Performance Optimization of Wireless Sensor Networks for Remote Monitoring

Xu Xu

March 2014

A thesis submitted for the degree of Doctor of Philosophy
of the Australian National University

Australian
National
University

Dedicated to my father, Ming Xu.

3 July 1957 - 24 February 2013

Declaration

This thesis is a presentation of the original work except where otherwise stated. I completed this work jointly with my supervisor, Associate Professor Weifa Liang. My contribution to the work is around 80%.

Xu Xu

12 March 2014

Acknowledgements

This thesis would not be accomplished without the help of many people and I would like to express my gratitude to them here.

My gratitude to my supervisor, Associate Professor Weifa Liang, is beyond expression. He has been constantly providing me great support and considerable encouragement to overcome difficulties in my study and life over the past few years. He guided me to become a qualified researcher through expert supervision to my PhD work with his perceptive insight and valuable experience. His rigorous academic attitude and straightforward character have impressed me and established a model for my future research. It is worth mentioning that he helped me out of the most difficult time in my life, easing my worries and resolving my confusions. Dr. Liang not only supervises my research but also keeps my life in the right direction.

I would like to thank the other members in my supervisor panel, Professor Brendan McKay, Dr. Tim Wark, and Professor Eryk Dutkiewicz. Special thanks are given to Dr. Tim Wark and Professor Brendan McKay for their brilliant ideas, professional guidance, and fruitful discussions. Without their support, this thesis would not be possible.

Many thanks to people in the Research School of Computer Science at the Australian National University, for their generous help and assistance in various aspects. They make me feel at home while I am studying overseas. Henry Gardner, Janette Garland, Di Wellach-Smith, Shaw Tyi Tan, Jadon Radcliffe, and James Fellows deserve to be especially appreciated.

I am grateful to my friends, Ting Cao, Baichen Chen, Richard Ren, Zichuan Xu, Qiufen Xia, Wenzheng Xu, and Elly Liang for their great friendship and

kind help during my years at ANU. Their friendship makes my time in Australia more colorful and interesting.

I express my special thankfulness to my beloved husband Yuming Lin, for his love, understanding, support, and tolerance to me every day in my life. He contributed a lot to my entire PhD study, including this thesis, and deserves recognition for any achievement on my part.

Lastly but most importantly, my deepest appreciation goes to my mother Caiyan Wang and late father Ming Xu. They are the greatest parents in the world, bringing me up and supporting whatever I decided to do. In particular, this thesis is dedicated to my respected father Ming Xu. He has spent his life loving me and witnessing each milestone in my life, he would be happy to see me accomplishing this thesis. May his soul rest in peace in heaven.

Publications

Journal

[1] Weifa Liang, Xiaojiang Ren, Xiaohua Jia, and Xu Xu. Monitoring quality maximization through fair rate allocation in harvesting sensor networks. *IEEE Transactions on Parallel and Distributed System*, Vol. 24, pp. 1827-1840, 2013.

[2] Weifa Liang, Jun Luo, and Xu Xu. Network lifetime maximization for time-sensitive data gathering in wireless sensor networks with a mobile sink. *Journal of Wireless Communications and Mobile Computing*, Vol. 13, pp. 1263-1280, Wiley-Blackwell publisher, 2011.

Conference

[1] Xu Xu, Weifa Liang, Xiaohua Jia, and Wenzheng Xu. Maximizing network throughput with minimal remote data transfer cost in unreliable wireless sensor networks. *In Proceedings of 14th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, ACM, July, 2013.

[2] Xu Xu. Network throughput maximization in unreliable sensor networks with a mobile sink. *In Proceedings of 9th International Wireless Communications & Mobile Computing Conference (IWCMC)*, IEEE, July, 2013.

[3] Xu Xu, Weifa Liang, and Zichuan Xu. Minimizing remote monitoring cost of wireless sensor networks. *In Proceedings of 2013 IEEE Wireless Communications and Networking Conference (WCNC)*, IEEE, April, 2013.

[4] Xu Xu, Weifa Liang, Tim Wark, and Jaein Jeong. Maximizing network lifetime via 3G gateway assignment in dual-radio sensor networks. *In Proceedings*

of 37th IEEE Conference on Local Computer Networks (LCN), IEEE, October, 2012.

[5] Xu Xu, and Weifa Liang. Monitoring quality optimization in wireless sensor networks with a mobile sink. *In Proceedings of 14th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, ACM, October, 2011.

[6] Xu Xu, Weifa Liang, and Tim Wark. Data quality maximization in sensor networks with a mobile sink. *In Proceedings of 7th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, IEEE, June, 2011.

[7] Xu Xu, and Weifa Liang. Placing optimal number of sinks in sensor networks for network lifetime maximization. *In Proceedings of 2011 IEEE International Conference on Communications (ICC)*, IEEE, June, 2011.

[8] Weifa Liang, Jun Luo, and Xu Xu. Prolonging network lifetime via a controlled mobile sink in wireless sensor networks. *In Proceedings of 2010 IEEE Global Telecommunications Conference (Globecom)*, IEEE, December, 2010.

Abstract

Wireless sensor networks (WSNs) have gained worldwide attention in recent years because of their great potential for a variety of applications such as hazardous environment exploration, military surveillance, habitat monitoring, seismic sensing, and so on. In this thesis we study the use of WSNs for remote monitoring, where a wireless sensor network is deployed in a remote region for sensing phenomena of interest while its data monitoring center is located in a metropolitan area that is geographically distant from the monitored region. This application scenario poses great challenges since such kind of monitoring is typically large scale and expected to be operational for a prolonged period without human involvement. Also, the long distance between the monitored region and the data monitoring center requires that the sensed data must be transferred by the employment of a third-party communication service, which incurs service costs. Existing methodologies for performance optimization of WSNs base on that both the sensor network and its data monitoring center are co-located, and therefore are no longer applicable to the remote monitoring scenario. Thus, developing new techniques and approaches for severely resource-constrained WSNs is desperately needed to maintain sustainable, unattended remote monitoring with low cost. Specifically, this thesis addresses the key issues and tackles problems in the deployment of WSNs for remote monitoring from the following aspects.

To maximize the lifetime of large-scale monitoring, we deal with the energy consumption imbalance issue by exploring multiple sinks. We develop scalable algorithms which determine the optimal number of sinks needed and their locations, thereby dynamically identifying the energy bottlenecks and

balancing the data relay workload throughout the network. We conduct experiments and the experimental results demonstrate that the proposed algorithms significantly prolong the network lifetime.

