, this phenomenon does not appear

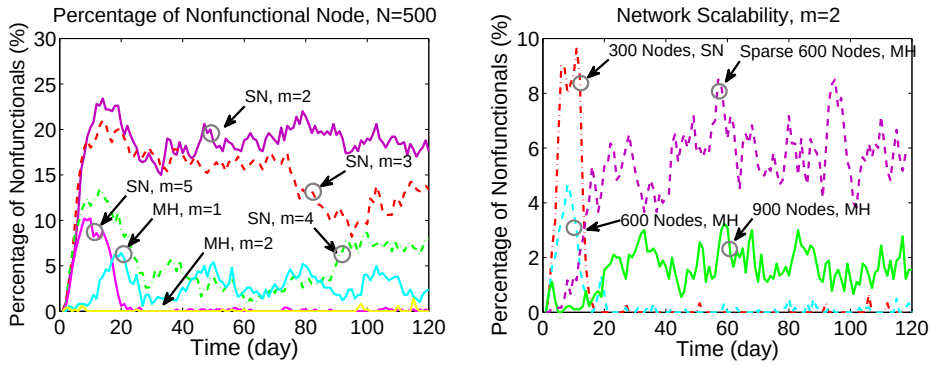


Figure 4.6: Comparison on the number of nonfunctional nodes. (a) Performance comparison when $N = 500$. (b) Scalability evaluation when $m = 2$.

in MH, which shows better robustness even with fewer MCs.

To see the scalability improvement more clearly, we have conducted additional evaluation in Fig. 4.6(b) where we set $m = 2$ and $N = 300$ for SN to provide a baseline and increase N from 600 to 900 nodes. As we can see, the number of nonfunctional nodes still stays below 5%, which indicates a 3-fold increase in the nodes MCs can cover compared to SN (900 nodes vs. 300 nodes). In addition, we have also evaluated the performance of MH in sparse networks where node density is low. To maintain the connectivity among nodes, we double the radius of the field and fix N at 600 nodes. The node density diminishes 75% from 0.3 nodes/m^2 to 0.075 nodes/m^2. We observe that the number of nonfunctional nodes jumps slightly above 5% at equilibrium (not large). The results indicate that the advantage of MH could be weakened in a sparse network with lower node density. However, in the worst case, it is still equivalent to SN without any multi-hop energy relay.

4.5.3 Energy Consumption vs. Replenishment

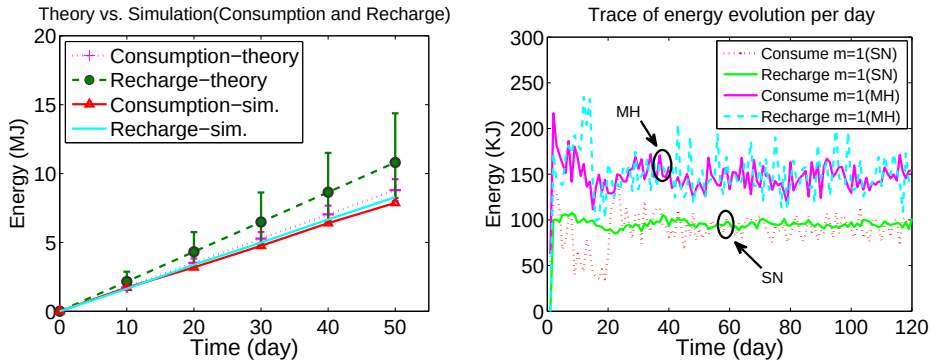
We now evaluate the amount of energy consumption and replenishment and validate the accuracies of our theoretical model. To better exhibit the gaps between curves, we plot the results for the first 50 days. Fig. 4.7(a) depicts energy consumption and replenishment curves for the theoretical and simulation results of MH, $m = 1$. For the theoretical consumption curve, we delineate the mean values with ranges representing standard deviations from the means. For the theoretical replenishment

curve, we use the average charging rate for the battery in [46] as a base and the maximum and minimum rates are indicated by the range of the curve. First, we observe that the replenishment curve is above the energy consumption curve for both theoretical and simulation results. This indicates that MCs can put more energy back into the network than consumed, which is consistent with our observations in Fig. 4.6(a) (that is, almost all the nodes are functional). Our theoretical analysis on the energy consumptions can achieve very high estimation accuracy, as indicated by the small gap between the two curves. The gap between replenishment curves is wider, which is due to the idle time between two successive recharge operations. When the number of MCs is sufficient, the recharge requests are sparse over time and MCs do not need to perform recharge continuously, thus the gap is in between.

We also trace the energy evolution of energy consumption and replenishment in Fig. 4.7(b). For SN, the energy consumption curve quickly drops from the very beginning until it hits a bottom around 20 days. As the MC slowly resolves nonfunctional nodes, these nodes resume normal operation (consume energy) which corresponds to the jump-up of the energy consumption curve at 20 days and the two curves enter an equilibrium after 40 days. On the other hand, for MH, a large gap is observed from SN, indicating 50% more energy being replenished into the network. The improved recharge capability is clearly observed during the first 20 days. That is, in contrast to the slow response in SN, the replenishment curve of MH surges when the energy consumption curve has a sharp decline. It means that whenever nodes are becoming nonfunctional and stop consuming energy, they are quickly recharged by the MC.

4.5.4 System Energy Cost

We now compare the energy cost of MH and SN and explore possible trade-offs between the two schemes. In Fig. 4.8, we evaluate the energy cost needed to maintain the same quality of service (nonfunctional $< 5\%$). In Fig. 4.8(a), for MH, we show energy costs from both node recharging and MC movement, as well as the sum of them and compare with the total cost of SN, while varying N from 250-1000. When $N = 250$, the total cost is almost equivalent while increasing N results in better efficiency for MH. This is because that when node density is higher, more nodes can be recharged simultaneously without the hassle of approaching them one



(a) (b)

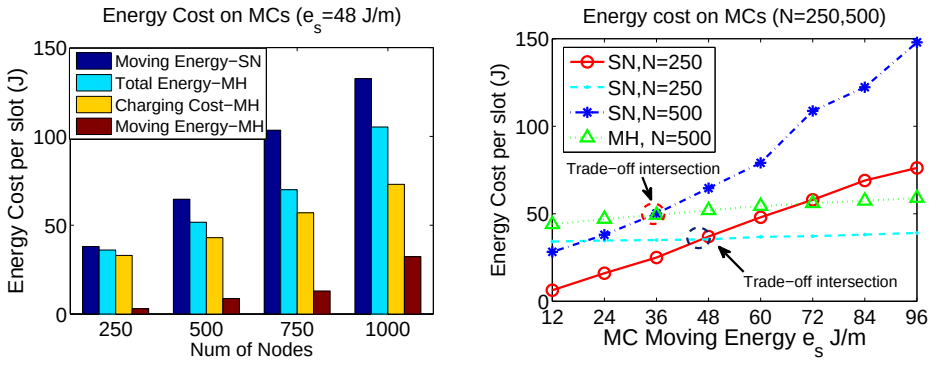
Figure 4.7: Energy consumption vs. replenishment $N = 500$. (a) Theoretical results vs. simulations (MH, $m = 1$). (b) Trace of energy evolution for SN and MH ($m = 1$).

by one. If multi-hop charging cost is much less than moving cost e_s, it is more cost-effective to use MH.

To visualize the trade-offs between MH and SN, we adjust the moving cost e_s from 12 to 96 J/m in Fig. 4.8(b) which represents different energy efficiencies of the MC's battery and motors. For $N = 250$, a trade-off point around 46 J/m is observed. When $e_s < 46$ J/m, SN is more cost-effective. A similar result is observed for $N = 500$ where the trade-off point is around 36 J/m. These results indicate that if energy charging cost can be compensated by shorter moving distances, MH would have less total cost. Based on these results, the network administrator can decide which scheme to use given the system parameters.

4.5.5 Trade-offs between Charging and Moving Costs

In this subsection, we further explore the subtle relations between the two optimization objectives by finding pareto solutions generated by the algorithm. Note that since the problem is NP-hard and intractable in polynomial time, the pareto solutions found by the algorithm are in fact suboptimal and within the approximation bounds discussed in Section 4.3.3. As shown in [91], a minimizer of the weighted combination of objectives in Eq. (4.4) and Eq. (4.5) is a pareto optimal solution to the original bi-objective problem in Eq. (4.3). To explore the solution space, we vary the weights w_1 and w_2 from 1 to 10 in small increments and delineate



(a) (b)

Figure 4.8: Comparison of energy cost on MCs to maintain nonfunctional nodes under 5%. (a) $e_s = 48$ J/m. (b) $e_s = 24$ to 96 J/m.

those solutions of MCs' charging cost and moving cost in Fig. 4.9(a). The y-axis represents MC's charging cost F_c (the first objective) and the x-axis represents MC's moving cost F_m (the second objective). In the post-optimization algorithm, the choice of different weights allows the MC to explore different solutions and it has a direct impact on the decision value $\triangle F$ as well as the recharge routes. From Fig. 4.9(a), we can see that the points along the *pareto-frontier* form a contour to bound the feasible solution space. The *pareto-frontier* consists of solutions that cannot be surpassed by any other alternative solutions. As analyzed in our algorithm designs, a trade-off is observed between the two optimization objectives. That is, when the MC's moving cost is reduced, the charging cost has to increase and vice versa.

Similarly, we also examine the trade-offs between the total system cost and recharge delay. As shown in Fig. 4.9(b), if we want to reduce system cost, a certain amount of nodes would suffer from extended recharge delay. These results validate our designs and analysis in the algorithm as we aim to reduce system cost as much as possible while minimizing the chances of battery depletion.

4.5.6 Evaluation of Recharge Delay and Service Interruptions

Since some nodes may have similar energy consumption rates, it is possible for them to request recharge at the same time. If the requests are scattered at different locations, due to limited multi-hop charging range, the MC may not be able to cover all the requests at once. In this case, late recharge is inevitable and its duration is

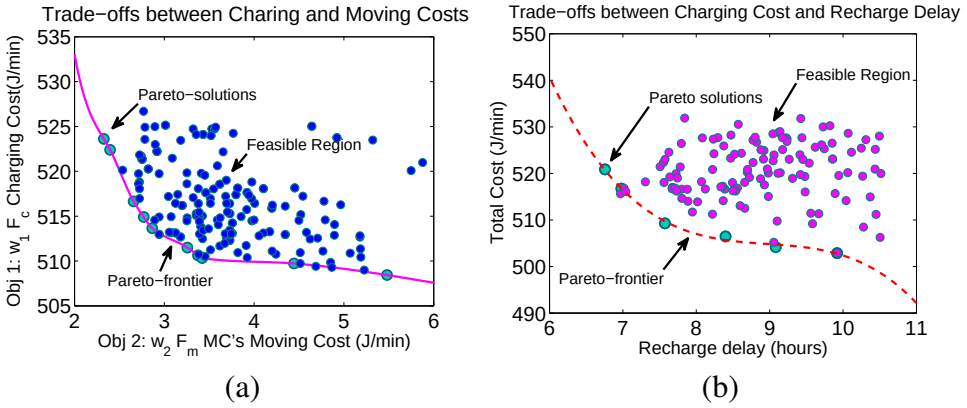
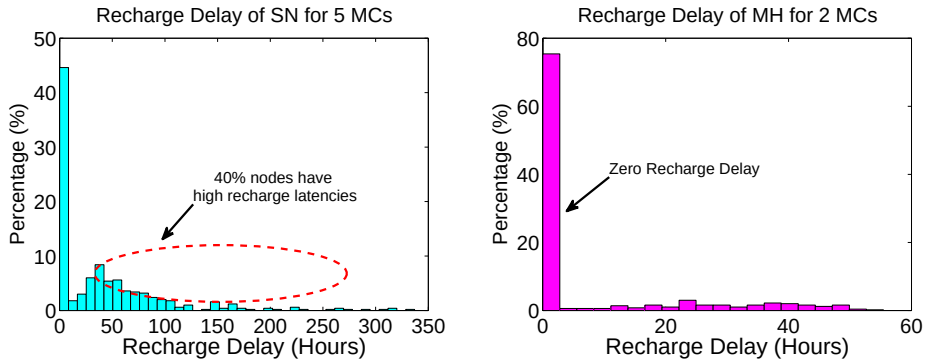


Figure 4.9: Evaluation of trade-offs in the network. (a) Trade-offs between MC's charging and moving costs. (b) Trade-offs between total system cost and recharge delay.

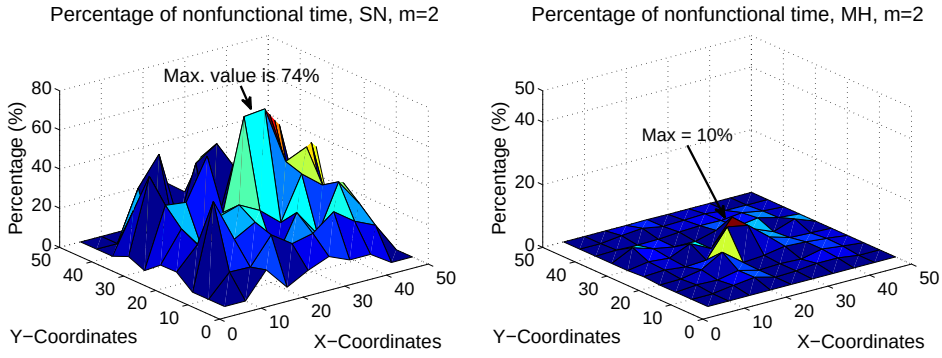
measured by recharge delay. Fig. 4.10 compares recharge delay of SN and MH. Recall from Section 4.5.2 that for $N = 500$, SN $m = 5$ and MH $m = 2$ have comparable nonfunctional percentage under 5%. For SN, Fig. 4.10(a) shows that some nodes would experience more than 50 hours of recharge delay. In other words, it means that once a node has requested for recharge, there are at least 50 nodes in MCs' service queues ahead of this node waiting for recharge. In contrast, Fig. 4.10(b) presents much better results with MH while the number of MCs is only $m = 2$. We can see that a majority (almost 80%) of nodes have even no recharge delay and very few nodes have recharge delay over 20 hours. The huge improvements are due to extended charging range which upgrades the single-server queue of SN into a multi-server queue in MH. The MCs have extra capabilities to handle energy requests in the vicinity thereby expediting the entire recharging process.

We also present the percentage of nonfunctional durations in a geographical view in Fig. 4.11 where x and y axes are field coordinates. The time duration while a node is in nonfunctional status greatly impacts the network operation. Such nodes are not able to sense the environment and may miss important events, constituting service interruptions. For fair comparison, we set $N = 500$ and $m = 2$ for both cases. SN results in a maximum of 75% time in nonfunctional status with the average over 40% widely spreading on the entire field. In sharp contrast, MH has the maximum of only 10% with an average below 3%. This shows that MH has significantly less service interruptions than SN.



(a) (b)

Figure 4.10: Comparison of recharge delay when SN and MH have similar non-functional percentage. (a) SN, $m = 5$. (b) MH, $m = 2$.



(a) (b)

Figure 4.11: Comparison of nonfunctional nodes' durations $N = 500, m = 2$. (a) SN. (b) MH.

4.6 Discussions

In practice, the effectiveness of multi-hop wireless charging could be affected by node density and topology. For sparse networks, it is possible that a node has no immediate neighbors to relay energy. In this case, our scheme still works, but reduces to a single node recharge method. In fact, due to the declining manufacturing cost of sensor nodes, and the needs to ensure robustness against node and communication failures and faults, they are usually deployed at densities much higher than needed for monitoring. Some applications even require k-coverage, where each point on the field is monitored by at least k sensors. For example, to detect forest fires, different parameters across multi-dimensions are collected to create a potential ignition map of the forest. For better reliability, indicator for each location is usually calculated based on the readings from multiple sensors. High density is desired for load balancing purposes as well. For example, nodes have higher densities near the sink so they can take turns forwarding data to extend network lifetime and improve robustness. Such high density deployment presents opportunities to apply our multi-hop recharging method. In reality, multi-hop wireless charging can make use of this node redundancy to improve network lifetime.

Another practical challenge is that the node topology may cause misalignment of sensor coils and degrade charging efficiency. Fortunately, recent research using coil arrays provides position-free solutions to the misalignment problem and it is found that charging efficiency increases from 4.8% to 64% [92]. Another option is to use mechanisms similar to "sliding antennas" [93] to fine tune and align the orientations of coils on demand.

The past several years have witnessed the rapid advance and maturity of wireless charging technology. One prominent example among others is *WiTricity*, a major player in the wireless charging market. It has recently released multiple products for consumer electronics, automobiles, medical and industrial applications. Its research and standardization efforts in wireless repeaters have effectively increased charging distance, scale and efficiency[94]. Our framework works under the same principle of resonant repeaters, which can be embedded under the floor, table or even walls to hop power in a room. Besides, researchers have accomplished a new milestone to extend charging distance significantly. They invented the *Dipole Coil Resonant System* based on refined coil structures that can power 40 smartphones

from 5 meters and a single device from 9 meters[95] (close to sensors' transmission range). Combined with resonant repeaters for energy relay, energy delivery over multiple hops as studied in our framework is not just feasible in principle, but could soon be implemented based on all such recent technology advances.