To eliminate imbalance of energy consumption among sensor nodes, a complementary strategy is to introduce a mobile sink for data gathering. However, the limited communication time between the mobile sink and nodes results in that only part of sensed data will be collected and the rest will be lost, for which we propose the concept of monitoring quality with the exploration of sensed data correlation among nodes. We devise a heuristic for monitoring quality maximization, which schedules the sink to collect data from selected nodes, and uses the collected data to recover the missing ones. We study the performance of the proposed heuristic and validate its effectiveness in improving the monitoring quality.

To strive for the fine trade-off between two performance metrics: throughput and cost, we investigate novel problems of minimizing cost with guaranteed throughput, and maximizing throughput with minimal cost. We develop approximation algorithms which find reliable data routing in the WSN and strategically balance workload on the sinks. We prove that the delivered solutions are fractional of the optimum.

We finally conclude our work and discuss potential research topics which derive from the studies of this thesis.

Contents

List of Figures

Chapter 1

Introduction

1.1 Introduction of Wireless Sensor Networks

Wireless sensor networks (WSNs) have received significant attention in the last few decades because their applications have brought in numerous benefits, including harsh environment accessibility, feasibility enhancement, labor and material savings, productivity increases, and convenience improvement. The conventional architecture of a wireless sensor network is illustrated in Fig. 1.1, which consists of one sink (also referred to as a base station), and sensor nodes with the number from a few to thousands. A sensor node is a converter that measures a physical quantity and converts it into a signal which can be read by an observer or an instrument [6, 47]. Sensor nodes are deployed in the monitoring region to sense phenomena of interest and cooperatively send the sensed data to the sink wirelessly via one-hop/multi-hop relays, depending on the radio transmit power at a source node and the distance between the node and the sink. The sink further forwards the data to the monitoring center for storage or processing, where the center is typically connected to the sink by the Internet or other types of wired networks.

WSNs have been applied in a variety of areas, such as industrial process control, home automation, disaster recovery, livestock tracking, biomedical health telemonitoring, hazardous environment exploration, and so on [3, 43, 76]. The wide range of applications can be classified into two categories: event detection and environment surveillance. In event detection applications, sen-

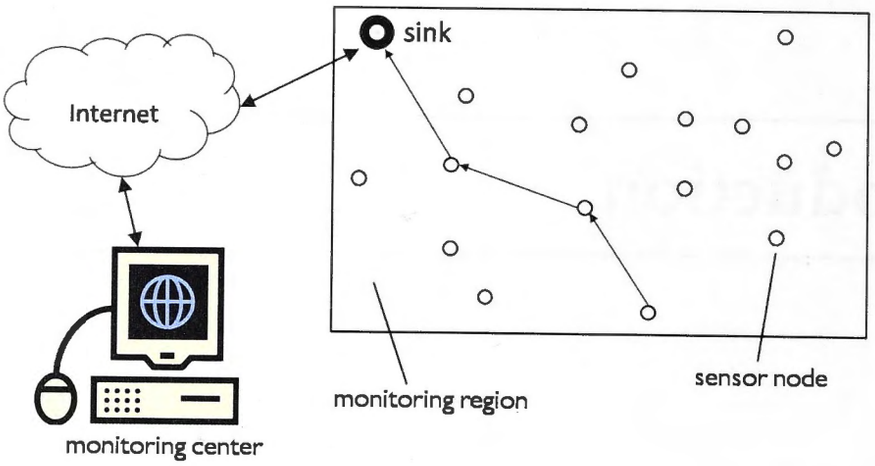


Figure 1.1: The conventional architecture of a wireless sensor network

sor nodes report data to the sink once specified events are detected. Simple events detection can be fulfilled by nodes individually, e.g., when the given blood pressure threshold is exceeded in a heart attack detection system [96], or when a vibration occurs in a seismic sensing application [85]. While detecting complicated events requires the collaboration of multiple nodes, either closely or remotely located in relation to each other. Taking the reconnaissance mission in the battlefield as an example, nodes need to communicate with each other to precisely locate an occurrence [86, 91]. In surveillance applications, nodes take periodic surveillance tasks, for example, a WSN used for process control will report parameters such as pressure, temperature, and density [35]. The reporting frequency for such periodic surveillance depends on specified requirements, which differ from one application to another. For example, the WSN application for health telemonitoring requests real-time data reporting with high fidelity, while that for climate surveillance tolerates much longer latency and even moderate data loss. In addition, the reporting frequency is also constrained by the hardware constraints of sensor nodes, such as limited sampling frequency and buffer size.

1.2 Deploying WSNs for Remote Monitoring

Remote monitoring is for the surveillance of remote regions while the monitoring centers are typically located in a few metropolitan areas that are geographically distant from the monitored regions. To achieve unattended monitoring, sensor networks can be deployed in these remote regions to monitor surrounding environmental phenomena and forward the sensed data to the monitoring centers in real time. Remote monitoring is appealing especially to countries like Australia, where most remote regions are sparsely populated and some of them need to be monitored. One example is to use WSNs on large farms in various states of Australia for growing-environment surveillance (light, humidity, temperature, etc.) and crop attributes monitoring. Sensed data is sent to the remotely located monitoring center for crop growing analysis. Such remote crop paradigm enables users to acquire detailed information from distant farms in no time, dramatically saving labor and cost. Another example is to have WSNs in virgin forests all over Australia to detect forest fires caused by the searing heat experienced each year across Australian states. The remote fire alarm system will greatly reduce the occupational hazard of the forest manager and minimize the potential of property loss and casualties in forest fires.

An essential difficulty in supporting such remote monitoring is transferring data over hundreds of kilometers from the deployed WSN to the monitoring center. Clearly, the multi-hop data collection in which both the data source and destination are located in the same region is no longer applicable. And what makes it more challenging is that there is hardly any wired network access in remote regions because of deficiencies in facilities such as power supply and cables.

To fulfill the remote data transfer from the WSN to the monitoring center, a third party communication service, such as 3G/4G or a satellite network, is leased from a telecommunication company and an amount of *service cost* is

incurred accordingly. With such a leased service, the monitoring center does not have to be a costly static data center, instead, it can be an inexpensive mobile device, such as a laptop or a cell phone. No matter where the device is, as long as it has access to the leased third party network, it will be able to receive data from the deployed WSN. To transmit the sensed data to the third party network, at least one node in the WSN needs to be registered to the service provider and these kinds of nodes are referred to as *sinks*. A sink is communication-compatible with sensor nodes in the WSN and facilities in the third party network, which typically work on different radio bands. Illustrated in Fig. 1.2, data transfer in the remote monitoring scenario consists of two stages: one is the *local data transmission* from the source nodes to the sinks within the monitoring region, and the other is the *remote data transmission* from the sinks to the monitoring center through the third party network.

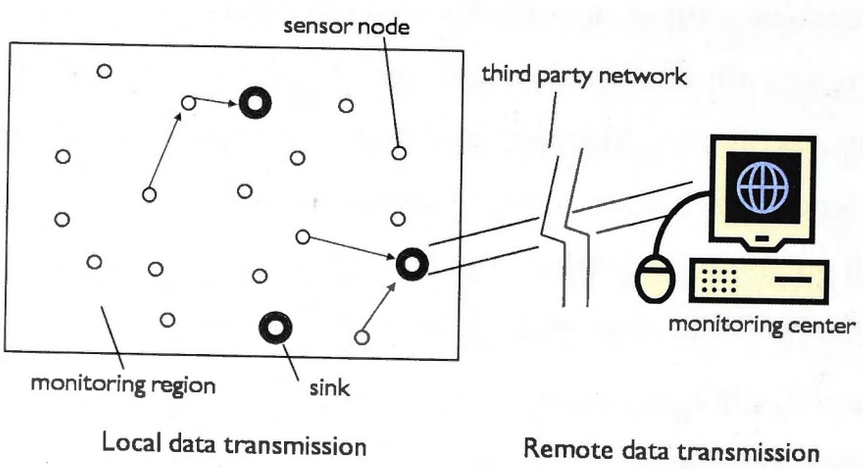


Figure 1.2: An illustration of remote monitoring

1.3 Challenges of Using WSNs for Remote Monitoring

The small and low-cost sensors have limited batteries, computational ability, and buffer sizes, thereby posing challenges for using WSNs for remote mon-

itoring in various aspects including scalability, longevity, monitoring quality, data delivery reliability, and service cost.

- The region monitored by the WSN is usually vast, e.g., a forest or a farm, and this requires the deployment of a large number of sensors in the region. Accordingly, the workload of individual sensors is heavy, resulting in higher complexity and difficulty of managing network resources, such as bandwidth, energy and storage at sensors. The conventional WSN architecture can hardly support large-scale monitoring and will cause network disconnectivity, traffic congestion, and buffer overflow.

- Environmental surveillance of a forest, swamp, or natural reserve is expected to be long-lasting. The longevity of a remote monitoring system depends on the lifetime of the deployed WSN. The remote regions monitored by WSNs are usually inaccessible areas, where the sensor nodes have to rely on limited onboard batteries due to the lack of wired power supply. Even if the energy-harvesting technology (e.g., solar power, wind power) is applied to recharging batteries, the charge rate is unstable and constrained by the solar panel size and the weather. Thus, the life span is a major issue for using WSNs for remote monitoring.

- The monitoring center typically requests sufficient data from the remotely deployed WSN for environment analysis or event recurrence. However, the data collection capacity of an individual sink is restricted by the constrained bandwidth, channel interference, and limited node-sink communication time. As a result, some sensed data will not be successfully collected by the sinks or transferred to the monitoring center, and the quality of the remote monitoring is compromised. Particularly, in WSNs with mobile elements, e.g., nodes and sinks carried by vehicles moving in the monitoring region to track a target or detect a specified phenomenon, routing tables are frequently updated and data is easily lost

during transmission.

- Wireless communication is unreliable in nature due to channel fading [28], interference, multi-path effects, data collisions, and so on. And if a WSN is deployed in a harsh environment with high dynamics, links usually have poor reliability. Consequences are high packet loss, error rates, and intermittent communication disruptions. Additionally, the low-powered radio frequency transceivers at sensor nodes worsen the problem. Therefore, the monitoring center will experience unexpected data loss and receive out of order packets, which poses users great difficulty in analyzing or further processing the received data.
- Leasing the third party network for remote data transfer incurs service cost, which is a unique characteristic for remote monitoring. The cost is charged by the service provider (telecommunication company), and its amount is dependent on the charging strategy and the network throughput. Sending a large volume of data through the third party network will be very costly and may result in a prohibitive penalty if the volume exceeds a pre-defined quota. For long-term monitoring, such cumulative expense would be very high and thereby undermine the applications of WSNs in remote monitoring.

It is challenging to resolve the above mentioned problems due to the constrained resources in WSNs. It is even more difficult to jointly achieve these performance objectives in remote monitoring. For example, selectively turning off nodes in the network for a certain period effectively conserves energy and prolongs the network lifetime; however, this causes data loss and may result in network disconnectivity. On the other hand, keeping every node working all the time provides high data fidelity yet depletes the batteries of nodes much faster and shortens the network lifetime. Similarly, a trade-off exists between the network throughput and the service cost. A large network throughput

is desirable to provide sufficient monitoring quality, however, the greater the volume of data sent through the third party network, the higher the cost. In general, one performance metric is optimized by compromising another or several others, and it is hard to encompass the entire desirable space once and for all [47].

1.4 Research Topics in Remote Monitoring

To provide sustainable monitoring of a remote region with sufficient information sent to the monitoring center at the expense of low cost on the long-distance data transfer, this thesis focuses on (1) making the WSN used for remote monitoring as long-lasting as possible, (2) maximizing the quality of remote monitoring, and (3) optimizing the network throughput with minimal service cost.

1.4.1 Network Lifetime Prolongation

The network lifetime prolongation problem has been extensively studied in a traditional WSN where there is only one static sink with hundreds of thousands of nodes. Such a multi-to-one data transmission model is typically supported by the multi-hop relay mechanism which efficiently conserves energy by utilizing intermediate relay nodes, compared with the one-hop routing protocols which result in prohibitive energy consumption on transmitting data from senders to far-mounted receivers. The multi-hop routing protocols are classified into two categories according to the structures of networks in which they are applied. For flat WSNs, the developed protocols include Flooding, Gossiping, SPIN (Sensor Protocol for Information via Negotiation) [41], and SAR (Sequential Assignment Routing) [20]. Whereas for hierarchical WSNs, LEACH (Low Energy Adaptive Clustering Hierarchy) [40], PEGAGIS (Power-

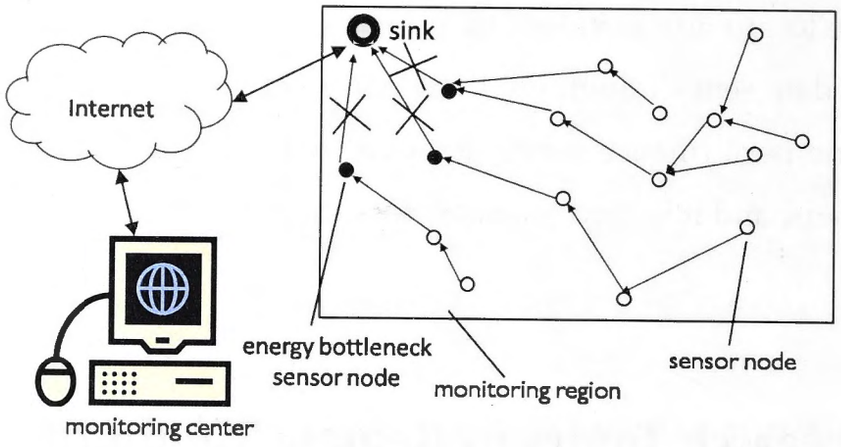


Figure 1.3: An illustration of the single static sink neighborhood problem

Efficient Gathering in Sensor Information Systems) [60], and TEEN (Threshold sensitive Energy Efficient sensor Network protocol) [65] have been devised.