4.7 Conclusions

In this chapter, we employ resonant repeaters to improve the efficiency and scalability of recharge in WRSNs. We present detailed procedures to calculate multi-hop wireless charging efficiency based on the laws in physics and electronics that have been overlooked by previous studies. We formulate the recharge scheduling problem into a multi-objective optimization problem, which is NP-hard. To achieve low-complexity, we propose a two-step approximation algorithm with bounded ratio for each objective followed by a post-optimization algorithm to further reduce the system cost. Finally, we evaluate the proposed framework by extensive simulations and compare with previous works. The results reveal much better network scalability and performance of our algorithm.

Chapter 5

Combine Wireless Charging and Solar Energy Harvesting

5.1 Introduction

From the previous chapters, we have seen that wireless charging is a promising technique that can power hundreds of nodes reliably from single to multi-hop wireless energy transmission. Since the rising energy demands in the network also increase the risks of electromagnetic exposure [38]. As a result, energy transmitters must comply with standards from Federal Communication Commission and limit their emitting power to human safe power densities ($< 1mW/cm^2$[40]). Nevertheless, nodes at data aggregation points (such as cluster heads in a clustered WSN) usually consume very high energy ($10 - 100mW$) due to data traffic. Thus limiting transmission power at wireless chargers can easily cause battery depletion and network interruption on such nodes.

In the meanwhile, there is another competitive technique for environmental energy harvesting that has low risk yet much higher power density. As shown in [96], among a variety of harvesting techniques, solar harvesting through photovoltaic conversion enjoys the highest power density ($15mW/cm^2$), which is renewable and risk-free. In practice, a solar panel commensurate with sensor's size is sufficient to meet the energy demands of cluster heads. However, availability of sunlight is subject to dynamics in the environment. Not only weather conditions would have a direct impact on the harvesting rates, but also a series of spatial-temporal factors

such as sunrise, sunset times, locations and their surroundings would affect deployment decisions of harvesting sensors.

Realizing the pros and cons of both technologies, in this chapter, we propose a hybrid framework to make use of their advantages and overcome their drawbacks. In the new framework, a majority of nodes are wireless-powered nodes (WNs) due to the low costs of charging coils. On the other hand, due to the relatively higher manufacturing and deploying costs, a small number of solar-powered nodes (SNs) are responsible for aggregating data. Normally, a fleet of MCs roam over the field to serve recharge requests from WNs and collect data from SNs. In contrast to WNs, SNs' energy from the ambient source is self-sufficient. This scheme provides effective energy replenishment at cluster heads so that they can complete high volume data transmissions. Meanwhile, the rest of WNs can be recharged by MCs on demand. The hybrid framework raises several new challenges. First, how many SNs are needed and where should we deploy them such that the total cost is minimized? Second, how to guarantee robustness of the network when sunlight is unavailable (e.g., cloudy/raining days)? Third, how to schedule the MCs to complete wireless charging and data gathering in the same tour? Can we further optimize system cost and improve network performance compared to previous approaches?

To answer these questions, in this paper we first study a placement problem in discrete form where SNs are deployed among the known WN locations. We formulate it into a *facility location problem* [97–100] to minimize the total cost of packet routing and node deployment. Due to its NP-hardness, we use the primal-dual method to develop a distributed $1.61(1+\epsilon)^2$-factor algorithm suitable for WSN applications based on the centralized paradigm in [99]. Then we show the locations of SNs can be further optimized within a cluster in continuous space and propose an iterative mechanism based on the Weiszfeld algorithm [101]. We also demonstrate how our algorithms can adapt to seasonal variations of sunlight by adjusting their locations accordingly. Second, we theoretically analyze network energy balance and propose a method to maintain such balance during cloudy/raining days. We find that using a smaller cluster size is effective to reduce energy consumptions and develop a distributed algorithm to appoint some selected WNs as temporary cluster heads until solar energy becomes available. Finally, we optimize MCs' routes for the joint wireless charging and mobile data gathering problem. Different from [41] in which MCs visit exact node locations, we point out that for

data gathering, it is only necessary for MCs to move into SN's transmission range. Based on this observation, we give a polynomial-time route improvement algorithm that can take shortcuts through SN's neighborhood to further reduce the cost. Since some nodes may require expedited recharge due to limited lifetime, we allow the MCs to perform *partial recharge* rather than to refill batteries to full capacity [33, 36, 41, 71, 83, 106]. Our objective is to simultaneously maximize the time MCs spend in recharging and prevent nodes from energy depletion, which is formulated into a Linear Programming problem. For easy implementation on MCs, we propose an efficient algorithm based on the particular structure of the lifetime constraints and validate its near-optimality by extensive simulations.

The rest of the chapter is organized as follows. Section 5.2 studies related works. Section 5.3 presents the network model and assumptions. Section 5.4 studies the placement problem of SNs. Section 5.5 provides theoretical analysis and discusses how to maintain energy balance using WNs. Section 5.6 optimizes MC's migration routes and explores partial recharge for further improvements. Section 5.7 evaluates the new framework and Section 5.8 concludes the chapter.

5.2 Related Works

In this section, we discuss some related works of wireless charging, environmental energy harvesting and tour planning algorithms.

5.2.1 Wireless Charging

Wireless charging technology has developed at an unprecedented pace recently. From the earlier pad-based charging systems (requiring close contact) [30] to the most recent mid-range wireless charging that allows energy delivery up to several meters [107], the technology is envisioned to revolutionize charging without the hassle of power cables.

The application of wireless charging has been also considered in battery-powered WSNs [33, 36, 38, 41, 71, 83, 106, 108, 109]. In [41], optimization of wireless charging and mobile data gathering is studied by combining the two utilities on a single MC. Based on the products from Powercast [31], wireless charging is explored in [33] to evaluate the new impact on deployment patterns and packet rout-

ing. In [36], an $\mathcal{O}(k^2k!)$ greedy algorithm (where k is the number of nodes) is developed to maximize network lifetime whereas the moving cost of wireless chargers is not considered. Upon realizing this problem, minimization of chargers' moving cost is considered in [71, 83]. In [83], a distributed real-time energy information gathering protocol is proposed first. Then based on the updated energy information, a weighted-sum algorithm considering nodes' lifetimes and MCs' moving costs is developed. The problem is further extended to jointly consider MCs' recharge capacities and sensors' dynamic battery deadlines in [71]. In [106], a joint routing and wireless charging scheme is proposed to improve network utilization and prolong network lifetime. Similarly, in [108], deployment problems of wireless chargers are studied to extend network lifetime.

However, since many wireless charging systems are radiation-based, exposure to radio-frequency energy implies potential health risks. The negative biological effects include increased possibility of tumor and other impairments. Therefore, the Federal Communication Commission (FCC) has imposed a regulatory limit to restrict the maximum transmitter output power to be under 1W and the maximum effective isotropic radiated power (EIRP) under 4W [40]. In practice, the omnidirectional propagation of electromagnetic wave causes health risks in all directions. Due to regulations of emitting power level on a single charger, multiple wireless chargers are considered in [38, 109]. In [38], the problem of how to adjust the transmitting power of wireless chargers such that overall electromagnetic exposure does not exceed a threshold is studied. A charger placement problem is formulated to guarantee all the locations to satisfy the safety requirements. In [109], optimization of "useful" energy transferred from chargers to nodes under safety concerns is considered. However, since the accumulative emitting power from multiple chargers is still restricted, nodes cannot perform energy-consuming applications.

5.2.2 Environmental Energy Harvesting

Environmental energy harvesting provides another alternative to extend network lifespan. Renewable energy from the environment such as solar, wind, vibration and thermal can be used effectively to power sensor nodes. Due to the dynamics in environmental energy sources, a majority of previous efforts focus on energy management of sensors [110–113]. In [110], a power management scheme to maximize

sensors' duty cycles is proposed. Energy is profiled based on moving average to predict future income and the duty cycles are adjusted accordingly. Similarly, methods from adaptive control theory are adopted in [111] to deal with environmental energy dynamics. Harvesting energy from light sources indoors is investigated in [112]. Energy allocation algorithms based on experimental measurements are developed to optimize energy storage on sensors. Joint energy management and resource allocation is considered in [113] for optimizing network performance. A local algorithm is developed to adjust sensors' sampling rates and adapt to the battery states. However, an inevitable drawback of these earlier works is that the network operations would be disrupted when those ambient energy sources are unavailable (e.g,. during cloudy/raining days in a solar harvesting network). In contrast, our proposed framework in this paper incorporates a combination of hybrid energy sources so that steady and productive network performance can be guaranteed.

5.2.3 Tour Planning of Mobile Vehicles

Tour planning of mobile vehicles for data collection in WSNs has been studied, see, for example in [20, 114–116]. The problem shares similarities to the well-known Traveling Salesmen Problem with Neighborhood (TSPN) [117, 118, 120] in which the salesman aims to find the shortest tour through city neighborhoods of arbitrary shapes. In the context of WSNs, due to the omnidirectional propagation of electromagnetic waves, the neighborhoods are usually assumed to be circles. In [20], the tour planning problem is formulated into a mixed-integer program and a spanning tree covering algorithm is proposed. However, the algorithm requires the vehicles to visit exact node locations for data gathering, which is usually unnecessary in practice. If the node's transmission range is considered, the performance can be further improved. The method proposed in [114] attempts to improve current solutions for TSPN. It first determines the shortest TSP routes among sensors without considering their transmission ranges. Then it searches along transmission boundaries to find the best hitting points such that the tour length is minimized. In [115], the problem is formulated into a label-covering tour problem. A complete graph is first constructed to represent all possible paths between nodes. The objective is to find an optimal path that covers all the nodes in their transmission ranges. Although an approximation algorithm is proposed for this NP-hard problem, the approximation

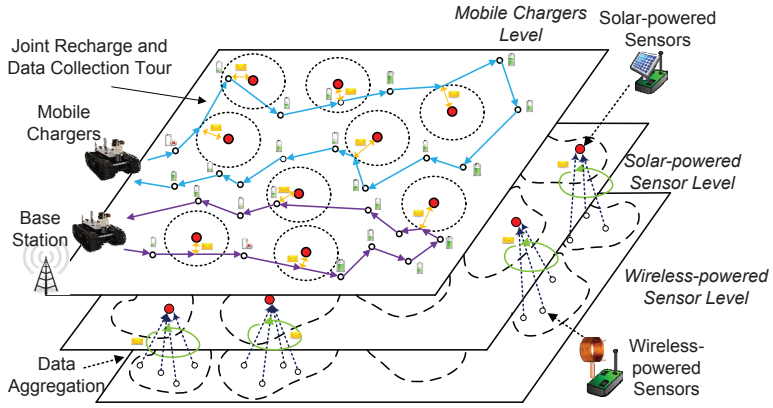


Figure 5.1: Overview of a three-level network hierarchy.

ratio also increases with the number of nodes in the tour. In [116], a progressive tour construction method is proposed. The method exploits the overlaps of transmission ranges of neighboring nodes so that the mobile vehicle can take shortcuts for cost savings. This method works effectively when there exist overlaps among nodes' transmission ranges. Different from [115, 116] where nodes' transmission ranges may have overlaps, in this paper, multi-hop clusters are formed such that one-hop transmission neighborhoods of cluster heads (SNs) are disjoint. In addition, different from all the previous works, a migration tour may incorporate both WNs and SNs in our framework. Therefore, we focus on how to minimize tour lengths by taking advantage of SNs' neighborhoods in a joint wireless charging and data gathering tour.

5.3 Network Model and Assumptions

In this section, we give an overview of the network model and assumptions of the new framework. Based on the energy sources, there are two types of nodes in the framework: wireless-powered nodes and solar-powered nodes. For brevity, we denote them by "WNs" and "SNs" respectively. Based on the functionality of network components, we divide the network into three hierarchical levels as shown in Fig. 5.1: wireless-powered sensor, solar-powered sensor and mobile charger levels.

The bottom level (wireless-powered sensor level) has N WNs uniformly randomly distributed on a square field of side length L. Since charging coils can be

cheaply manufactured, WNs are deployed in high density to perform basic sensing missions such as environmental readings, target tracking, etc. In particular, to monitor location-dependent solar radiation strength, each node has an illuminance sensor and reports its reading with other data to the base station. The energy consumption for transmitting a packet follows the widely adopted model [70], $e_t = e_0 + e_1 d_r^{\alpha}$, where e_t is the transmitting energy, e_0 and e_1 are the energy consumed in electronics and amplifiers, d_r is the distance between the transmitter and the receiver ($d_r \leq r$, r is the transmission range), and α is the path loss exponent. To perform sensing tasks, sensors also consume e_s energy for each packet. For message exchange, we assume the network is connected. Each WN generates packets independently following a Poisson process with average rate λ. Each WN is powered by a 750 mAh rechargeable NiMH battery and the recharge time T_r is 78 minutes. If the energy drops below a threshold, e.g., 50%, it sends out a request to the MCs for scheduling energy replenishment.

The middle level (solar-powered sensor level) is comprised of self-sustaining, energy harvesting nodes. Normally, when solar energy income is sufficient, SNs act as cluster heads for aggregating sensed data. However, when energy supply is not enough during cloudy/raining days, the network re-selects WNs as cluster heads so they can rely on consistent wireless energy supply from the MCs. To minimize routing and deploying costs, SNs should be deployed at advantageous locations. Due to varying nature of sun's angles during a year, building obstructions and tree shades may exhibit different spatial-temporal patterns. Therefore, SN locations should be re-calculated based on the updated data once in a while (e.g., several weeks). The energy harvesting rates are modeled according to [103], which will be discussed in Section 5.5. We assume the size of solar panel is chosen to be large enough to harvest enough energy for aggregating all the data. We adopt a commercially available panel of $10 \times 10 cm^2$ size which is connected to a 3V, 2150 mAh lithium-ion battery.

The top level (mobile charger level) manages a fleet of m MCs through the base station. Proposed in [41], MCs are equipped with high-capacity batteries and powerful antennas for energy replenishment and data collection. Coordination among the MCs is conducted via long range communications to exchange status, position, energy request, etc. They also have positioning devices (e.g., GPS, gyroscope, etc) to locate sensor positions so that they can approach them in close proximity for

Table 5.1: List of Notations

Notation	Definition
N	Number of wireless-powered sensors
s	Number of solar-powered sensors (calculated by algorithm)
m	Number of mobile chargers
L	Side length of squared sensing field
r	Transmission range of sensor nodes
e_t, e_r, e_s	Energy consumption to transmit, receive, generate a packet
λ	Average packet rate of Poisson distributed traffic
p_s, p_m	Monetary expenses of SNs and MCs, respectively
C_h	Battery capacity of sensor nodes
T_r	Recharge time of sensor's battery from zero to full capacity
v	Average moving speed of MCs

wireless charging at high efficiency. We assume each node is only recharged by one MC at a time and the emitting power at the wireless charger also complies with FCC's regulations to minimize health risks. Depending on updated geographical solar energy distribution, MCs can deploy SNs at appropriate locations. In case some nodes in the recharge sequences are bound to deplete their energy, the MCs can expedite the process by partially refilling their batteries. Since SNs and MCs are the main components to sustain network operations and their manufacturing costs are much higher than WNs, we assume their monetary expenses are p_s (for SNs) and p_m (for MCs), respectively and $p_s < p_m$. Finally, we summarize some important notations used in this paper in Table 5.1.

5.4 Solar-powered Sensor Layer: Placement Problem

In this section, we study the Solar-powered sensor Placement Problem (SPP). It determines where to place SNs such that the total cost is minimized. In SPP, there are two types of costs: packet routing cost and sensor deploying cost. According to the energy model on packet transmissions [70], the routing cost is proportional to the number of hops thus the distance to the cluster head. The deploying cost is related to expense p_s and the strength of sunlight at a specific location. Since harvested energy exhibits slow variation due to seasonal changes of sun's angle,

solar radiations at a fixed location also change slowly during a year. According to the illuminance readings from sensors, we denote the average solar strength at sensor location i by l_i. The deploying cost can be defined as the ratio p_s to l_i, which can be explained as the price we pay to gather a unit of solar energy from a specific location. If more SNs are deployed, nodes would have less relaying distance to the SNs. Thus, less routing cost can be achieved. On the other hand, more SNs would increase the deploying cost so our objective is to minimize the sum of routing cost and deploying cost.

These observations suggest that our problem is in close analogy to the classic *Facility Location Problem* (FLP) [97–100], which is NP-hard. In FLP, a set of facilities and cities are given. There is an opening cost associated with each facility and a transportation cost between any pair of facility and city. The goal is to connect each city to an open facility while minimizing the sum of transportation cost and opening cost. Due to NP-hardness, obtaining an optimal solution in polynomial time is infeasible. In practice, approximation algorithms that can achieve certain factors to the optimal solution are always preferred. After the first polynomial time 3.16-approximation algorithm is proposed in [97], there has been encouraging progress in improving the approximation ratio and running time. An $\mathcal{O}(n^2 \log n)$ algorithm is proposed in [98] with an approximation ratio of 3 based on the primal-dual method. This bound is soon improved by [99] from 1.86 to 1.61 which is very close to the upper limit 1.46-ratio that polynomial-time algorithms can achieve [100].

However, the aforementioned efforts only focus on centralized algorithms whereas distributed implementation of FLP is rare in the literature. In dynamic wireless environments, a centralized algorithm requires the collection of variables across multiple dimensions to form global knowledge, which is usually time-consuming and not cost-effective. To this end, we propose a distributed $1.61(1+\epsilon^2)$-approximation algorithm based on the centralized approach in [99]. Next, we first formalize SPP and illustrate the centralized 1.61-approximation algorithm. Then we propose a distributed version of the algorithm. Since the locations of SNs are not necessarily constrained to WN locations, we further improve the solution using the Weiszfeld algorithm [101]. Finally, we discuss how to re-deploy SNs in order to adapt variations in solar strength at different times.

5.4.1 Placement of Solar-powered Sensors

In this subsection, we formalize the problem and describe both the centralized and distributed versions of the algorithm.

Centralized Placement Algorithm

First, let us formalize SPP. We denote the sets of SNs and WNs by $\mathcal{S}$ and $\mathcal{N}$, respectively. For simplicity, we first study discrete SPP by assuming SNs can only be co-located at WNs' locations, $\mathcal{S} \subset \mathcal{N}$. We consider a graph $G = (V, E)$ where vertices are sensor nodes and edges are connections. c_{ij} is the routing cost between nodes i and j, which is the energy consumed for transmitting packets. f_i is the deploying cost of SN i, and $f_i = p_s/l_i$, where l_i is the solar strength at node i. Since the energy consumed by WNs for data transmissions ultimately comes from the MCs, to convert c_{ij}'s energy units into monetary cost, we scale c_{ij} by how much the base station has paid for consuming per watt of energy to recharge MC's battery. The decision variable x_{ij} is 1 if WN j is assigned to SN i; otherwise, it is 0. y_i is 1 if we place an SN at i; otherwise, it is 0. Initially, all WNs are candidate locations for SNs. Our objective is to minimize the total cost by finding the locations for SNs.

$$\mathbf{P1}: \quad \min \sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{N}} c_{ij} x_{ij} + \sum_{i \in \mathcal{S}} f_i \tag{5.1}$$

Subject to

$$\sum_{i \in \mathcal{S}} x_{ij} \geq 1; j \in \mathcal{N} \tag{5.2}$$

$$x_{ij} \leq y_i; i \in \mathcal{S}, j \in \mathcal{N} \tag{5.3}$$

$$x_{ij}, y_i \in \{0, 1\}; i \in \mathcal{S}, j \in \mathcal{N} \tag{5.4}$$

Constraints (5.2) and (5.3) impose that each WN is only connected to one SN. A centralized 1.61-approximation algorithm is proposed in [99]. Since we will use it as a guideline for designing the distributed algorithm, we briefly describe the centralized algorithm below. For each SN i, we introduce a set $\mathcal{B}_i$ to represent its connected WNs ($\mathcal{B}_i \subseteq \mathcal{N}$). In each step, the algorithm selects the node i^* with the

Table 5.2: Centralized 1.61-factor SN Placement Algorithm

Input: Set of WN $\mathcal{N}$.
Output: Set of SN $\mathcal{S}$ and $\mathcal{B}_i$, $i \in \mathcal{S}$.
While ($\mathcal{N} \neq \emptyset$)
Find $i^* = \arg\min_{i \in \mathcal{N}} \big(\sum_{j \in \mathcal{B}_i} c_{ij} + f_i - \sum_{j \in \mathcal{B}'_i} (c_{i'j} - c_{ij}) \big) / |\mathcal{B}_i|$.
Deploy i^*, connect $\forall j \in \mathcal{B}_i \bigcup \mathcal{B}'_i$ to i^*, $\mathcal{N} \leftarrow \mathcal{N} - \mathcal{B}_i$.
End While

minimum average cost

$$i^* = \arg\min_{i \in \mathcal{N}} \big[\sum_{j \in \mathcal{B}_i} c_{ij} + f_i - \sum_{j \in \mathcal{B}'_i} (c_{i'j} - c_{ij}) \big] / |\mathcal{B}_i|. \tag{5.5}$$

Node i' is a deployed SN that WN j has already connected to. $\mathcal{B}'_i$ is the set of these already connected WNs which would be benefited by altering their connections to the new SN i. Hence, a saving of routing cost $\sum_{j \in \mathcal{B}'_i} (c_{i'j} - c_{ij})$ should be deducted from the total cost. To find the minimum average cost for each candidate SN i, we can sort the cost in an ascending order and select the least one. This would result in $|\mathcal{B}_i|$ WNs being chosen each time. After i^* is found, we deploy an SN at its location and update all the WNs in $\mathcal{B}_i \bigcup \mathcal{B}'_i$ to connect with node i^*. The iteration continues to add SNs until all WNs are connected to them. The centralized algorithm has $\mathcal{O}(N^3)$ complexity and is summarized in Table 5.2.

($1.61 + \epsilon^2$)-factor Distributed Algorithm

To understand the nature of the problem, we first formulate the dual problem of **P1**. The introduction of dual variables will help design the distributed problem.

$$\mathbf{P2}: \quad \max_{j \in \mathcal{N}} a_j \tag{5.6}$$

Subject to

$$a_j - b_{ij} \leq c_{ij}; i \in \mathcal{S}, j \in \mathcal{N} \tag{5.7}$$

$$\sum_{j \in \mathcal{N}} b_{ij} \leq f_i; i \in \mathcal{S} \tag{5.8}$$

$$a_i, b_{ij} \geq 0; i \in \mathcal{S}, j \in \mathcal{N} \tag{5.9}$$

Table 5.3: Distributed $1.61(1+\epsilon)^2$-factor Algorithm for WN j

If j is not connected to any SN,
send a message to i with offer $a_j \leftarrow \max(a_j - c_{ij}, 0)$.
Else If j is connected to a deployed SN i',
send a message to i with offer $a_j \leftarrow \max(c_{i'j} - c_{ij}, 0)$.
End If
Raise offer $a_j \leftarrow (1+\epsilon)a_j$.

Here, we can think the dual variable a_j as a monetary offer from node j to the total expense for deploying an SN. Constraints (5.7)-(5.8) can be combined into $\sum_{j\in\mathcal{N}} \max(a_j - c_{ij}, 0) \leq f_i$ for SN $i \in \mathcal{S}$. It means that if offer a_j is raised for all the WNs at the same pace, and at the moment the total offer minus the total cost is equivalent to f_i, an SN can be successfully deployed at i. This method is known as the *dual ascent procedure* [99] and its 1.61-factor approximation is proved in [99] following the centralized paradigm. Based on [99], we propose a distributed $1.61(1+\epsilon)^2$-approximation algorithm next, where ϵ is a small fraction greater than zero.

First, WNs will send out their offers to SNs. If a WN is not connected to any $i \in \mathcal{S}$, the value of the offer is set to $\max(a_j - c_{ij}, 0)$; otherwise, the value is set to $\max(c_{i'j} - c_{ij}, 0)$. At the other side, SNs receive the offering messages from WNs. If an SN j is not yet deployed while its received total offers $\sum_{j\in\mathcal{N}} \max(a_j - c_{ij}, 0)$ are greater than or equal to f_i, we can successfully deploy an SN at i. If j is deployed and the offer value $a_j = c_{ij}$, we connect j to i by sending a connection message to j. In the next round, WNs increase their offers a_j by a ratio of $(1+\epsilon)$. After the locations for SNs have been calculated, the WNs send out deploying requests to the MCs. Then an MC is dispatched from the base station to deploy SNs at their designated locations. The distributed algorithm on WNs and SNs is summarized in Table 5.3 and Table 5.4 and has the following properties.

Property 1: In principle, the distributed and centralized algorithms are equivalent.

Proof. We sequentialize the distributed algorithm into execution rounds. For the distributed algorithm, each round consists of a number of message sending and receiving by respective WNs and SNs. In each round, the total offers received from all the nodes are $\sum_{j\in\mathcal{N}} a_j = \sum_{j\in\mathcal{N}} c_{ij} + f_i$. For some nodes already connected,

Table 5.4: Distributed $1.61(1+\epsilon)^2$-factor Algorithm for SN i

Receive offering messages from WNs.
If i is not yet deployed AND $\sum_{j\in\mathcal{N}} \max(a_j - c_{ij}, 0) \geq f_i$
deploy an SN at i's location, $\mathcal{S} \leftarrow \mathcal{S} + i$.
$\forall j \in \mathcal{N}$, $\mathcal{B}_i \leftarrow \mathcal{B}_i + j$. Connect j to SN i.
Else If i has been deployed AND $a_j = c_{ij}$
connect j to i, $\mathcal{B}_i \leftarrow \mathcal{B}_i + j$, $\mathcal{B'}_i \leftarrow \mathcal{B'}_i - j$.
Send a connection request message to j.
End If

the new offers are $\sum_{j\in\mathcal{N}} a_j = \sum_{j\in\mathcal{N}} c_{i'j} - c_{ij}$, which should be deducted from the total offers to reflect the adjusted value. We can see that this result is exactly the term in (5.5). Since the offer value is increased at a rate $(1+\epsilon)$, an SN that meets the lowest total offer will be selected in the earliest time, which is equivalent to selecting the least average cost in the centralized algorithm. Therefore, we can see that the mechanism of the distributed algorithm is analogous to the centralized algorithm in [99]. □

Property 2: The distributed algorithm terminates in $\mathcal{O}(\log_{1+\epsilon} f_m)$ rounds, where $f_m = \max f_i, i \in \mathcal{N}$. The total message overhead is $\mathcal{O}((\log_{1+\epsilon} f_m)N^2)$.

Proof. Clearly, when the offering amount a_j increases at a rate $(1+\epsilon)$, reaching the maximum value of f_i requires $\mathcal{O}(\log_{1+\epsilon} f_m)$. In each round, the message overhead is bounded by $\mathcal{O}(N^2)$ so the overall message overhead is $\mathcal{O}((\log_{1+\epsilon} f_m)N^2)$. □

Property 3: The distributed algorithm achieves $1.61(\epsilon+1)^2$-factor approximation to the optimal solution.

Proof. First, denote optimal offers in the centralized algorithm [99] by a_j and the distributed algorithm by a'_j, $j \in \mathcal{N}$. A *Factor Revealing LP* is constructed by [99]. For SN i, $k = |\mathcal{B}_i|$, the optimal solution is to solve the following maximization problem

$$\mathbf{P3}: \quad z_k = \max \sum_{j=1}^{k} a_j / (\sum_{j=1}^{k} c_{ij} + f_i) \tag{5.10}$$

Subject to

$$a_j \leq a_{j+1}, \forall j \in \{1, 2, \ldots, k-1\} \tag{5.11}$$

$$\sum_{j=1}^{k} \max(a_j - c_{il}, 0) \leq f_i, \forall l \in \{1, 2, \ldots, k\} \tag{5.12}$$

$$a_j \leq a_l + c_{ij} + c_{il}, \forall j, l \in \{1, 2, \ldots, k\} \tag{5.13}$$

$$a_j, c_{ij}, c_{il}, f_i \geq 0, \forall j, l \in \{1, 2, \ldots, k\} \tag{5.14}$$

For the maximization problem to be bounded, a_j should also be bounded. It implies that at least one of the constraints of (5.12) and (5.13) is tight (i.e., changing from inequality into equality). *Case 1*: Eq. (5.12) is tight; *Case 2*: Eq. (5.13) is tight, thus a_l is also bounded.

For the distributed algorithm, we can formulate it into a similar Factor Revealing LP except that constraint (5.12) becomes $\sum_{j=1}^{k} \max\left(\frac{a'_j}{1+\epsilon} - c_{il}, 0\right) \leq f_i$ and constraint (5.13) becomes $\frac{a'_j}{1+\epsilon} \leq a'_l + c_{ij} + c_{il}$ since increasing offers at the same pace in the centralized scheme would deploy SN i at most $(1+\epsilon)$ time earlier compared to the distributed algorithm. For Case 1, since $f_i \geq 0$ and the constraint is tight, $a_j - c_{il} \geq 0$ and $\frac{a'_j}{1+\epsilon} - c_{il} \geq 0$ hold for the centralized and distributed algorithms, respectively. Thus, $\frac{a'_j}{a_j} \leq 1 + \epsilon$. For Case 2: $\frac{a'_j}{1+\epsilon} = a'_l + c_{ij} + c_{il}$ and $\sum_{l=1}^{k} \max(\frac{a'_l}{1+\epsilon} - c_{il}, 0) \leq f_i$ is also tight to bound a'_l. The latter suggests that the ratio in Case 1 $\frac{a'_l}{a_l} \leq 1 + \epsilon$ can be applied here:

$$\frac{a'_j}{1+\epsilon} \leq a_l(1+\epsilon) + c_{ij} + c_{il} \leq (1+\epsilon)(a_l + c_{ij} + c_{il}) \tag{5.15}$$

Then by taking the ratio of Eq. (5.15) to Eq. (5.13), we have $\frac{a'_j}{a_j} \leq (1+\epsilon)^2$ for Case 2. Since the approximation ratio to the optimal solution a^*_j is proved by [99], $\frac{a_j}{a^*_j} \leq 1.61$. Thus, our algorithm has at most $\frac{a'_j}{a^*_j} \leq 1.61(1+\epsilon)^2$ approximation to the optimal solution. □

5.4.2 Placement of SNs in Continuous Space

Since MCs enjoy the freedom to place SNs at any feasible location in the field, in this subsection, we explore the placement of SNs in continuous space. Clearly, the continuous problem is much harder than its discrete version (SPP). Thus we start from the discrete results and relax them into the continuous domain. Intuitively, the asymptotic behavior of the discrete problem should approach the continuous problem when the number of nodes is infinite. However, in our problem, the number is limited. Thus discrete results from the SPP provide a feasible, sub-optimal solution to the continuous problem.

Our objective is to re-locate SNs in the continuous domain so the total cost which consists of the intra-cluster routing cost and the deploying cost is minimal. We find that, in most cases, variations of geographical solar radiation are quite small in a cluster (unless some spots are covered by shades) so the total cost is usually dominated by the routing cost. Changing SNs to new locations might reduce the routing cost (by ΔC) but lead to an increase of deploying cost (by ΔF). If the reduction in routing cost is higher than the increment in deploying cost ($\Delta C - \Delta F > 0$), there is still an extra saving in the total cost. To minimize the total cost, a naive approach is to divide the field into grids and enumerate through all possible locations. This method obviously requires enormous computation power and its accuracy also depends on the density of the grid.

We shift our focus to minimizing intra-cluster routing cost, and at the same time, we look for locations that offer the largest overall savings $\Delta C - \Delta F$. From SPP, a set $\mathcal{S}$ is obtained ($\mathcal{S} \subseteq \mathcal{N}$) with the corresponding cluster set $\mathcal{B}_i$, $\forall i \in \mathcal{S}$. We observe that the problem resembles the well-known *Weber problem*[101, 102] which finds the geometric mean for the set of $\mathcal{B}_i$ nodes. A well-known algorithm to solve the problem is to use an iterative procedure due to Weiszfeld[101]. In order to consider the deploying cost also, we introduce an additional step to the original Weiszfeld algorithm.

The algorithm is illustrated below. First, we initialize SN's location x_0 at node i's coordinates (from the output of SPP). The Weiszfeld algorithm uses a recursive

Table 5.5: Extended Weiszfeld algorithm for a cluster

Initialize SN's location x_0 to i's coordinates obtained by SPP.
While $x_{k+1} - x_k > \varepsilon$
$W(x_k) = \big[\sum_{j=1}^{\lvert\mathcal{B}_i\rvert} \frac{\alpha_j}{c(x_k,j)}\big] / \big[\sum_{j=1}^{\lvert\mathcal{B}_i\rvert} \frac{1}{c(x_k,j)}\big]$, $x_{k+1} = W(x_k)$.
Record deploying cost F_k at x_k.
End While
$n \leftarrow k$ is the number of iterations until convergence.
Find $k_m = \arg\min_{1 \le i \le n} (C_i + F_i)$ and place SN at location x_{k_m}.