In spite of the merits, the multi-hop data transmission structure does cause new problems, among which the most challenging is the *single static sink neighborhood problem* [58,99], as illustrated in Fig. 1.3. The single static sink receives data from its neighboring nodes, which not only report their own sensed data but also relay data from the other nodes and thus become energy bottlenecks in the WSN. Once these bottleneck nodes deplete their batteries, the sink will be disconnected from the sensor network and network lifetime will end, even if the rest nodes are still fully operational with sufficient residual energy. The energy imbalance issue becomes even worse in large scale WSNs since the sink neighboring nodes have to relay data for a great number of nodes and have their batteries depleting very quickly.

A straightforward approach to tackling the single static sink neighborhood problem is to deploy multiple sinks in the network, which enable data from sensor nodes to flow to different destinations, instead of to a single one. Accordingly, the data relay workload is distributed to a larger number of nodes, thereby mitigating the energy bottleneck problem and prolonging the network lifetime [16,50,99]. Multiple sinks can be applied in WSNs to monitor remote

wild regions such as virgin forests, swamps, and deserts, where it is difficult for sinks to move once they are deployed. The multi-sink arrangement problem has been studied under both heterogeneous and homogeneous WSNs.

In a heterogeneous WSN, where sinks are more powerful than the other nodes, the *multi-sink placement problem* is to deploy a favorable number of sinks at strategic locations in the network to prolong the network lifetime. Network deployment plays a significant role in system performance and has been studied in a variety of fields, such as power station placement in urban planning, database placement in cloud storage, gateway deployment in backbone networks, and router arrangement in LANs. In wireless sensor networks, the node placement largely influences the network performance. For example, variations in node density can eventually lead to unbalanced traffic [94], and a uniform node distribution would result in energy bottlenecks in the network [70]. Thus, on-demand and careful node placement is essential to shape the network topology for desired network performance - low packet loss rate, short delivery latency, or long network lifetime [61, 92, 94, 106]. In most WSN applications, however, nodes are randomly deployed and little control can be exerted. Therefore, controlled placement is pursued on the sinks. Sinks can simply be deployed at pre-defined locations in WSNs, such as Internet access points, or locations with low Signal-to-Noise Ratios (SNRs). However, the determination of sink placement should take into account both the objective and multiple constraint factors. The complexity of solving the sink placement problem depends on the priority of node assignment and sink positioning [106]. If nodes are partitioned into distinct groups prior to finalizing the locations of sinks, the multi-sink placement problem becomes local and can be solved independently within each group [69, 108]. Otherwise, the problem is NP-hard, proven by Bogdanov *et al.* in [16] through a reduction to the dominating set problem in a unit disk graph. The multi-sink placement problem studied in this thesis falls into the latter category.

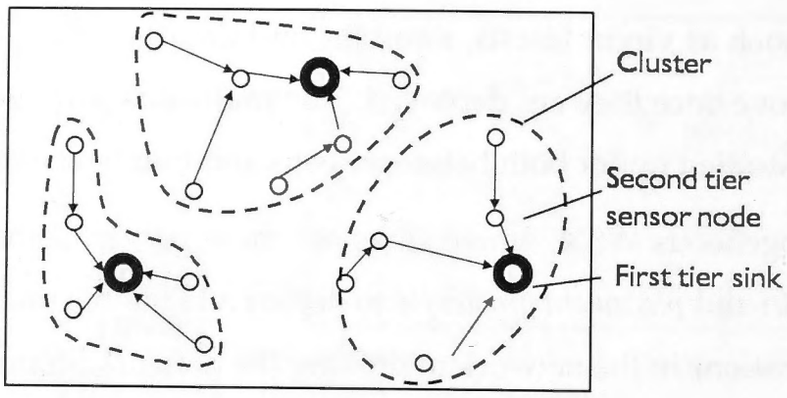


Figure 1.4: A hierarchical wireless sensor network

Whereas in a homogeneous WSN, all sensor nodes are identical, the *multi-sink assignment problem* is to assign selected nodes as sinks and rotate the assignment among nodes to achieve energy balance in the network. A subset of nodes is identified as sinks to collect from other nodes and forward these data to the third party network. Accordingly, sensor nodes are partitioned into two tiers, as illustrated in Fig. 1.4. Nodes in the first tier act as gateways between the second tier nodes and the monitoring center, and they are referred to as gateways, cluster heads, or sinks (in this thesis). Such hierarchical network structure has been proven to be superior to the flat structure in various aspects: enhancing the network scalability [36], reducing routing table sizes at individual nodes by localizing data routing within each cluster [8], conserving communication bandwidth by reducing inter-cluster transmission interference and message exchange [107]. Moreover, clusters are independent from each other and the data routing in one cluster will not be affected by topology changes or route updates in other clusters. The multi-sink assignment problem has been studied for energy conversation by aggregating the collected data at sinks to reduce the number of relayed packets [25], and switching selected nodes to the low-power sleep mode to reduce the rate of energy consumption [103]. A node is chosen to be the sink either because it is richer in resource compared with others, or as pre-assigned by the network designer. The identified sinks then

ship data to the monitoring center directly or collaboratively with each other. Since sinks bear a heavier workload than the other nodes and consume energy faster, dynamic sink-rotation among nodes is important to balance the energy consumption throughout the network. Accordingly, clustering protocols and sink selection approaches have been proposed and proven to be effective in prolonging network lifetime [38, 40, 107].

1.4.2 Monitoring Quality Maximization

Another strategy for the static single sink neighborhood problem is introducing mobile sinks, which travel in the monitoring region and visit all or some of the deployed nodes for data collection. In that case, the workload of data relay is shifted from individual nodes to mobile sinks, which therefore conserves energy at sensor nodes, reduces data collision, and improves delivery reliability. In general, sink mobility has been demonstrated to be a blessing rather than a curse for network performance in lifetime, scalability [99], throughput [75], connectivity [55], and latency [9]. Mobile sinks are appropriate for use in monitoring regions where there are tracks for the sinks to move along, such as natural reserves, battlefields, and farms. Being communication compatible with both sensor nodes and the third party facilities, the sinks traverse the monitoring region to gather data from individual nodes when entering their transmission ranges, and transmit the received data to the monitoring center through the third party network in real-time or at specified moments.

The quality of monitoring a remote region by using a WSN with mobile sinks is mainly compromised by two factors. (1) The variations in sink locations result in frequent routing tables updates at sensor nodes and high probability of data loss, especially if sinks change their locations too often which makes routing tables out-of-date very soon. As a solution, data can be buffered locally at sensor nodes and sent to sinks only when the topology is stable.

However, under some circumstances, sinks keep moving without stop and data collection has to be accomplished in an unstable topology. Also, even if sinks have stops in their travels, the buffer size of each node is finite [81] and thus data has to be transmitted when sinks are still moving to avoid buffer overflow. (2) The limited communication time between mobile sinks and nodes also incurs data loss [99, 101]. If mobile sinks do not stop or stay long enough at specific spots in the network, they will not be able to collect all data stored at the nearby nodes, especially when the amount of stored data is large [84, 110]. Specifically, the monitoring quality depends on the mobility pattern of sinks and the structure of data collection.

According to the freedom of motion, the sink mobility patterns can be classified into *uncontrolled mobility* [49, 77, 80] and *controlled mobility* [14, 15, 63, 64]. (1) Uncontrolled mobile sinks are typically robots, livestock, or vehicles with pre-defined routes, and under the uncontrolled pattern, the trajectory, velocity, direction, and stops of each mobile sink are unable to be programmed for the purpose of a particular goal. One sub-category of this pattern is random moving, where mobile sinks are carried by robots or livestock which roam the sensor network. Shi *et al.* [80] provided theoretical results on the optimal movement of a mobile sink by converting the problems of sink mobility and flow routing from time domain to space domain. The uncontrolled pattern is easy and lightweight to explore, since no network information is required and no movement needs to be arranged. However, optimal network performance can hardly be achieved under the random mobility pattern. The other sub-category of the uncontrolled mobility pattern is predictable movement, where mobile sinks move according to a certain strategic plan or along a given trail [91]. For example, mobile sinks carried on shuttles or airplanes follow the trajectories of the means of transportation which have been routed and scheduled. This pre-defined mobility, though not usually planned for achieving specific goals of network performance, can be utilized to notify the future

locations of a sink and to arrange data routing in advance to avoid unexpected route changes and data loss. (2) Under the controlled mobility pattern, sinks are controlled to move, stop, and sojourn in the monitoring region, and their trajectories are to be found consisting of a series of sojourn locations together with the sojourn time at each of these locations [58, 81, 90, 93, 101]. The sinks can be attached to vehicles or people that follow instructions to move within the monitoring region. The first attempt at controlled mobility was made by Gandham *et al.* in [32], and an ILP (Integer Linear Program) model was studied to determine the locations of multiple mobile sinks, aiming at minimizing the maximum energy consumption among nodes. Later, Luo *et al.* [63] formulated the network lifetime maximization problem as a min-max problem, under the simplified hypothesis that all sensors are uniformly deployed in a circle region. Wang *et al* [90] considered a grid network and studied a joint optimization problem of determining the sink trajectory and its sojourn time at certain deployed nodes so that the network lifetime is maximized. Basagni *et al.* [15] have incorporated two bottleneck constraints on the mobile sink: the maximum distance between its two consecutive stops and the minimum sojourn time at each stop. Liang *et al.* [58] recently considered an additional constraint on the length of the mobile sink trajectory and investigated the length-constrained trajectory finding problem. The controlled mobility pattern is superior to the uncontrolled pattern in improving network performance such as network lifetime [32], data delivery latency [9], and traffic balance [58], by manipulating and customizing sinks movements. For example, a controlled mobile sink can reach a sensor node before buffer overflow occurs at this node to avoid data loss, or before a pre-defined deadline to ensure the required data delivery latency [9], which can hardly be achieved by uncontrolled mobile sinks. Beneficial to network performance though it is, dealing with the controlled mobility is more challenging due to high dynamics and constraint factors [58].

More flexible data collection modes are available in a mobile-sink WSN, compared with the traditional many-to-one data routing mechanism in the conventional WSN with only one static sink. The data collection modes are classified into three modes: one-hop mode, multi-hop mode, and hybrid mode. (1) In the one-hop mode, data is buffered at individual nodes and collected when a sink moves into the transmission ranges of the nodes [81]. The one-hop data collection is the most energy efficient since no data relay is engaged, it however suffers from the delivery latency, which consists of two segments. One is the time for data to be collected by sinks after being generated at nodes. The other is the time for data to be delivered from sinks to the monitoring center, which can be neglected if mobile sinks forward data immediately after they receive them. (2) In the multi-hop mode, mobile sinks collect data generated from all nodes at the same time via multi-hop relay [58]. That is, by visiting and communicating with only one node, a mobile sink is able to gather data from all the other nodes. Such data collection spots are referred to as sink sojourn locations and all these locations form the trajectory of the mobile sink. This mode minimizes data latency at the expense of a shortened network lifetime due to the energy imbalance issue - nodes close to sink sojourn locations are heavily loaded. To mitigate this issue, sink sojourn locations are preferred to be dispersed over the network instead of being intensively distributed, and thus avoid frequent routing updates at nodes and corresponding energy overhead for message exchange. (3) In the hybrid mode, sinks visit a subset of nodes, referred to as *gateways* [98, 101] or *rendezvous points* [93], from each of which sinks collect data generated from a section of nodes in the network. Multiple gateways need to be visited so that the mobile sink can collect data generated from all nodes. This mode eliminates individual node visiting in the one-hop mode and alleviates unbalanced energy consumption under the multi-hop mode, thereby achieving a fine trade-off between data latency and network lifetime.