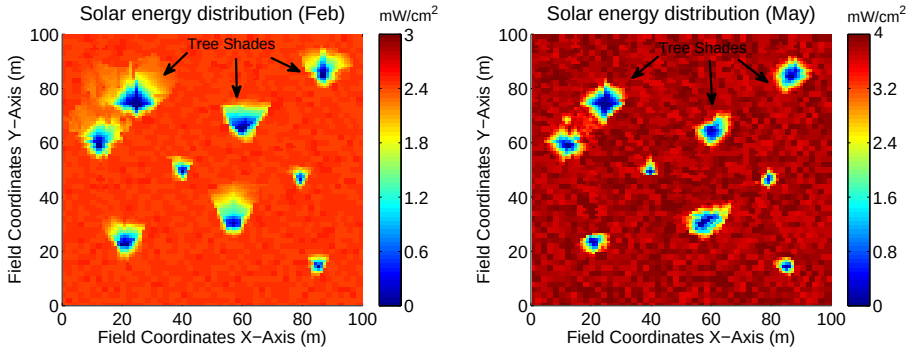
function $W(\cdot)$ to find the optimal cost. The computation of the k-th iteration is,

$$W(x_k) = \big[\sum_{j=1}^{|\mathcal{B}_i|} \frac{\alpha_j}{c(x_k, j)}\big] / \big[\sum_{j=1}^{|\mathcal{B}_i|} \frac{1}{c(x_k, j)}\big] \tag{5.16}$$

where $c(x_k, j)$ is the routing cost of transmitting a packet from location x_k to node j, and α_j are the coordinates of node $j \in \mathcal{B}_i$. The iteration continues by executing $x_{k+1} = W(x_k)$ until location changes are less than a small error bound ε. Since it has been proved in [102] that the Weiszfeld algorithm can converge to an optimum, the intra-cluster routing cost $C_k = \sum_{j=1}^{|\mathcal{B}_i|} \frac{1}{c(x_k,j)}$ for the k-th iteration is larger than C_{k+1} in the $(k+1)$-th iteration. However, the corresponding deploying cost may not share the same property due to the relative randomness in geographical solar energy distribution. For an algorithm that converges in n iterations, an additional step of finding the minimal deploying cost $k_m = \arg\min_{1 \le i \le n} (C_i + F_i)$ is needed when the iteration is over. Then location x_{k_m} is the final solution of the extended Weiszfeld algorithm. The algorithm is summarized in Table 5.5.

5.4.3 Adapt to Solar Variations

Here, we demonstrate how to change SNs' locations to adapt to solar variations. During different seasons of a year, the sun's angle towards earth surface varies slowly. Consequently, the harvested energy at different locations reflects such changes due to building obstructions, tree shades, etc. Fig. 5.2 shows the heatmaps of a sensing field on our campus gathered at different locations in February and May



(a) (b)

Figure 5.2: Geographic solar energy distribution in different months. (a) February. (b) May.

(Longitude at North $40°$). We can see that the areas falling into tree shades are quite different. This has a direct impact on the deploying cost f_i at a location. Second, the strength of solar radiation also varies dramatically. In February, the maximum level is 3 mW/cm^2 and an increase of 13% is observed in May. These observations suggest that SN locations should be re-calculated after some time T_c. Otherwise, they might be covered in shades with limited harvesting capabilities.

Our algorithm fully exploits the distributed nature of WSNs. During operation, each node records solar strength at its location periodically and maintains a trailing average for the past T_c time. f_i is updated every T_c time accordingly. Once a new deployment is initiated, SNs' locations are calculated using the distributed algorithm. After their locations are found, an MC is dispatched to re-locate corresponding SNs to designated locations.

5.5 Wireless-powered Sensor Level: Maintaining Energy Balance

In this section, we study the wireless-powered sensor level. Our main objective is to maintain network energy balance on WNs in different scenarios. First, we derive energy balance when SNs operate during sunny days. To facilitate our analysis, we denote the number of SNs obtained by the SPP algorithm as $s = |\mathcal{S}|$ and the maximum hop count from WN to its assigned SN as h. We assume that a total

budget B for s SNs and m MCs, $sp_s + mp_m \leq B$. We explore the relationship between s and m given their manufacturing costs p_s and p_m ($p_s < p_m$). Second, during cloudy/raining days, energy balance might be broken. In this case, we further study how to regain such balance by refilling the energy gap. We propose to utilize several WNs to act as temporary cluster heads for aggregating data. A numerical range of WN cluster heads is first derived followed by a distributed algorithm to determine which WNs should be selected.

5.5.1 Energy Balance

First, let us consider energy consumptions in the network. For s shortest path routing trees rooted at SNs, the total energy consumption is

$$\begin{aligned} E_c &= \sum_{j\in\mathcal{N}} \big[\lambda(e_t + e_s) + \sum_{i\in\mathcal{C}_j} \lambda(e_t + e_r)\big]T \\ &\leq \sum_{i=1}^{h} \big[N_i(e_t + e_s) + \sum_{\substack{j=i+1,\\ i\neq h}}^{h} N_j(e_t + e_r)\big]\lambda sT \\ &= \big[(\frac{2}{3}h^3 - \frac{1}{2}h^2 - \frac{1}{6}h)(e_t + e_r) + h^2(e_t + e_s)\big]\pi r^2\rho\lambda sT \end{aligned} \tag{5.17}$$

where $\mathcal{C}_j$ is the set of child nodes of $j \in \mathcal{N}$, $N_i = (2i-1)\pi r^2\rho$. The inequality holds because 1) a cluster can be estimated as a circle of radius $R = hr$ which consists of h concentric rings [14]; 2) summation of consumptions from all circle-shaped clusters has overlapping areas between neighboring clusters.

The harvested solar energy can be estimated by the empirical model proposed in [103]. The model provides a year-round analysis of solar radiations from weather stations and relates power levels to a quadratic equation on the time t of the day,

$$E = (a_1(t + a_2)^2 + a_3)(1 - \sigma). \tag{5.18}$$

The shape of Eq. (5.18) is determined by parameters $a_1 - a_3$ that vary seasonally for different months. For example, for the month of May, $a_1 = -1.1$, $a_2 = -13.5$ and $a_3 = 43.5$. t_1 and t_2 are the respective time of sunrise and sunset ($t_1 = -\sqrt{-\frac{a_3}{a_1}} - a_2, t_2 = \sqrt{-\frac{a_3}{a_1}} - a_2$). σ is the percentage of cloud cover from

weather reports. For T days, energy harvested by SNs is

$$E_s = s \sum_{i=1}^{T} \int_{t_1}^{t_2} \big[a_1(t+a_2)^2 + a_3\big](1-\sigma_i)dt \tag{5.19}$$

The wireless energy replenished by MCs into the network is governed by the battery charging rates C_h/T_r. Thus, the amount of wireless energy replenished by m MCs in T can be calculated by $E_w = (mTC_h)/T_r$, $m \neq 0$. Then network energy balance is achieved when

$$\begin{aligned} &E_c \leq E_s + E_w \\ &E_c < \big[\big(\frac{2}{3}h^3 - \frac{1}{2}h^2 - \frac{1}{6}h\big)(e_t + e_r) + h^2(e_t + e_s)\big]\pi r^2 \rho\lambda sT \\ &\leq s \sum_{i=1}^{T} \int_{t_1}^{t_2} \big[a_1(t+a_2)^2 + a_3)\big](1-\sigma_i)dt + \frac{mTC_h}{T_r} \end{aligned} \tag{5.20}$$

Since $\pi(hr)^2 s \geq L^2$, we have $\sqrt{L^2/(sr^2\pi)} \leq h$. By plugging it into Eq. (5.20) and taking approximation $e_t \approx e_r$, we obtain a relationship between s and m

$$\frac{LT_r e_t \rho\lambda}{3\sqrt{\pi}C_h r}\big(\frac{4L^2}{\sqrt{s}} - \pi r^2\sqrt{s}\big) - Xs + \big(\frac{1}{2}e_t + e_s\big)\frac{L^2\rho\lambda T_r}{C_h} \leq m, \tag{5.21}$$

where X is

$$X = \big[\frac{T_r}{C_h T}\sum_{i=1}^{T}(1-\sigma_i)\big]\big[\frac{a_1}{3}t^3 + a_1 a_2 t^2 + (a_1 a_2^2 + a_3)t\big]\big|_{t=t_1}^{t=t_2}. \tag{5.22}$$

5.5.2 Analysis of Budget and System Cost

The relationship between s and m in Eq. (5.21) can be explained graphically. Fig. 5.3(a) shows a group of energy balance curves when $N = 250 \sim 750$. Any point on a curve serves the same purpose for balancing network energy and there is no preference between choosing SNs or MCs as long as the balance holds. In fact, these curves can also be interpreted as the *indifference curves* in microeconomics. An indifference curve shows a collection of different goods between which the consumer is indifferent and every point on the curve results in the same utility. For example, when $N = 500$, point A requires 2 SNs and 8 MCs, which is equivalent

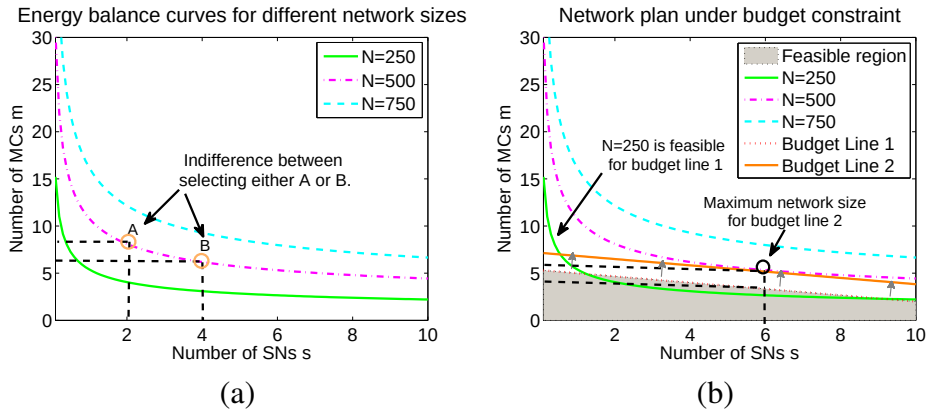


Figure 5.3: Network plans under budget constraints. (a) Energy balance curves for different network sizes. (b) Optimal choices under budget constraints.

to point B of 4 SNs and 6 MCs. Note that keeping adding SNs reduces MCs at a diminishing marginal rate. This is because that when the number of SNs is small, adding more SNs helps alleviate routing cost significantly (saving energy); however, this benefit gradually diminishes as more SNs are deployed.

Based on the budget, we can find whether a network plan is feasible to maintain energy balance. Since s is determined by the SPP algorithm, the corresponding number of MCs m can be found from the budget line $m = -\frac{P_m}{P_s}s + \frac{B}{P_s}$ (point (s, m)). If this point is above a balance curve, it means that the corresponding network size can satisfy energy balance. For example, point $(6, 3)$ on budget line 1: $m = -\frac{1}{3}s + 5$ is above the balance curve of $N = 250$ which indicates that the selection of $s = 6$, $m = 3$ is feasible. The feasible region for budget line 1 is marked as the shaded area. Furthermore, to find the maximum network size a given budget can sustain, we gradually increase N until point (s, m) is no longer above the balance curve. For point $(6, 5)$ on budget line 2, the maximum network size is $N = 500$. As the network size increases, the budget should be increased as well, which is to shift the budget line upward in Fig. 5.3(b). In this way, network administrators can quickly find an appropriate network plan, given the budget and available choices of SNs and MCs.

5.5.3 Adaptive Re-selection of Cluster Heads

An imperfection of solar energy is that sunlight is not always available. For example, during raining seasons, the network could experience consecutive cloudy or raining days and SNs are unable to harvest enough energy. To sustain network operation, our framework should adaptively switch cluster heads to WNs for aggregating sensed data. In this section, we first discuss how to maintain energy balance in the absence of solar energy. Then, we propose an algorithm to re-select cluster heads among WNs.

Maintaining Energy Balance

Since the number of MCs m is fixed for a network plan, we cannot expect more energy income from the energy replenishing side. To this end, we should reduce energy consumptions to restore energy balance. That is, the energy gap during cloudy days should be filled by reducing consumption for at least the same amount. Intuitively, introducing more cluster heads can effectively reduce energy consumption because more aggregation points would shorten packet relay paths. In other words, this is equivalent to having smaller k-hop clusters ($k < h$). The following property validates our intuition.

Property 4: For a network originally clustered by s SNs with cluster size $h > 2$, in the case that solar energy is unavailable, we can always restore network energy balance by reducing cluster size.

Proof. By assumption, for SNs to successfully aggregate and transmit data, $E_s^* \geq sh^2r^2\pi\rho\lambda(e_t + e_r)T$. Plugging this into Eq. (5.20), we have

$$\frac{mC_h}{T_r} > \big[\big(\frac{2}{3}h^3 - \frac{3}{2}h^2 - \frac{1}{6}h\big)(e_t + e_r) + h^2(e_t + e_s)\big]\pi r^2\rho\lambda s \tag{5.23}$$

For $h > 2$, $\frac{2}{3}h^3 - \frac{3}{2}h^2 - \frac{1}{6}h > 0$ so $\frac{mC_h}{T_r} > h^2r^2\pi\rho(e_t + e_s)\lambda s > N(e_t + e_s)\lambda$. $N(e_t+e_s)\lambda$ is exactly the energy consumed by sensors to generate and transmit data in one-hop communication to the MCs and the inequality states that the recharging rates from MCs is enough to support one-hop communications. Thus, we have proved that for $h > 2$, in the worst case, we can always use one-hop mobile data gathering to restore energy balance. □

Remarks: An anomaly is $h = 1, 2$. Because the original cluster sizes are already very small, we cannot reduce energy consumption further by having smaller clusters. In these cases, a notification message should be sent to request for more MCs.

However, one-hop mobile data gathering only occurs in the worst case. Normally, we have $1 < k < h$ so our objective is to calculate how many WN cluster heads are needed given k. From the previous subsection, the solar radiation model indicates the energy harvested peaks when $\sigma \approx 0$ (perfect weather condition). Let us denote the number of new WN cluster heads by s' ($s' > s$). The maximum amount of energy harvested is E_s^* when $\sigma = 0$ in the ideal case. $E_c(h)$ is the energy consumption with h-hop clusters. $X_c(k)$ is the energy consumption for each k-hop cluster (plugging k into Eq. (5.17) and getting rid of s'). Since we require $E_s^* \leq E_c(h) - s'X_c(k)$, a range for s' is

$$\frac{L^2}{\pi(kr)^2} < s' \leq \frac{E_c(h) - E_s^*}{X_c(k)}. \tag{5.24}$$

By fixing k, any s' satisfying Eq. (5.24) will guarantee energy balance of the network. Next, we develop a distributed algorithm to find WN cluster heads.

Head Re-selection Problem

We now further explore the *Head Re-selection Problem* (HRP) which finds k-hop clusters with s' cluster heads satisfying Eq. (5.24). On one hand, since WN cluster heads will be traversed by MCs for data collection, the number of such nodes should be minimized to save MCs' moving cost. On the other hand, for heads to cover all the nodes within k hops, s' should be sufficiently large; otherwise, clusters will exceed k hops and more likely break the energy balance. Hence, our objective is to select a minimum number of heads and ensure that the shortest path from any node to its nearest head does not exceed k hops. It is not difficult to see that HRP is the *minimum k-hop dominating set problem* which is NP-hard [104].

A distributed algorithm for this problem is proposed in [104] for ad-hoc networks. The algorithm requires two rounds of k-hop message flooding for all the nodes. Since flooding is usually less preferred in energy constrained WSNs, we will not adopt the algorithm in [104]. Instead, we leverage the range in Eq. (5.24)

as a basis for HRP. That is, as s' grows, hop distance from a node to its nearest head should decrease. Thus, we can start from the lower bound and increase s' iteratively until all the nodes are covered in k hops or the upper bound is reached. To find which WNs should become cluster heads, we extend the *furthest first traversal* algorithm proposed in [105]. The algorithm selects the node with the maximum distance from the current node to become the head in the next round. Unfortunately, the algorithm cannot be applied directly to our problem because: 1) it is centralized and not efficient to implement in distributed WSNs; 2) it may lead to inefficient selections. A new head might be chosen in the vicinity of an established one thereby causing a large overlap between neighboring clusters. This is not efficient and may also violate Eq. (5.24). Hence, we leverage the principle of *furthest first traversal* and propose a new distributed algorithm.

When gathered data at an MC indicates solar energy is not sufficient to support SNs, the MC sends a *head notification* message to any arbitrary WN whose battery has just been replenished and sets a counter to 0. The message specifies the cluster size k hops ($k = h - 1$ initially and decreased by 1 in each trial till $k = 1$). Upon receiving the head notification message, the WN declares itself as a new head and builds a shortest path tree (e.g., using Bellman-Ford algorithm). Each node also maintains a routing entry to store minimum hop distance to a head. Those entries are updated when a new shortest path tree is formed. If a node j's entry indicates the minimum hop distance to a head i is less than or equal to k, it sends a *join* message to i to "join" the cluster as a member. Otherwise, it sends a *resume* message to node i to let the head selection continue. Within a timeout period, if the head receives a resume message, it means that there still exist some node(s) uncovered and the selection process should continue.