1.4.3 Network Throughput Optimization with Minimal Service Cost

Network throughput is defined as the volume of data that can flow through a network within a given period. It is usually measured in bits per second (bit/s or bps) or data packets every pre-defined interval. In this thesis, we define network throughput as the amount of data received by the monitoring center within a specified period. The throughput is compromised by unreliable wireless data transmission in the WSN, and an obvious solution is data retransmission, which was originally proposed for wired networks. Retransmission occurs more often in wireless networks than in wired networks, and multiple routing protocols designed for wired networks have been amended accordingly. For example, the non-congestion loss from the packet sender is hidden [12, 17], and the sender is adapted to realize that the data loss is not mainly due to congestion [48]. Another strategy is multi-path routing where data is routed through multiple paths with potentially high reliability [82]. Multi-path routing does not necessarily cause increased energy consumption because it can be superior to the shortest path routing in terms of the total number of transmissions required to deliver data successfully to the destination [27]. Though retransmission and multi-path strategies improve network throughput to some extent, data loss is still unavoidable in unreliable wireless sensor networks.

The reliability of links in the WSN has a great impact on network throughput. A sensor network with robust links delivers a large network throughput if data is always routed through the most reliable path. However, it is hard to tell a path reliability in advance, because the real-time link quality is not known till link failures have in fact occurred. To deal with this issue, researchers have studied predicting link reliability, and the related work can be classified into three categories based on the type of parameters that they study for prediction.

Studies in the first category used hardware parameters (e.g., RSSI, SNR) to detect physical states of a link and predict its reliability [23,31]. These hardware-based parameters can be obtained quickly and inexpensively. However, the accuracy of such prediction is low because the parameters are only updated when a packet is well received, and even they are successfully updated, the link reliability cannot be fully reflected. In the second category [18], software-based parameters (e.g., PRR, ETX) are calculated according to the number of transmitted/received packets. These parameters are able to reflect link reliability more accurately compared with hardware-based parameters, yet high calculation overhead is incurred with low agility. The third category is a hybrid of the first two, using both hardware and software based parameters and making the best use of their advantages. In addition to studying parameters, research on link prediction utilized the temporal and spatial correlation in link quality [13,30,66]. For example, Bas *et al.* [13] applied statistical channel models to studying spatial and temporal characteristics of link quality under various environments with corresponding metrics. They modelled link temporal correlation by a sinusoidal correlation function and showed the dependency of the function on the average link quality.

In addition to reliability of individual links, the routing structure also plays an important role in determining network throughput. For example, in a network consisting of links with uniform reliability, the larger the average number of hops between data sources and destinations, the lower the successful data delivery rate and the smaller the network throughput. The problem of reliable data routing has been investigated over the last decade [66,83,87]. Researchers have been studying the design of reliable data routing protocol with statistical or estimated link reliability information [66,100,102]. Meier *et al.* [66] conducted extensive experiments to analyze link reliability information statistically, based on which they devised a data routing strategy. Related studies on data transmission in unreliable WSNs also include network transport pro-

tocols which improve the end-to-end data transmission reliability [10, 11, 79], and routing protocols that find more reliable data forwarding [62, 105, 109]. One popular track is the hop-by-hop error recovery [83, 87]. Instead of only using the final destination node to detect data loss and request retransmission, intermediate nodes also track data loss. This is efficient in resource cost on end-to-end data recovery, which grows exponentially with the increase in source-destination distance. Wan [87] claimed that loss detection and recovery were preferred to be limited to a small number of hops (ideally one), and presented a simple, scalable, and robust transport protocol PSFQ (Pump Slowly, Fetch Quickly) to find reliable data transmission in WSNs. Another track focuses on routing protocol design for the network throughput improvement [62, 105, 109]. Ye *et al.* [105] provided a robust data delivery protocol in which each node keeps a cost for forwarding a packet along a certain path to the sink, and it is the receiver, not the sender, to decide whether to relay a packet by comparing its cost with that of the sender. Loh [62] used two metrics to manage data routing, and the proposed routing protocol supports reliable data delivery and low latency at the same time.

The service cost of transferring data by the employment of a third party network is proposed in this thesis for the first time. Given a certain network throughput, the amount of service cost depends on the charging strategy of the service provider (telecommunication company). It is usually charged on the basis of a specified charging period, such as fortnightly or monthly, with the fixed+penalty cost model which is widely adopted in cellphone and home-use broadband service nowadays. The cost model consists of a *fixed cost* for a certain data *quota* within the charging period, and a *penalty cost* applied to any data usage exceeding the quota during this period. In a WSN for remote monitoring, such a fixed+penalty model is applied to each sink and the service cost is the sum of costs on sinks in the WSN.

The maximization of network throughput contradicts the minimization of

service cost. Intuitively, sending all sensed data to the monitoring center provides the highest network throughput yet tends to incur very high service cost; whereas no data transmission through the third party network results in zero service cost with nil network throughput. Thus, the highest network throughput and lowest service cost cannot be achieved simultaneously. And the trade-off between the two metrics is non-trivial. For example, a larger throughput does not necessarily correspond to a higher service cost, and vice versa. Both the amount of network throughput and service cost are influenced by the number of sinks, their locations, and the volume of data relayed through individual sinks. On the one hand, an appropriate number of sinks deployed at favorable locations in the network would increase end-to-end data delivery reliability, and lead to a large network throughput. At the same time, the service cost is not necessarily to be high, which is comprised by moderate fixed costs and little penalty if the data relay workload is properly arranged among sinks. On the other hand, undesirable sink deployment and data routing would reduce network throughput with undiminished service cost.

1.5 Research Aims of This Thesis

This thesis aims to develop techniques and approaches for the deployment of WSNs to maintain long-lasting, high-quality, low-cost remote monitoring. Specifically, we study the network lifetime prolongation by exploring multi-sink placement and assignment, monitoring quality maximization in the mobile-sink WSN, and optimization of network throughput and service cost of remote data transfer. The research aims of this thesis are listed as follows.

- Multi-sink placement is explored to extend the network lifetime, with two problems to be addressed. The first one is how many sinks need to be deployed and where to deploy them. An increased number of sinks

reduces the average number of hops between a sensor node and a sink, thereby decreasing the energy consumed on multi-hop data routing and improving data delivery robustness. However, such improvement in network performance is at the expense of an increased cost, since a sink is usually very costly. In addition to the number of sinks, their positioning has a great impact on network performance, and needs to be studied along with the constraint of any physical barrier in the deployed environment. Therefore, we need to explore the trade-off between network lifetime and the number of sinks, and determine the optimal locations of the sinks subject to environmental constraints. The second problem is how to route data from nodes to the deployed sinks. For energy efficiency, we need to properly allocate nodes and route their data to the sinks. The data relay workload among nodes that are directly connected to the sinks should be balanced to mitigate the energy bottleneck issue.