If a resume message is received, the head computes a shortest path tree using the Bellman-Ford algorithm. To avoid inefficient head selection, nodes should also report to the head whether they are cluster members or not. Then a new *head notification* message is generated and sent along the shortest path tree to the node with the maximum hop distance and enough battery energy which is not a cluster member yet. The counter is then increased by one. Otherwise, if no resume message is received during the timeout period, the head declares that clustering is successful by sending a *complete* message to all the heads. Upon receiving the complete message, heads report to the MC of cluster information. Note that if the counter exceeds the

Table 5.6: Distributed Head Re-selection Algorithm for WN i, $i \in \mathcal{N}$

MC sends *HeadMsg* to a WN (with enough energy), counter $c \leftarrow 0$, sets HeadMsg.hop to $k (k < h)$. Set of cluster heads $\mathcal{H} \leftarrow \emptyset$.
If Recv(HeadMsg.ID $= i$ AND $c \leq \frac{E_c(h)-E_s^*}{X_c(k)}$)
$d_{ij} = \min_{j \in \mathcal{N}}$ HopCount(i, j) (Bellman-Ford-SPT(i)),$\mathcal{H} \leftarrow \mathcal{H} + i$.
Send *new routing msg* regarding new head i to all the nodes.
Set time-out period T waiting for *resume* messages.
If Recv(ResumeMsg.ID $= i$) within T
$u = \arg\max \min_{j \in \mathcal{N}}$ HopCount(i, j), $c \leftarrow c + 1$.
Send *HeadMsg* to u.
Else
Clustering is completed and broadcast *complete msg*.
End If
Else If Recv(NewRoutingMsg.ID is i)AND $\min_{j \in \mathcal{H}}$HopCount$(i, j) > k$.
Send ResumeMsg to the new head.
Else If Recv(NewRoutingMsg.ID is i)AND $\min_{j \in \mathcal{H}}$HopCount$(i, j) \leq k$.
Send JoinMsg to $u = \arg\min_{j \in \mathcal{H}}$ HopCount(i, j),
Declare as cluster member of u ($\mathcal{B}_u \leftarrow \mathcal{B}_u + i, \mathcal{N} \leftarrow \mathcal{N} - i$).
Else If Recv(HeadMsg.ID $= i$ AND $c > \frac{E_c-E_s^*}{X_c(k)}$)
$k \leftarrow k - 1$, broadcast a *restart* message.
Else Forward message according to routing entries.
End If

upper bound in Eq. (5.24), the current k is not feasible to maintain energy balance so it should be further decreased. In this case, the head should broadcast a message to restart the whole process and choose a smaller k. The psuedocode of this algorithm is given in Table 5.6. Based on *Property 4*, the distributed HRP algorithm can always find a set of cluster heads in $\mathcal{O}(hS)$ rounds and the worse case message overhead is $\mathcal{O}(hSN^2)$, where S is the upper bound in Eq. (5.24).

5.6 Mobile Charger Layer: Joint Wireless Charging and Mobile Data Gathering

In this section, we focus on optimizing trajectories of MCs. Proposed in [41], the instrumenting radio modules on MCs realize joint wireless charging and mobile data gathering on a single MC. This design certainly reduces manufacturing cost of MCs and system cost. However, because the effective wireless charging range is very limited (0.5-1m), the method in [41] requires the MC to stop at the exact WN location to perform simultaneous data gathering and recharge. However, in our framework, since SNs are powered by solar energy, it is only necessary for the MC to enter the transmission range ("touch" the transmission boundaries) to collect data from SNs. This creates opportunities to further optimize MCs' trajectories.

5.6.1 Planning of Joint Data Gathering and Recharging Tours

Initial Center Tour

We assume MC i has been assigned a touring sequence. The sequence defines an ordered set of nodes that starts from the base station b, traverses through WNs w_i and SNs (cluster heads) a_j, $w_i \in \mathcal{N}, a_j \in \mathcal{S}$, and finally returns to the base station for uploading data and recharging MC's own battery. Recharge scheduling algorithm proposed in [71] can be used conveniently to take recharging and data gathering requests together and calculate an initial touring sequence for each MC. Normally, the cluster size is larger than one hop ($h > 1$), and the transmission range around SNs form disjoint disks with identical radius. Since the initial sequence does not distinguish an SN from a WN and stops at the center of SN's transmission radius, we call it "Initial Center Tour" and denote its length as L_c. For such a tour with n WNs and s SNs, we have the following property.

Property 5: For an optimal tour with length L_r^*, when L_r^* is much larger than transmission range r, L_c is within $\left(1 + \frac{8}{\pi} + \epsilon\right) \approx 3.55 + \epsilon$ to the optimal L_r^*.

Proof. Our proof is based on [118]. Since SNs can be represented by disjoint disks, the sum of feasible areas for s SNs is $s\pi r^2$. We consider a larger disk of radius $2r$ so any point in disk of radius r can be enclosed. The total area $s\pi r^2$ should be less

than the area swept by the disk of $2r$,

$$s\pi r^2 + n\pi r_w^2 \leq 4rL_r^* + 4r^2\pi. \tag{5.25}$$

The wireless charging range r_w is much smaller than r ($r_w \ll r$). If we enforce the MC to go through disk centers, an extra distance less than $2r$ has to be made (entering and leaving the center). Thus, L_c is bounded by

$$\begin{aligned} L_c &\leq L_r^* + 2rs \leq L_r^* + 2r\frac{4rL_r^* + 4r^2\pi - n\pi r_w^2}{\pi r^2} \\ \frac{L_c}{L_r^*} &\leq \left(1 + \frac{8}{\pi}\right) + \frac{8r}{L_r^*} \leq 1 + \frac{8}{\pi} + \epsilon \end{aligned} \tag{5.26}$$

In the first step, we use Eq. (5.25) for s. In the last step, we omit the last term $n(\frac{r_w}{r})^2$, as $\frac{r_w}{r} \approx 0$. Since r is much smaller than L_r^*, we denote $\frac{8r}{L_r^*} \leq \epsilon$ where ϵ is a small fraction close to 0. □

Exhaustive Search

Although the initial center tour guarantees a $(3.55 + \epsilon)$-factor approximation, the solution can be further improved if the MC can take shortcuts through the disks. This problem is known as the *Traveling Salesmen Problem with Neighborhoods* and no efficient solution exists [117, 118]. However, in our problem, we can take advantage of WNs in the sequence and greatly reduce computation complexity. For all WNs in a sequence (and the base station), we order them in pairs (b, w_1), (w_1, w_2), ..., (w_n, b). For each pair (w_i, w_{i+1}), there could be at most s SNs in between. Let us start the analysis with $s = 1$. Fig. 5.4 shows that there are two cases: 1) the path connecting w_i and w_{i+1} directly cuts through the disk (Fig. 5.4(a)). In this case, the MC does not need to change directions. It only stops for a period of data uploading time (several minutes) in the disk. 2) the disk does not intersect with the path so there should exist a point on the boundaries of the disk that can minimize the path ($\min(a+b)$ in Fig. 5.4(b)). A naive approach is to divide the disk perimeter into l segments and find out which one yields the minimum distance. The method is used in [114] to find optimal hitting points on disk boundaries and its accuracy is proportional to l.

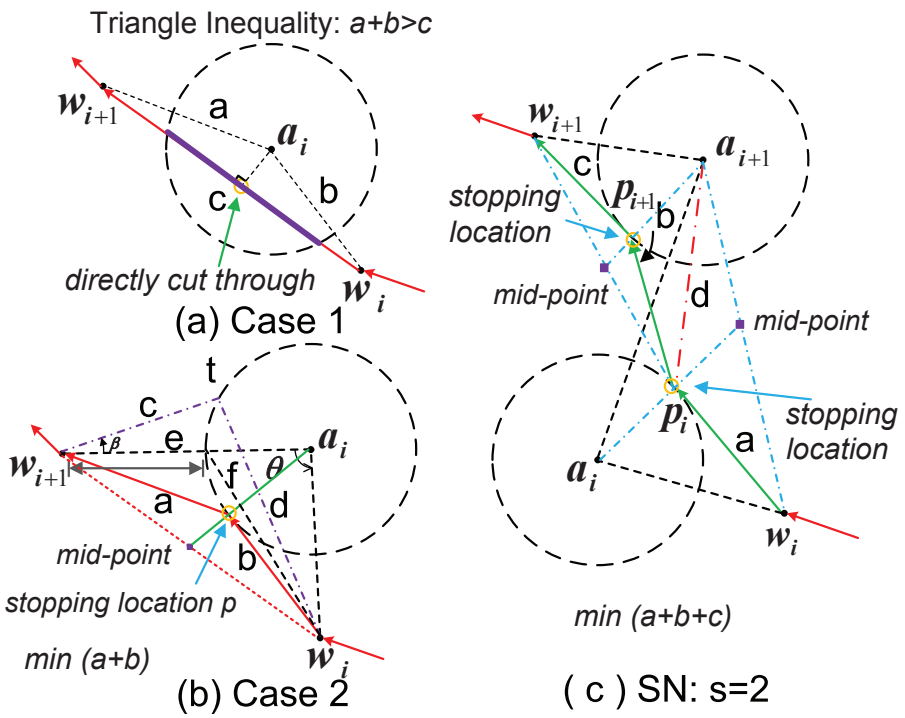


Figure 5.4: Analysis of shortest path through feasible regions around SNs.

A closer look at Fig. 5.4(b) suggests a possible reduction of search space. Let us define the angle between lines connecting $w_i a_i$ and $w_{i+1} a_i$ as θ. For a point t on the arc outside the sector, there is an angle $\beta > 0$ between lines $w_{i+1} t$ and $w_{i+1} a_i$. Clearly, $c + d > e + f$ so any point t outside the sector gives an inferior solution compared to a point within the sector of θ. Thus, we can narrow down the search space to the points on the arc within angle θ between $w_i a_i$ and $w_{i+1} a_i$, so examination of only a fraction $f = \frac{\theta l}{2\pi}$ points is enough. Nevertheless, computation complexity of exhaustive search still grows exponentially when there are s SNs between w_i and w_{i+1} (with complexity $\mathcal{O}(f^s)$), thus a faster method is needed.

Minimizing Sum of Squared Distance

Exhaustive search quickly turns out to be impractical in reality. If the problem can be solved analytically, computational complexity can be greatly reduced. Since the expressions of distance involving square roots tend to yield intractable computations, we minimize the sum of squared distance instead. In fact, sum of squared

distance has been used in many applications such as the well-known K-means algorithm [66]. The estimation error to the actual sum of distance will be evaluated by simulations. Likewise, we start our analysis from $s = 1$ and derive the following property.

Property 6: The point p on disk a_i that minimizes $d = (|w_i p|)^2 + (|w_{i+1} p|)^2$ is the intersection between line $w_m a_i$ and the disk, where w_m is the mid-point between the coordinates of w_i and w_{i+1}.

Proof. Although the property seems to be true by visual judgment of Fig. 5.4(b), a geometric proof is difficult. Thus, we calculate p in terms of cartesian coordinates. Denote coordinates of w_i,w_{i+1} and p as $(x_i, y_i), (x_{i+1}, y_{i+1}), (x, y)$, respectively. Assume the origin of coordinate system resides at the disk center. The function of the disk is $x^2 + y^2 = r^2$. We use the *Lagrangian multiplier* method to find minimal sum of squared distance. After taking partial derivatives, the variables are

$$\begin{aligned} \frac{L_x}{\partial x} &= 4x - 2(x_i + x_{i+1}) + 2x\lambda, \\ \frac{L_y}{\partial y} &= 4y - 2(y_i + y_{i+1}) + 2y\lambda \\ \frac{L_\lambda}{\partial \lambda} &= x^2 + y^2 - r^2 \end{aligned} \tag{5.27}$$

After some calculations, the coordinates for p are

$$\begin{aligned} x &= \frac{(x_i + x_{i+1})r}{\sqrt{(x_i + x_{i+1})^2 + (y_i + y_{i+1})^2}}, \\ y &= \frac{(y_i + y_{i+1})r}{\sqrt{(x_i + x_{i+1})^2 + (y_i + y_{i+1})^2}}. \end{aligned} \tag{5.28}$$

On the other hand, the coordinates of w_m are $(\frac{x_i + x_{i+1}}{2}, \frac{y_i + y_{i+1}}{2})$. We plug the function of line $w_m a_i$, $y = \frac{y_i + y_{i+1}}{x_i + x_{i+1}} x$ into the disk function of a_i, and obtain two intersection points

$$x = \pm\sqrt{\frac{r^2}{(\frac{y_i + y_{i+1}}{x_i + x_{i+1}})^2 + 1}}, y = \pm\sqrt{\frac{r^2}{(\frac{x_i + x_{i+1}}{y_i + y_{i+1}})^2 + 1}} \tag{5.29}$$

We can see one of the solutions in Eq. (5.29) is exactly Eq. (5.28), so the property

Table 5.7: Route Improvement Algorithm for MCs

Input: Sequence $\langle b, w_1, \ldots, a_i, \ldots, w_j, \ldots, w_n, b\rangle$.
Set of SNs between w_j and w_{j+1}, $\mathcal{S}_j$, $\mathcal{S} = \bigcup_{j\in\mathcal{N}} \mathcal{S}_j$.
Output: Coordinates (x_i, y_i) MC should visit near a_i.
While $\mathcal{S}_j \neq \emptyset$
For $a_i \in \mathcal{S}_j$, find coordinates of WNs w_j, w_{j+1} in sequence.
If a_{i+1} is also between w_j, w_{j+1}.$x_{j+1} = x_{a_{i+1}}$, $y_{j+1} = y_{a_{i+1}}$.
Else (x_{j+1}, y_{j+1}) is set to w_{j+1}'s coordinates.
End If
Establish cartesian coordinate system originated at center of a_i.
$x_i = (x_j + x_{j+1})r/\sqrt{(x_j + x_{j+1})^2 + (y_j + y_{j+1})^2}$,
$y_i = (y_j + y_{j+1})r/\sqrt{(x_j + x_{j+1})^2 + (y_j + y_{j+1})^2}$.
$\mathcal{S}_j \leftarrow \mathcal{S}_j - a_i$.
End While

is proved. □

Next, we consider the case of $s = 2$ without a direct cut as illustrated in Fig. 5.4(c). To use our method, computing each touching point on a disk needs two fixed points. For two disks, since the touching points can change simultaneously, minimization of sum of distance $(a + b + c)$ by considering multiple variables is very difficult analytically. Instead, we use the center of a_{i+1} as the reference point and calculate p_i on disk a_i to minimize $(a + d)$ first. Then, based on p_i, we calculate p_{i+1} on a_{i+1} to $\min(b + c)$. In this way, each computation only involves one variable. The method can be easily extended to the case when there are s SNs between w_i and w_{i+1}, so a total of $\mathcal{O}(s)$ computation is needed, which reduces the exponential-$\mathcal{O}(f^s)$ exhaustive search algorithm to linear time. We summarize the route improvement algorithm in Table 5.7.

5.6.2 Optimizing Recharging Time

We now optimize the allocation of MCs' recharge times for different WNs to avoid battery depletion. After the touring sequences have been found, the MCs need to traverse through WNs and the neighborhoods of SNs for wireless charging and data gathering. We assume the energy recharged into sensors' batteries is roughly proportional to the recharging time. In fact, most batteries exhibit such property

as the longer they are being recharged, the more energy will be stored (under the battery capacity). An ideal situation is to replenish all the nodes to full capacity and hopefully no sensor depletes its battery before the recharge begins[83]. However, due to the nontrivial recharge time, the MCs have to reside at the WN locations for some time (e.g., 30-78 mins). This may easily cause energy depletion of subsequent nodes in the sequence.

Linear Program Formulation

Here, we take a new approach that gives the MCs more flexibility so they can recharge more sensors in fixed time and sustain their battery energy at working levels. That is, instead of fully replenishing sensors' batteries[33, 36, 41, 71, 83, 106], we allow partial recharge. In case a node is bound to deplete its energy, the MC can expedite all recharge schedules before that node.

Let us denote the touring sequence found in Section 5.6.1 by $\langle 1, 2, \cdots, i, \cdots, n \rangle$. There are two types of sojourn time at sensor nodes. For WN i, the MC stays for a period of t_i for recharge; for SN j, the MC spends a fixed time τ_j to collect data packets. $\tau_j = p_j/B$, where p_j is the amount of data at SN j's buffer and b is the data rate. A special case is when the MC's trajectory directly cuts through SN's transmission range (Fig. 5.4(a)). If $\frac{p_j}{b} \leq \frac{d_c}{v}$, $\tau_j = \frac{d_c}{v}$ where d_c is the length of chord within SN's transmission range. It means that if the data transmission time is less than MC's traveling time inside SN's transmission range, the MC can collect all the data without stopping. We denote MC's traveling time between two consecutive nodes in the sequence by $\tau_{i,i+1} = d_{i,i+1}/v$. Each WN i has a residual lifetime L_i. Our objective is to maximize the sum of recharge time under a pre-determined packet delay T. At the same time, we should also guarantee that all the recharge requests are met before their lifetime expirations. It can be formulated as a Linear Programming (LP) problem,

$$\mathbf{P4}: \quad \max \sum_{i \in \mathcal{N}} t_i \tag{5.30}$$

Subject to

$$\sum_{i=1}^{j-1}(\tau_{i,i+1}+t_i)+\sum_{i<j,i\in\mathcal{S}}\tau_i \le L_j; 2\le j\le n \tag{5.31}$$

$$0<t_i\le T_r; 1\le i\le n \tag{5.32}$$

$$\sum_{i=1}^{n-1}\tau_{i,i+1}+\sum_{i=1}^{n}t_i+\sum_{i\in\mathcal{S}}\tau_i\le T \tag{5.33}$$

Constraint (5.31) in fact contains $(n-1)$ constraints and each one ensures that for a sensor, the sum of all the time spent during recharging, data gathering and traveling for all previous nodes in the sequence does not exceed its lifetime. Constraint (5.32) states that the maximum recharge time is to replenish the battery to full capacity and Constraints (5.33) imposes a delay bound T for the entire recharge sequence.

Recharge Time Assignment

Although the problem can be calculated by standard LP solvers (e.g., using the simplex method), their worst case performance may take exponential time[121]. Thus, in this subsection, we develop a new algorithm by exploiting the particular structure of the problem. Since $\tau_{i,i+1}$ and τ_i can be calculated (constants) once the recharge sequence has been determined, we can simplify Constraints (5.31) and (5.33) as, $\sum_{i=1}^{j-1} t_i < L'_j$, $\sum_{i=1}^{n} t_i < T'$, where

$$L'_j = L_j - \sum_{i=1}^{j-1}\tau_{i,i+1} - \sum_{i<j,i\in\mathcal{S}}\tau_i, T' = T - \sum_{i=1}^{n-1}\tau_{i,i+1} - \sum_{i\in\mathcal{S}}\tau_i.$$

In other words, L'_j and T' represent the maximum recharge time from node 1 to $j-1$ and the entire sequence respectively. It is not difficult to observe that there are generally two cases for the optimal solution.

Case I: $\forall j \in \mathcal{N}$, $L'_j \ge (j-1)T_r$ and $T' \ge nT_r$. In this case, the MC can recharge all the nodes to full capacity, i.e., $t_i^* = T_r, \forall i \in \mathcal{N}$ and $\sum_{i\in\mathcal{N}} t_i^* = nT_r$.

Case II: $\exists j \in \mathcal{N}$, $L'_j < (j-1)T_r$ or $T' < nT_r$. In this case, recharge time prior to j may not take T_r so new assignments of recharge time should be performed. In

fact, the maximization should take the value of a node's lifetime when its lifetime constraint is tight. Based on this observation and the iterative structure of Constraint (5.31), we propose a time assignment algorithm. It assigns recharge time proportional to the energy demands. The algorithm starts from the node with the minimum lifetime $j_m = \arg\min (L'_j, T'), j \in \mathcal{N}$. Then for node i in the sequence ($1 \leq i < j_m$), an amount of $t_i = L'_{j_m} d_i / (\sum_{j=1}^{j_m-1} d_j)$ recharge time is assigned. d_i is the energy demand from node i. If assigned $t_i > T_r$, t_i should be bounded by T_r according to Constraint (5.32). The remaining time $t_i - T_r$ can be evenly distributed among the other nodes from $i+1$ to j_m. For the node's lifetime, we have the following property.

Property 6: For the time assignment of node i prior to j_m in the recharge sequence $i < j_m$, the constraint $\sum_{k=1}^{i-1} t_k < L'_i$ holds.

Proof. The property can be proved since

$$\sum_{k=1}^{i-1} t_k = L'_{j_m} (\sum_{k=1}^{i-1} d_k / \sum_{k=1}^{j_m-1} d_k) < L'_{j_m} < L'_i. \tag{5.34}$$

□

Based on *Property 6*, we can see that once j_m has been selected and recharge time is assigned for all the nodes before j_m, the lifetime constraints still hold for these nodes. Hence, we can proceed to find the next node with minimum lifetime from $(j_m + 1)$ to n and repeat the same procedure for recharge time assignments. The iteration continues until the recharge sequence is exhausted. In most cases, the time complexity of the algorithm is $\mathcal{O}(n^2)$ since it needs to iterate through $\mathcal{O}(n)$ nodes and each one requires $\mathcal{O}(n)$ time assignments. In the worst case, the algorithm needs $\mathcal{O}(n^3)$ since an extra $\mathcal{O}(n)$ assignments are needed once the calculated recharge time exceeds T_r. The algorithm is summarized in Table 5.8.

5.6.3 An Example of Proposed Hybrid Framework

Finally, we demonstrate a complete example of the proposed framework in Fig. 5.5. In Fig. 5.5(a), 8 SNs are placed to organize 250 WNs into clusters. Their initial locations calculated by the distributed SPP algorithm are marked by triangles and

Table 5.8: Recharge Time Assignment Algorithm

Initialize $j_s \leftarrow 1$, $j_m = \arg\min(\mathcal{L}', T')$
While $j_m \neq n$
Initialize time variables $\varphi_i \leftarrow 0$ $(j_s \leq i \leq j_m - 1)$.
For i from j_s to $j_m - 1$
$t_i = \min(T_r, L'_{j_m} d_i / \sum_{j=j_s}^{j_m-1} d_j + \varphi_i)$.
If $t_i = T_r$, for $i + 1 \leq k \leq j_m - 1$
$\varphi_k \leftarrow \varphi_k + (L'_{j_m} d_i / \sum_{i=j_s}^{j_m-1} d_i - T_r)/(j_m - i - 1)$
End
End For
$j_s \leftarrow j_m$, update $\mathcal{L}' \leftarrow \{L'_{j_m+1}, \cdots, L'_n, T'\}$
Find the next, $j_m = \arg\min(\mathcal{L}', T')$
End While

improved by the intra-cluster Weiszfeld algorithm shown as the crosses. In case of shortage of sunlight, Fig. 5.5(b) shows the results from the HRP algorithm to re-allocate cluster heads to WNs. Here, the loss in energy harvested from the 8 SNs can be compensated by introducing 13 WN cluster heads to reduce hop distance. For wireless charging and data gathering, Fig. 5.5(c) shows an initial center tour that covers 9 energy requests and 7 data uploading sites (SNs). The route is improved by our algorithm in Fig. 5.5 (d) with a saving of 17% moving energy on the MCs.

5.7 Performance Evaluations

In this section, we evaluate the performance of the framework by a discrete-event simulator and compare it with a network solely relied on wireless energy [41, 71, 108]. In the simulation, all the cluster heads are replenished by the MCs in the *wireless-powered framework*. $N = 500$ nodes are uniformly randomly distributed over a square field of $L = 150$ m. Sensors have identical transmission range of $r = 12$ m, and consume $e_s = 0.05J$ for generating a sensing packet and $e_t = e_r = 0.02J$ for transmitting/receiving a packet. Each time slot is 1 min and the traffic follows a Poisson distribution with average $\lambda = 3$ pkt/min. WNs have battery capacity $C_h = 750mAh$ and require $T_r = 78$ mins for recharge. SNs have larger

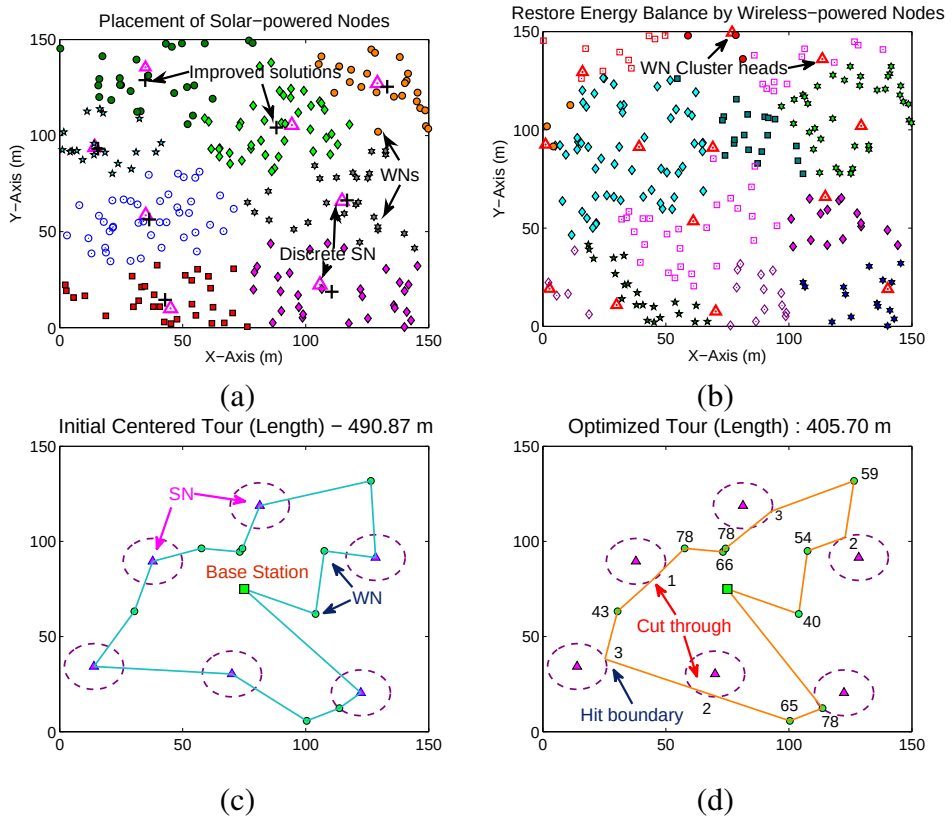


Figure 5.5: A complete example of the framework. (a) Placement of SNs. (b) Restoring energy balance by WNs. (c) Initial center tour. (d) Optimized joint tour with MC's stopping time.

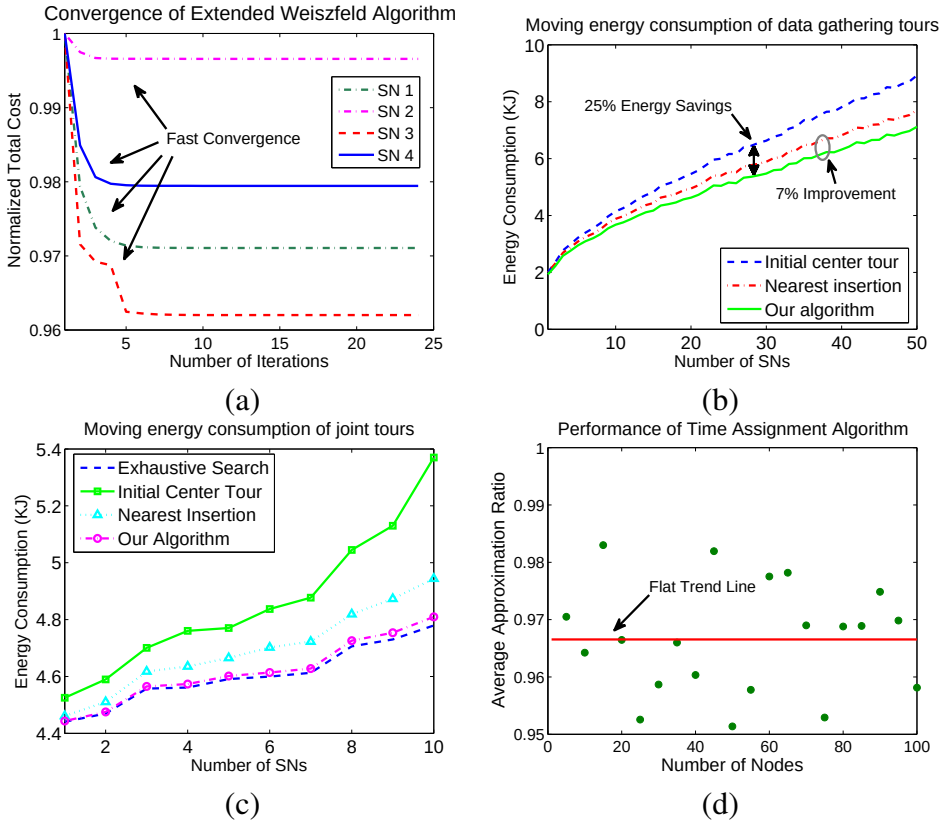


Figure 5.6: Evaluation of the proposed algorithms. (a) Convergence of Weiszfeld algorithm. (b) Improvements of tour for only data gathering sites (SNs). (c) Improvements of joint wireless charging and mobile data gathering tour. (d) Average approximation ratios of the recharge time assignment algorithm.

capacity of $2150mAh$. MC consumes 5 J/m while moving at $v = 1$ m/s. We use real meteorological trace at Stony Brook, NY from [119], which has a complete archive of weather conditions. The simulation time starts from December and lasts for 12 months.

5.7.1 Evaluation of Algorithms

First, we evaluate the performance of the proposed algorithms.

Extended Weiszfeld algorithm

We validate the convergence property of the extended Weiszfeld algorithm for finding the SN locations in continuous space. For clarity, we randomly pick 4 SNs and trace the evolution of their total costs in Fig. 5.6(a). We normalize the total cost for better visualization. First, we observe that the Weiszfeld algorithm can converge extremely fast (within 5-6 iterations) and offer an additional 2-5% energy savings. Second, since the deploying cost is usually much less than the routing cost in our problem, the total costs are dominated by the changes in the routing costs. Therefore, the optimal values reach the minimum when the algorithm converges. Otherwise, the algorithm will use the minimum total cost during the iteration process as the final solution (explained in Section 5.4.2).

Route Improvement Algorithm

We evaluate the route improvement algorithm by comparing it with the algorithms proposed in [41, 120]. The algorithm in [120] continuously finds the closest hitting points on the boundaries of the disks and we call it *nearest insertion algorithm*. The tour passing through the disk centers is used in [41] for joint wireless charging and data gathering and we denote it by *initial center tour*. Fig. 5.6(b) compares MC's moving energy using the three methods. First, we can see that our algorithm provides an average of 25% energy saving compared to the initial center tour. In fact, more energy saving can be achieved with a larger transmission range since an MC only needs to visit the transmission boundaries for gathering data. Second, the results further indicate 5-7% improvements over the nearest insertion algorithm [120]. This is because that selecting the closest hitting point on a disk cannot guarantee that the sum of distance to the neighboring nodes is minimal. In contrast, our algorithm finds a point on the disk that minimizes the sum of squared distance. To examine the gap between minimizing the sum of squared distance and the actual distance, we conduct more evaluations in Fig. 5.6(c) by considering a joint route comprised of WNs and SNs. Since WNs outnumber SNs by a considerable amount, we maintain a 10 to 1 ratio between WNs and SNs. To provide a baseline, an exact solution is found by exhaustive search using [114]. Surprisingly, our algorithm has only an average of 1% difference to the exact solution whereas reducing computation complexity from exponential to linear time. In addition, with mixed WNs and

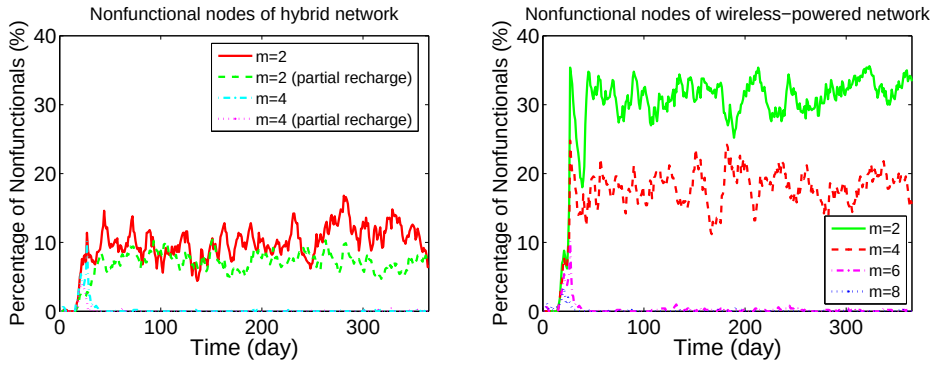
SNs, our algorithm also outperforms the nearest insertion algorithm by 5-10%.

Recharge Time Assignment Algorithm

We also evaluate the performance of the recharge time assignment algorithm and compare with optimal solutions from standard LP solver. Recharge sequences from 5 to 100 nodes are evaluated with nodes' battery energy and lifetime randomly distributed. The results are averaged over 100 simulation runs. Fig. 5.6 (d) illustrates difference between our algorithm and the optimal solution. We observe that our algorithm is able to achieve results very close (within 5%) to the optimal LP solver. As we increase the number of nodes in the sequence, the approximation ratio does not degrade as indicated by the flat trend line. In the simulation, we discover that in most cases, our algorithm can find the optimal solution. However, in some special cases, there are several consecutive nodes with limited lifetimes that require partial recharge of previous nodes. For these cases, our approach may not always yield optimal results but remains very close to them.

5.7.2 Nonfunctional Nodes

One of the key performance metrics for recharging is nonfunctional nodes. Once a node's battery is depleted, it stops working and becomes nonfunctional until its battery is replenished. To sustain perpetual operations, nodes should be alive all the time; otherwise, they will degrade sensing qualities and node communications. Fig. 5.7 compares the percentage of nonfunctional nodes between hybrid (full or partial recharge) and wireless-powered frameworks [41, 71, 108]. The SPP algorithm generates $s = 11$ SNs. To compare the performance, we change the number of MCs m. Fig. 5.7(a) shows the results from the hybrid framework when $m = 2 \sim 4$. We can see that 2 MCs can keep the percentage of nonfunctional nodes around 10% and 4 MCs can almost achieve perpetual operations. Partial recharge offers even better performance with less than 10% nonfunctional nodes when $m = 2$. In contrast, $m = 2$ for wireless-powered network results in 30% nonfunctional nodes in Fig. 5.7(b) and an increase to $m = 6$ still barely eliminates all battery depletions at equilibrium. These observations clearly demonstrate that the hybrid framework can improve network performance significantly. Since for a wireless-powered network, cluster heads consume energy much faster, MCs need to visit them more frequently,



(a) (b)

Figure 5.7: Number of nonfunctional nodes. (a) Hybrid framework. (b) Wireless-powered framework.

which reduces the chances for other nodes to get recharged. However, SNs are replenished by solar energy which has much higher power density so MCs have more leverage to take care of the rest of the network.