- Multi-sink assignment is studied with the following three aims. First, the energy cost of nodes is to be modelled. In a homogeneous WSN, a node is assigned to work in either the sink mode or the normal mode, and it transits between the two modes within the network lifetime. To enable the two working modes at a single sensor, a supportive architecture is to be designed and mode transition needs to be discussed. Based on the architecture, the energy consumption components of nodes in different modes are to be analyzed and formulated. Second, a subset of nodes should be identified and assigned as sinks. Since a node in the sink mode depletes energy much faster than that in the normal node, a mechanism is to be devised to choose an appropriate set of sinks with the objective of balancing the energy consumption throughout the network. Third, a sink-rotation strategy is to be developed. Rotating the set of sinks is beneficial to mitigating the energy imbalance issue yet frequent rotation

results in energy overhead in message exchange and shorten the network lifetime. The rotation mechanism (e.g., periodic or irregular with to-be-determined intervals) and the rotation interval are to be determined.

- The quality of monitoring a region is to be maximized under the mobile-sink scenario, which involves three issues to be dealt with. First and foremost, monitoring quality should be formulated. This is determined by not only the amount of received data but also the sufficiency of these data to reflect the situation of the monitored region. A large volume of redundant data is less informative than a smaller amount of representative data. Thus, a subset of all sensed data is to be chosen so that the monitored situation is most accurately reflected, subject to the amount of data that can be collected by the mobile sink. Next, the mobile sink is to be scheduled to collect the chosen representative data by visiting selected nodes in the WSN. Accordingly, the data routing from these nodes to the mobile sink is to be found. Lastly, the missing data can be recovered by the received data if there exists a correlation amongst sensed data and if such correlation can be explored.

- The trajectory of a mobile sink is to be found considering the following constraints. (1) Travelling region. The mobile sink either moves along a pre-defined trajectory [101] or within a given area [98]. (2) Sojourn locations. The mobile sink only stops at certain locations instead of anywhere in the monitoring region [32]. Or, it must visit a set of given locations at specified moments [98]. (3) Travelling distance. The mobile sink, usually carried by a vehicle powered by petrol or electricity, is only able to travel before the power runs out [58]. (4) Maximum distance between two consecutive stops. The mobile sink moves at most a specified distance away from its current sojourn location to reduce the possibility of data loss and buffer overflow [58,64]. (5) Minimum sojourn time at each stop. The mo-

bile sink should avoid changing locations too frequently considering the overhead of data routing reconstruction and message exchange. Subject to one or more of the above constraints, the problem of finding a trajectory is very difficult and effective algorithms need to be developed.

- A fine trade-off between network throughput and service cost needs to be explored. The impact of the network throughput on the service cost should be researched. Both the amount of throughput and cost are dependent on how data are routed from source nodes to the sinks. On the one hand, data should be routed along the most reliable paths, aiming at minimizing data loss. On the other hand, data routed to individual sinks should be balanced to avoid exceeding quotas at some sinks with penalties being applied, while quotas at some other sinks are far from being fully utilized and the pre-paid fixed costs are wasted in a sense. Due to the contradiction in achieving the optimization of the two performance metrics, their balance is to be discussed in pre-defined scenarios with specified objectives and constraints. Under a specific scenario, the number of sinks is to be determined and the workload at individual sinks is to be adjusted.

1.6 Thesis Contributions

The main contribution of the thesis is to systematically study the use of wireless sensor networks for remote monitoring, including proposing new concepts, formulating non-trivial optimization problems, and developing novel approaches to solve them. The proposed techniques balance energy consumption throughout a WSN to prolong the network lifetime significantly, route data strategically to reduce data loss and improve the monitoring quality considerably, and adjust data relay workload at individual sinks to optimize net-

work throughput and service cost. Specifically, the thesis contributions are listed as follows.

- Remote monitoring is proposed for the surveillance of a remote region of interest, with monitored phenomena sent to its data monitoring center, which is located geographically distant from the monitored region, by the deployment of a WSN in the region and the employment of a third party communication service for remote data transfer. Under this scenario, key issues are addressed and three optimization problems are studied in this thesis: network lifetime prolongation, monitoring quality maximization, and network throughput optimization with minimal service cost.

- Multiple sinks are explored for network lifetime prolongation. The multi-sink arrangement problem is studied in Chapter 2. Nodes are assigned as sinks dynamically during the network lifetime to balance energy consumption throughout the WSN. The supportive sensor architecture is designed and the energy cost models of nodes working in various modes are formulated with different identified energy consumption components. A scalable algorithm is developed to determine the appropriate set of sinks and schedule the sink rotation. During each rotation, a set of nodes is identified to be switched off for further energy conservation. The proposed algorithm for the dynamic sink assignment significantly prolongs network lifetime while effectively maintaining a required network throughput. Additionally, the multi-sink placement problem is investigated in Chapter 2, where sinks are more powerful nodes and are strategically deployed in the network. The proposed strategy properly trades-off the number of sinks and the network lifetime. Specifically, it identifies the optimal number of sinks, finds the most favorable locations for sink placement, and designs a data routing protocol for energy consumption

balance among nodes.

- The monitoring quality maximization problem in a mobile-sink WSN is explored in Chapter 3, where the motion of the sink is constrained and to be arranged. The data loss during transmission due to the limited node-sink communication time is investigated and the data correlation is studied to recover the missing data by the received data. Based on the analysis of data loss and the data correlation, monitoring quality is formulated for the first time. For the monitoring quality maximization problem, a three-stage heuristic is devised, which selects nodes whose sensed data can reflect the situation of the monitored region most accurately, finds a trajectory meeting the motion constraints for the mobile sink, and schedules energy-efficient data routing from the selected nodes to the sink. The impact of the sink motion constraints on network performance is also investigated through experiments.

- Network throughput is defined and service cost is modelled in Chapter 4. The optimization of these two performance metrics is addressed in two scenarios with specified objectives and constraints. In the budget-oriented scenario, the service cost is to be minimized with guaranteed network throughput. A heuristic is devised for it by identifying the most favorable number of sinks and optimizing the data routing. The performance of the proposed heuristic is analyzed statistically and proven to be appealing. In the throughput-oriented scenario, the network throughput is to be maximized with minimal service cost. An approximate algorithm is developed, which finds the most reliable data routing in an unreliable WSN and controls the service cost at each sink. The delivered solution is proven to be fractional of the optimum.

- For each proposed algorithm, extensive experiments by simulation are conducted. The impacts of constraint parameters on the algorithm per-

formance are studied and the effectiveness of the proposed algorithms is validated in improving network performance in various aspects. The performance of the proposed algorithms is compared with that of comparable existing approaches and their superiority is demonstrated.