5.7.3 System Cost

The cost to maintain network operations is another important performance metric. In our framework, there are two types of costs. The first type comes from network maintenance which basically involves energy expenditures at the MCs while moving and recharging. Since the energy expense for recharging is necessary for sensor nodes, we focus on the energy cost while the MCs are moving. Fig. 5.8(a) and (b) trace the evolution of MCs' moving energy cost when our goals are maintaining nonfunctional percentage under 15% and 1%, respectively. First, let us examine the cost brought by partial recharge. Although it is not obvious on Fig. 5.8(a), by taking their mean values, we are able to see that partial recharge results in slightly higher cost (3.57 KJ vs. 3.49 KJ). This is because that partial recharge allows MCs to recharge more nodes in a fixed time period (MCs move more frequently). Second, we compare the moving cost to wireless-powered networks. Since it requires more MCs to maintain energy balance, their moving costs inevitably increase which are almost doubled if evaluated by their mean values (mean 6.91 KJ vs. 3.49 KJ of the hybrid network). Similar results are observed in Fig. 5.8(b) when the objective is to maintain nonfunctional percentage below 1%. The difference is that

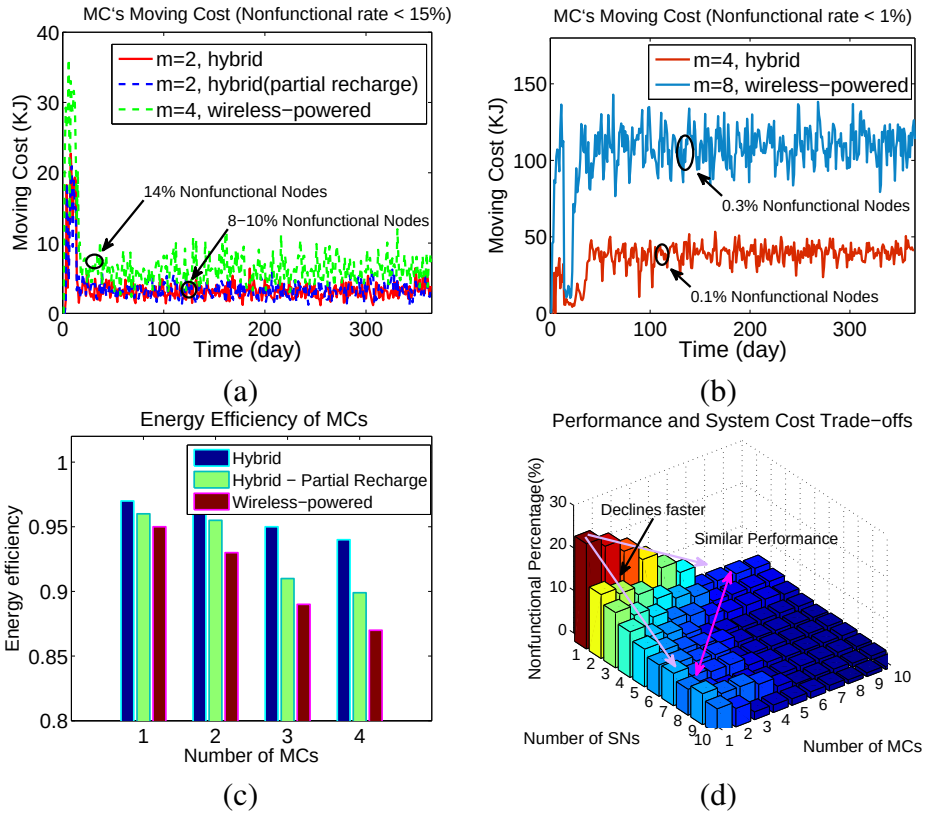


Figure 5.8: Evaluation of system cost. (a) Evolution of MC's moving cost when nonfunctional rate is less than 15%. (b) Evolution of MC's moving cost when nonfunctional rate is less than 1%. (c) Energy efficiency of MCs. (d) Trade-offs between performance and system cost.

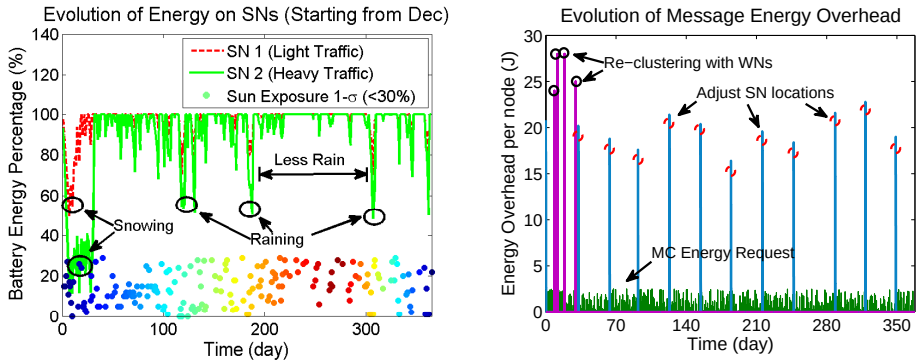
the gap between the two curves becomes even wider (mean value 36 KJ vs. 105 KJ), which indicates that the wireless-powered network almost needs 3 times the energy of the hybrid network. From these results, we can see that hybrid networks are more energy efficient for higher performance requirements.

We have also evaluated energy efficiency for different numbers of MCs in Fig. 5.8(c). Energy efficiency is defined as the ratio between energy replenished and the total energy consumed on the MCs. The hybrid network enjoys the highest overall energy efficiency as more than 90% energy can be used for recharge while partial recharge has slightly lower efficiency due to possibly more movements. The wireless-powered network has the lowest energy efficiency since the MCs have to move even more often to recharge cluster heads.

The second type of cost is the fixed cost of SNs and MCs. The theoretical results in Section 5.5 indicate that by having more SNs, we are able to save the cost on MCs since they are usually more expensive. To see the trade-offs between the numbers of SNs and MCs, we show the impact on nonfunctional percentage in Fig. 5.8(d) in which the X and Y axes represent the numbers of SNs and MCs and the Z axis represents the percentage of nonfunctional nodes. First, we observe that introducing SNs helps reduce the nonfunctional percentage much faster than MCs. Second, it is also interesting that with more SNs, even fewer MCs can achieve similar performance. For example, 8 SNs and 1-2 MC result in similar performance to 6-8 MCs and 1 SN. These results suggest that hybrid networks are more cost-effective and energy efficient than wireless-powered networks.

5.7.4 Harvested Energy and Message Overhead

To validate our algorithm design, we evaluate the evolution of SN's energy and network message overhead. Fig. 5.9(a) traces SNs' energy with weather conditions represented by percentage of solar exposure $(1 - \sigma)$ obtained in [119]. We focus on two typical nodes with light and heavy data traffic. We can see that through the month of December, energy storage continuously declines due to weak solar strength in winter. In addition, there are also several consecutive snowing days so SNs are unable to harvest enough energy and re-selection of cluster heads among WNs is needed. This gives SNs opportunities to recover their energy (during 40-50 days). For the remaining simulation, although a few consecutive raining days are



(a) (b)

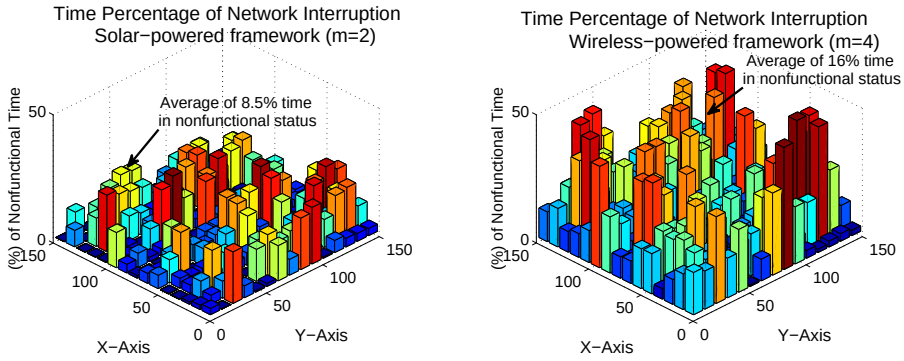
Figure 5.9: Evolution of harvested solar energy and message energy overhead. (a) Energy variations of SNs. (b) Message energy overhead.

observed, the energy gaps are quickly filled. This is because that solar radiation has strengthened since spring and energy storage is sufficient to sustain network operations. For example, from summer to fall, SNs can maintain their battery energy over 80% due to favorable weather conditions and strong solar radiation.

Fig. 5.9(b) demonstrates the energy consumed by exchanging control messages such as SN clustering, re-selection of WN cluster heads and energy information requests. For each month, MCs initiate a new calculation of SN's placement pattern to reflect the updated geographical solar radiations (e.g., Fig. 5.2). These message overhead is shown as the blue spikes at the beginning of each month. Note that in the simulation, when SNs' energy is less than 30% due to insufficient solar energy, they request to re-cluster by WNs. The overhead is represented by those higher spikes (in purple), which corresponds to the time when SNs' energy drops in Fig. 5.9(a). Our results indicate that re-clustering using WNs only occur during winter time though there are some consecutive raining periods in other seasons. From Fig. 5.9, we have validated that our algorithm can adapt to weather conditions effectively.

5.7.5 Geographical Distributions of Service Interruption

Finally, we examine geographical distributions of service interruptions. Our objective is to see how long nodes are in nonfunctional status and their geographical distributions. Since cluster heads are responsible for aggregating and uploading



(a) (b)

Figure 5.10: Geographical distributions of service interruption. (a) Hybrid framework $m = 2$. (b) Wireless-powered framework $m = 4$.

sensed data, their survivals are critical for the entire network. A breakdown may lead to severe packet loss, network interruption and extended data latency. For fair comparison, we use the results from Section 5.7.2 and set $m = 2$ for the hybrid framework and $m = 4$ for the wireless-powered framework so that both cases have a similar number of nonfunctional nodes. From Fig. 5.10(a), we observe that the distribution of nonfunctional nodes is quite even in the hybrid framework. In contrast, nodes around the cluster heads (including the heads as well) are more prone to deplete battery energy in the wireless-powered network (20% more nonfunctional time in Fig. 5.10(b)). On average, a node in the hybrid framework has only 8.5% time in nonfunctional status whereas it would experience 16% nonfunctional time in the wireless-powered network. The sharp contrast is because that for the wireless-powered network, MCs need to not only take care of cluster heads but also their surrounding areas. This may cause the MCs to move frequently between head locations and overwhelm their recharge capabilities. However, for the hybrid framework, MCs do not need to recharge SNs so the resources can be re-distributed among WNs to reduce their nonfunctional rates.

5.8 Conclusions

In this chapter, we consider a hybrid framework that combines the advantages of wireless charging and solar energy harvesting technologies. We study a three-level

network consisting of SNs, WNs and MCs levels. First, we study how to minimize the total cost of deploying a set of SNs. The problem is formulated into a facility location problem and a $1.61(1+\epsilon^2)$-factor distributed algorithm is proposed. The solution is further improved by using intra-cluster Weiszfeld algorithm in continuous space. Second, we examine the energy balance in the network and develop a distributed head re-selection algorithm to designate some WNs as cluster heads when solar energy is not available during raining/cloudy days. Third, we focus on how to optimize the joint tour consisting of both wireless charging and data gathering sites for the MCs. A linear-time algorithm is proposed that can approach very closely to the exact solution and reduce at least 5% MC's moving energy compared to previous solutions. We also propose to partially refill sensors' energy to further reduce battery depletion and develop an efficient algorithm to solve the problem with high accuracy. Finally, based on real weather data, we demonstrate through simulations the effectiveness and efficiency of the hybrid framework that can improve network performance significantly.

Chapter 6

Conclusions

This dissertation focuses on the applications of wireless charging technology in sensor networks. A suite of mathematical models and algorithms have been proposed to address critical issues in this emerging research field. In particular, first, a real-time energy information gathering protocol is proposed. Based on such information, on-line algorithms are proposed to handle both emergency and normal recharge operations. Second, a mathematical model is established based on energy neutrality of the network. The model provides a theoretical estimation for network plans. MC's moving cost and recharge capacity is further brought into consideration of the framework and an adaptive recharge scheduling algorithm is proposed. Third, we extend the single-hop wireless charging into multi-hops by embedding resonant repeaters on sensor nodes. This low-cost scheme provides extra network scalability and improves MC's recharge capability. A multi-hop recharge scheduling algorithm is proposed based on the physical calculations of charging efficiencies and cost trade-offs in the network are analyzed in a multi-objective optimization. Finally, a hybrid framework is investigated by combining solar and wireless-powered sensors. Both centralized and distributed location algorithms are developed for locating the solar-powered sensors. An effective scheme is proposed to re-gain energy balance when sunlight is unavailable. MC's routes are further optimized by considering a joint route while recharging and gathering data.

In sum, in this dissertation, we have conducted extensive and comprehensive studies around the fundamental problem of powering sensor nodes. We have proposed efficient algorithms to improve recharge scheduling, charging capability, net-

work latency while minimizing system cost. This work has laid the foundations for various kinds of sensing applications including the current Internet-of-Things by tackling the fundamental energy issues. As we can see, it engages in both industrial and academic research and would have a significant impact on principles and paradigms for the future of wireless sensor networks.

Bibliography

[1] I. Akyildiz, "Wireless sensor networks: a survey," *Computer networks*, vol. 38, no. 4, pp. 393-422, 2002.

[2] K. Sohrabi, "Protocols for self-organization of a wireless sensor network," *IEEE personal communications*, vol. 7, no. 5, pp. 16-27, 2000.

[3] W. A. Geoffrey, "Deploying a wireless sensor network on an active volcano," *IEEE Internet Computing*, vol. 10, no. 2, pp. 18-25, 2006.

[4] L. Yu, N. Wang, X. Meng, "Real-time forest fire detection with wireless sensor networks," *IEEE International Conference on Wireless Communications, Networking and Mobile Computing*, vol.2, pp.1214-1217, 2005.

[5] W. Ye, J. Heidemann, D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," *IEEE INFOCOM*, vol.3, pp. 1567-1576, 2002.

[6] A. Keshavarzian, H. Lee and L. Venkatraman, "Wakeup scheduling in wireless sensor networks," *ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, pp. 322-333, 2006.

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

[8] C. Ma and Y. Yang, "A battery-aware scheme for routing in wireless ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 8, pp. 3919-3932, Oct. 2011.

[9] C. Ma, Z. Zhang and Y. Yang, "Battery-aware scheduling in wireless mesh networks," *ACM/Springer Mobile Networks & Applications (MONET)*, vol. 13, pp. 228-241, 2008.

[10] C. Ma and Y. Yang, "Battery-aware routing for streaming data transmissions in wireless sensor networks," *ACM/Springer Mobile Networks & Applications (MONET)*, vol. 11, no. 5, pp. 757-767, October 2006.

[11] C. J. Hwan and L. Tassiulas, "Maximum lifetime routing in wireless sensor networks." *IEEE/ACM Transactions on Networking*, vol. 12, no. 4, pp. 609-619, 2004.

[12] M. Bhardwaj and A.P. Chandrakasan, "Bounding the lifetime of sensor networks via optimal role assignments," *IEEE INFOCOM*, 2002.

[13] M. Cardei, M. T. Thai, Y. Li, W. Wu, "Energy-efficient target coverage in wireless sensor networks," *IEEE INFOCOM*, 2005.

[14] X. Wu, G. Chen and S. Das, "Avoiding energy holes in wireless sensor networks with nonuniform node distribution," *IEEE Transactions on Parallel and Distributed Systems*, vol.19, no.5, pp. 710-720, 2008.

[15] M. Ma and Y. Yang, "SenCar: An energy efficient data gathering mechanism for large scale multihop sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 10, pp. 1476-1488, October 2007.

[16] J. Luo, J. P. Hubaux, "Joint sink mobility and routing to maximize the lifetime of wireless sensor networks: the case of constrained mobility," *IEEE/ACM Transactions on Networking*, vol. 18, no. 3, pp. 871-884, June 2010.

[17] M. Zhao and Y. Yang, "Bounded relay hop mobile data gathering in wireless sensor networks," *IEEE Transactions on Computers*, vol. 61, no. 2, pp. 265-277, Feb. 2012.

[18] M. Zhao. M. Ma and Y. Yang, "Efficient data gathering with mobile collectors and space-division multiple access technique in wireless sensor networks," *IEEE Transactions on Computers*, vol. 60, no. 3, pp. 400-417, March 2011.

[19] M. Zhao and Y. Yang, "Optimization based distributed algorithms for mobile data gathering in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 11, no. 10, pp. 1464-1477, October 2012.

[20] M. Ma, Y. Yang and M. Zhao, "Tour planning for mobile data gathering mechanisms in wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 62, no. 4, pp. 1472-1483, May 2013.

[21] M. Zhao, Y. Yang and C. Wang, "Mobile data gathering with load balanced clustering and dual data uploading in wireless sensor networks" to appear in *IEEE Transactions on Mobile Computing*, 2015.

[22] T. Voigt, H. Ritter, J. Schiller ,"Utilizing solar power in wireless sensor networks," *IEEE International Conference on Local Computer Networks (LCN)*, 2003.

[23] M. Rahimi, H. Shah, G. Sukhatme, J. Heideman and D. Estrin, "Studying the feasibility of energy harvesting in a mobile sensor network," *IEEE International Conference on Robotics and Automation*, 2003.

[24] C. Wang, S. Guo and Y. Yang, "Energy-efficient mobile data collection in energy-harvesting wireless sensor networks," *The 20th IEEE International Conference on Parallel and Distributed Systems (ICPADS 2014)*, Hsinchu, Taiwan, Dec. 2014.

[25] J. Paradiso, T. Starner, "Energy scanverging for mobile and wireless electronics," *IEEE Journal of Pervasive Computing*, vol. 4, no. 1, pp. 18-27, 2005.

[26] C. Park and P. H. Chou, "AmbiMax: autonomous energy harvesting platform for multi-supply wireless sensor nodes," *IEEE International Conference on Sensing, Communication, and Networking (SECON)*, vol. 1, pp. 168-177, 2006.

[27] N. Tesla, "Apparatus for transmitting electrical energy," U.S. Patent 11119732, Dec. 1914.

[28] 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.

[29] A. Kurs, R. Moffatt and M. Soljacic, "Simultaneous mid-range power transfer to multiple devices,"*Applied Physics Letter*, vol. 96, no. 4, article 4102, Jan. 2010.

[30] Powermat, "http://www.powermat.com."

[31] Powercast Corp, "http://www.powercastco.com".

[32] Hevo power, "http://www.hevopower.com."

[33] B. Tong, Z. Li, G. Wang and W. Zhang, "How wireless power charging technology affects sensor network deployment and routing," *IEEE Distributed Computing Systems (ICDCS)*, 2010.

[34] S. He, J. Chen, F. Jiang, D. Yau, G. Xing and Y. Sun,"Energy provisioning in wireless rechargeable sensor networks," *IEEE Transactions on Mobile Computing*, vol. 12, no. 10, pp. 1931-1942, Oct. 2013.

[35] L. Fu, P. Cheng, Y. Gu, J. Chen and T. He, "Minimizing charging delay in wireless rechargeable sensor networks," *IEEE INFOCOM*, pp. 2922-2930, 2013.

[36] Y. Peng, Z. Li, W. Zhang and D. Qiao, "Prolonging sensor network lifetime through wireless charging," *IEEE Real-Time Systems Symposium (RTSS)*, pp. 129-139, 2010.

[37] Z. Li, P. Yang, W. Zhang, and D. Qiao, "J-RoC: a Joint Routing and Charging Scheme to Prolong Sensor Network Lifetime", *IEEE International Conference on Network Protocols (ICNP)*, 2011.

[38] H. Dai, Y. Liu, G. Chen, X. Wu and T. He,"Safe charging for wireless power transfer," *IEEE INFOCOM*, pp. 1105-1113, 2014.

[39] Y. Shu, P. Cheng, Y. Gu, J. Chen and T. He,"TOC: Localizing wireless rechargeable sensors with time of charge," *IEEE INFOCOM*, pp. 388-396, 2014.

[40] Online: "http://www.afar.net/tutorials/fcc-rules".

[41] M. Zhao, J. Li and Y. Yang, "Joint mobile energy replenishment and data gathering in wireless rechargeable sensor networks," *IEEE Transactions on Mobile Computing*, vol. 13, no. 12, pp. 2689-2705, 2014.

[42] Y. Shi, L. Xie, T. Hou and H. Sherali, "On renewable sensor networks with wireless energy transfer," *IEEE INFOCOM*, pp. 1350-1358, 2011.

[43] L. Xie, Y. Shi, T. Hou, W. Lou, H. Sherali and S. Midkiff, "On the renewable sensor networks with wireless energy transfer: the multi-node case," *IEEE International Conference on Sensing, Communication, and Networking (SECON)*, 2012.

[44] S. Guo, C. Wang and Y. Yang, "Joint mobile data gathering and energy provisioning in wireless rechargeable sensor networks," *IEEE Transactions on Mobile Computing*, vol. 13, no. 12, pp. 2836-2852, 2014.

[45] C. Wang, J. Li, F. Ye and Y. Yang, "NETWRAP: An NDN based real-time wireless recharging framework for wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 13, no. 6, pp. 1283-1297, 2014.

[46] Panasonic Ni-MH battery handbook, "http://www2.renovaar.ee/userfiles/Panasonic_Ni-MH_Handbook.pdf".

[47] W. R. Heinzelman, A. Chandrakasan and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," *IEEE Proceedings of the 33rd Annual Hawaii International Conference on System Sciences (HICSS)*, 2000.

[48] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: a hybrid, energy-efficient approach", *IEEE Transations on Mobile Computing*, vol. 3, no. 4, 2004.

[49] P. Vansteenwegen, W. Souffriau and D. Van Oudheusden, "The orienteering problem: a survey," *European Journal of Operation Research*, vol. 209, no. 1, 2011.

[50] C. Miller, A. Tucker and R. Zemlin,"Integer programming formulations and travelling salesman problems," *Journal of the ACM*, pp. 326-329, 1960.

[51] D. Feillet, P. Dejax and M. Gendreau, "Traveling salesman problems with profits," *Transportation Science*, vol. 39, no. 2, 2005.

[52] T. Tsiligirides, "Heuristic methods applied to orienteering," *Journal of the Operation Research Society*, vol. 35, no. 9, pp. 797-809, 1984.

[53] B. Golden, A. Assad and R. Dahl, "The orienteering problem," *Naval Research Logistics* 34, pp. 307-318, 1987.

[54] I. Chao, B. Golden and E. Wasil, "A fast and effective heuristic for the orienteering problem," *European Journal of Operation Research*, vol. 88, no. 3, 1996.

[55] M. Fischetti, J. S. Gonzalez and P. Toth, "Solving the orienteering problem through branch-and-cut," *INFORMS Journal on Computing*, vol. 10, no. 2, pp. 133-148, 1998.

[56] R. Karp. "Reducibility Among Combinatorial Problems," *Complexity of Computer Computations*, pp. 85-103, 1972.

[57] B. Gavish, "A note on the formulation of the m-salesman traveling salesman problem," *Management Science,* 1976.

[58] P. Toth and V. Daniele, "The vehicle routing problem," *Society for Industrial and Applied Mathematics*, 2001.

[59] T. Ralph, "On the capacitated vehicle routing problem," *Mathematical programming*, vol. 94 no. 2-3, pp. 343-359, 2003.

[60] B. Chandran and S. Raghavan, "Modeling and solving the capacitated vehicle routing problem on trees," *The Vehicle Routing Problem: Latest Advances and New Challenges*, pp 239-261, 2008.

[61] R. Fukasawa, "Robust branch-and-cut-and-price for the capacitated vehicle routing problem," *Mathematical programming*, vol. 106, no. 3, pp. 491-511, May, 2006.

[62] M. W. P. Savelsbergh, "Local search in routing problems with time windows," *Annals of Operation Research*, pp. 285-305, 1985.

[63] N. Bansal, A. Blum, S. Chawla and A. Meyerson, "Approximation algorithms for Deadline-TSP and Vehicle Routing with Time Windows," *ACM Symposium on Theory of Computing (STOC)*, 2004.

[64] H.C. Lau, M. Sim and K. M. Teo,"Vehicle routing problem with time windows and a limited number of vehicles",*European Journal of Operation Research*, 148, pp. 559 - 569, 2003.

[65] E. Taillard, P. Badeau, M. Gendreau, F. Geurtin and J.Y. Potvin, "A tabu search heuristic for the vehicle routing problem with soft time windows,"*Transportation Science*, vol. 31, pp. 170- 186, 1997.

[66] J. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, no. 14, 1967.

[67] L.R. Esau and K.C. Williams, "On teleprocessing system design: part II-a method for approximating the optimal network," *IBM System Journal*, vol. 5, pp. 142-147, 1966.

[68] P. Jaillet, "Probabilistic traveling salesman problem," Ph.D. Dissertation, MIT, 1985.

[69] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, *Introduction to Algorithms*, MIT Press, 2001.

[70] W. R. Heinzelman, A. Chandrakasan and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," *IEEE HICSS*, 2000.

[71] C Wang, J Li, F Ye, Y. Yang, "Recharging Schedules for Wireless Sensor Networks with Vehicle Movement Costs and Capacity Constraints",*IEEE SECON*, 2014.

[72] S. Ross, *A First Course in Probability*, 8th Ed, Prentice Hall, 2009.

[73] R. Kershner, "The number of circles covering a set," *American Journal of Mathematics*, vol. 61, pp. 665-671, 1939.

[74] V. Rai and R. N. Mahapartra, "Lifetime modeling of a sensor network,"*IEEE DATE*, 2005.

[75] M. Desrochers, J.K. Lenstra, M.W.P. Savelsbergh and F. Soumis, "Vehicle routing with time windows: optimization and approximation," Elsevier Science Publisher, 1988.

[76] Battery Calculator, "http://www.evsource.com/battery_calculator.php."

[77] R. Jain, D. M. Chiu, W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," *DEC Research Report TR-301*.

[78] W. Zhong, C. Lee and S. Hui, "Wireless power domino-resonator systems with noncoaxial axes and circular structures," *IEEE Trans. Power Electronics*, vol. 27, no. 11, Nov. 2012.

[79] J.O. Mur-Miranda, G. Fanti, Y. Feng, K. Omanakuttan, R. Ongie, A. Setjoadi and N. Sharpe, "Wireless power transfer using weakly coupled magnetostatic resonators," *IEEE ECCE*, 2010, pp. 4179-4186.

[80] F. Zhang, S. Hackworth, W. Fu and M. Sun, "The relay effect on wireless power transfer using witricity,"*IEEE Conf. Electromagn. Field Comput.*, 2010.

[81] B. Lee, A. Hillenius and D. Ricketts, "Magnetic resonant wireless power delivery for distrbuted sensor and wireless systems," *IEEE WiSNet*, 2012.

[82] X. Liu, J. Luo, K. Han and G. Shi, "Fueling wireless networks perpetually: A case of multi-hop wireless power distribution," *IEEE PIMRC*, 2013.

[83] C. Wang, J. Li, F. Ye and Y. Yang, "NETWRAP: An NDN based real-time wireless recharging framework for wireless sensor networks," *IEEE Trans. on Mobile Computing*, vol. 13, no. 6, 2014, pp. 1283-1297.

[84] L. Xie, Y. Shi, T. Hou, W. Lou, H. Sherali and S. Midkiff, "On the renewable sensor networks with wireless energy transfer: the multi-node case," *IEEE SECON*, 2012.

[85] G. Nagy and S. Salhi, "Location-routing: issues, models and methods," *European Journal of Operation Research*, no. 177, pp. 649-672, 2007.

[86] D. Zuckerman, "Linear degree extractors and the inapproximability of max clique and chromatic number," *ACM symposium on Theory of computing*, pp. 681-690, 2006.

[87] G. Frederickson, M. Hecht and C. Kim, "Approximation algorithms for some routing problems," *S*IAM J. Comp., vol. 7, no. 2, pp. 178-193, 1978.

[88] N. Christofides, "Worst-case analysis of a new heuristic for the travelling salesman problem," TR 388, CMU, 1976.

[89] C. Lund and M. Yannakakis, "On the hardness of approximating minimization problems," *Journal of ACM*, vol. 41, no. 5, 1994, pp. 960-981.

[90] T. Marler and J. Arora, "The weighted sum method for multi-objective optimization: new insights," *Structural and Multidisciplinary Optimization*, vol. 41, no. 6, pp 853-862, Jun. 2010.

[91] F. Szidarovszky, M. Gerson and L. Duckstein, "Techniques for multi-objective decision making in systems management," *Elsevier*, 1986.

[92] K. Mori, H. Lim, S. Iguchi, K. Ishida, M. Takamiya and T. Sakurai, "Positioning-free resonant wireless power transmission sheet with staggered repeater coil array (SRCA)," *IEEE Antennas and Wireless Propagation Letters*, vol.11, pp. 1710-1713, 2012.

[93] F. Adib, S. Kumar, O. Aryan, S. Gollakota and D. Katabi, "Interference alignment by motion," *ACM Mobicom*, 2013.

[94] The next generation of wireless power, "http://witricity.com/assets/witricity_infographic_r13_small.pdf".

[95] http://www.gizmag.com/kaist-dipole-coil-resonant-system-wireless-charging/31876

[96] V. Raghunathan, A. Kansal, J. Hsu, J. Friedman and M. Srivastava,"Design considerations for solar energy harvesting wireless embedded systems," *IEEE IPSN*, 2005.

[97] D. Shmoys, E. Tardos and K. Aardal, "Approximation algorithms for facility location problems," *ACM STOC*, 1997.

[98] K. Jain and V. Vazirani, "Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation," *JACM* vol. 48, no. 2, pp. 274-296, 2001.

[99] K. Jain, M. Mahdian, E. Markakis, A. Saberi and V. Vazirani, "Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP," *JACM*, vol. 50, no. 6, pp. 795-824, 2003.

[100] S. Guha and S. Khuller, "Greedy strikes back: improved facility location algorithms," *Journal of Algorithms*, vol. 31, no. 1, pp. 228-248, 1999.

[101] E. Weiszfeld and F. Plastria, "(Translated version) On the point for which the sum of the distances to n given points is minimum," *Annals of Op. Research*, vol. 167, no. 1, pp. 7-41, 2009.

[102] H. Kuhn, "A note on Fermat's problem," *Mathematical Programming*, vol. 4, no. 1, pp. 98-107, 1973.

[103] N. Sharma, J. Gummeson and D. Irwin, "Cloudy computing: leveraging weather forecasts in energy harvesting sensor systems," *IEEE SECON*, 2010.

[104] A. Amis, R. Prakash, T. Vuong and D. Huynh,"Max-min d-cluster formation in wireless ad hoc networks," *IEEE INFOCOM*, 2000.

[105] T. Gonzalez, "Clustering to minimize the maximum intercluster distance," *Theoretical Computer Science*, vol. 38, pp. 293-306, 1985.

[106] Z. Li, Y. Peng, W. Zhang, and D. Qiao, "J-RoC: a joint routing and charging scheme to prolong sensor network lifetime," *IEEE ICNP*, 2011.

[107] UBeam Technology, Website: "http://ubeam.com/".

[108] S. He, J. Chen, F. Jiang, D. Yau, G. Xing and Y. Sun, "Energy provisioning in wireless rechargeable sensor networks," *IEEE TMC*, vol. 12, no. 10, pp. 1931-1942, Oct. 2013.

[109] S. Nikoletseas, R. Theofanis and R. Christoforos, "Low radiation efficient wireless energy transfer in wireless distributed systems," *IEEE ICDCS*, 2015.

[110] A. Kansal, J. Hsu, M. Srivastava and V. Raqhunathan, "Harvesting aware power management for sensor networks," *43rd ACM/IEEE Design Automation Conference*, 2006.

[111] C. Vigorito, D. Ganesan and A. Barto, "Adaptove control of duty cycling in energy-harvesting wireless sensor networks," *IEEE SECON*, 2007.

[112] M. Gorlatova, A. Wallwater, G. Zussman, "Networking low-power energy harvesting devices: measurements and algorithms," *IEEE INFOCOM*, 2011.

[113] R. Liu, P. Sinha and C. E. Koksal, "Joint energy management and resource allocation in rechargeable sensor networks," *IEEE INFOCOM*, 2010.

[114] B. Yuan, M. Orlowska and S. Sadiq, "On the optimal robot routing problem in wireless sensor networks,"*IEEE TKDE*, vol. 19, no. 9, 2007.

[115] R. Sugihara and R. Gupta, "Path planning of data mules in sensor networks,"*ACM Trans. Sen. Netw*, vol. 8, no. 1, pp. 1-27, Aug. 2011.

[116] L. He, J. Pan and J. Xu, "A progressive approach to reducing data collection latency in wireless sensor networks with mobile elements," *IEEE TMC*, vol. 12, no. 7, pp. 1308-1320, July 2013.

[117] S. Shmuel and O. Schwartz, "On the complexity of approximating TSP with neighborhoods and related problems," *Computational Complexity*, 2006.

[118] A, Dumitrescu and J. Mitchell, "Approximation algorithms for TSP with neighborhoods in the plane," *ACM Symp. Discrete Algorithms*, 2001.

[119] Weather underground: "www.wunderground.com/history/".

[120] K. Elbassioni, A. Fishkin, N. Mustafa and R. Sitters, "Approximation algorithms for Euclidean group TSP," *ICALP*, 2005.

[121] C. H. Papadimitriou and K. Steiglitz, "Combinatorial optimization: algorithms and complexity," *Dover Publications*, 1998.