1.7 Thesis Overview

The remainder of the thesis is organized as follows. Chapter 2 analyzes energy cost models of nodes in WSNs, discusses the problems of multi-sink placement and assignment in two types of WSNs, and proposes solutions for them separately. Chapter 3 focuses on maximizing monitoring quality when using a mobile sink for data collection in the WSN. It also studies the impact of sink motion constraints on network performance. Chapter 4 models the performance metrics to evaluate the use of WSNs for remote monitoring and formulates optimization problems under different scenarios, for which algorithms with guaranteed performance are developed accordingly. Chapter 5 summarizes the thesis and proposes future work.

Chapter 2

Network Lifetime Maximization by Multiple Sinks

2.1 Introduction

Multi-sink sensor networks are proposed to improve network performance including lifetime [99], scalability [69], average data delivery latency [108], system throughput [75], and network connectivity [29]. Particularly, using multiple sinks is proven to be effective in mitigating the energy imbalance problem by distributing data routing workload more evenly throughout the network. In this chapter, we consider deploying multiple sinks for data collection in a WSN, and study network lifetime maximization of such a multi-sink WSN. To be able to communicate with both the sensor nodes, which typically work on low-power radios (e.g., IEEE 802.15.4), and third party facilities, which usually employ high-bandwidth radios (e.g., 3G or 4G), the sinks must be equipped with dual radios: a low-power radio for local communication with sensor nodes in the WSN, and a high-bandwidth radio which is compatible with the third party communication protocol. We first analyze the energy cost models of sinks and sensor nodes, and define the multi-sink assignment problem and the multi-sink placement problem in homogeneous and heterogeneous WSNs respectively, both with the objective of network lifetime prolongation. We then provide heuristics for them. We finally conduct extensive experiments

by simulation to evaluate the proposed algorithms and investigate the impact of different constraint parameters on the network lifetime.

The multi-sink assignment problem in homogeneous WSNs is to dynamically assign and rotate nodes to work as sinks, aiming at balancing energy consumption and prolonging network lifetime. It has been extensively studied in the traditional WSN [38, 40, 107]. Low-energy adaptive clustering hierarchy (LEACH) proposed by Heinzelman *et al.* in [40] is one of the most well-known dynamic sink assignment protocols. Based on the assumption that the network lifetime consists of a number of rounds, LEACH randomly selects a fixed number of sensor nodes as cluster heads in different rounds. Meanwhile, the radio components of non-cluster-head nodes are turned off all the time except during data transmission, in order to minimize the energy dissipated at individual sensor nodes. Each node decides whether to become a cluster head independently, according to the suggested percentage of cluster heads in the network and the number of times it has been selected as cluster head so far. LEACH has been improved by later works [38, 107]. For example, Handy *et al.* [38] modified LEACH by replacing the stochastic cluster head selection with deterministic selection. They analyzed bad-case-scenarios where all cluster heads are located near the network edges and data from other sensor nodes have to bridge long routes to reach the cluster heads. They also proposed a new threshold for each node to become a cluster head by considering its current residual energy. The modified strategy was shown to increase the network lifetime delivered by LEACH by 30%.

The dynamic multi-sink assignment problem studied in this chapter differs from the related work in several aspects, including energy cost model, network lifetime definition, sink selection strategy, and data routing approach. With two types of radios embedded on each node, the energy consumption components on individual nodes vary from each other, which depend on their working modes. Another unique feature is that we incorporate network through-

put as a constraint to the objective of network lifetime maximization, and explore a fine trade-off between them. In addition, we take residual energy at individual sensor nodes into consideration when choosing sinks, in the hope of balancing energy consumption more evenly throughout the network.

The multi-sink placement problem in heterogeneous WSNs is to identify the optimal number of sinks and find their locations with the objective of distributing data relay workload more evenly in the network. Previous researches on the multi-sink placement problem aimed to improve network performance in various aspects including network lifetime [99], data delivery latency [72, 108], network scalability [69], and system throughput [75]. Among them, the studies that focused on network lifetime usually assumed that the number of sinks is pre-defined [16,51,69,75] or bounded by a given value [32]. As an example, Bogdanov *et al.* [16] considered the multiple sink placement problem under different data generation rates. They dealt with several special communication graphs by proposing heuristic algorithms. With the objective of maximizing network lifetime, they aimed to find the optimal positioning of multiple sinks through the network flow technique, assuming that the possible locations of sinks were exactly the locations of the sensors. Qiu *et al.* [75] proposed two linear programming solutions by incorporating an interference model and a fault tolerance model into the problem formulation. Kim *et al.* [51] proposed how to place multiple sinks and how to route data traffic from sensor nodes to these sinks. The proposed algorithm improved the network lifetime as well as the fairness. Kim *et al.* [50] studied the same problem by employing linear programming and K-means clustering techniques, under the constraints of the residual energy of sensors, the data generation rate, and potential sink locations. Gandham *et al.* [32] optimized the placement of sinks by formulating the problem as an integer linear programming (ILP), aiming at minimizing the maximum energy consumption among sensors while minimizing the total energy cost on communication. Another track of research

related to the multi-sink placement is seeking the minimal number of relay nodes in WSNs [88,89,95]. Xu *et al.* [95] aimed to find the minimum number of relay nodes and their locations to meet the constraints of network lifetime and connectivity. This problem was shown to be equivalent to finding the minimum set covering, thus NP-hard, and a sub-optimal solution was proposed. Their later work [88, 89] studied the smallest number of relay nodes for routing traffic balance. Experimental results showed that the proposed approach provides near optimal solutions in a number of scenarios.

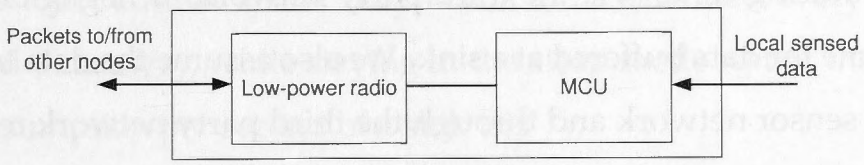
Different from the previous work about the multi-sink placement problem, we study the identification of sink positions from a given location space, and devise a routing protocol with the constraint of an upper-bound on the number of hops between each node and its nearest deployed sink. We jointly deal with the optimal number of sinks for placement and the data routing for network lifetime maximization, subject to the mentioned constraints. Moreover, the proposed data routing is based on tree structure, due to the difficulty of flow control [16, 75] at each node at each time instance.

The remainder of this chapter is organized as follows. Section 2.2 formulates the energy cost model and defines the multi-sink assignment and placement problems. Section 2.3 and 2.4 propose corresponding algorithms for the network lifetime prolongation problems. Section 2.5 includes a series of simulations to validate the proposed algorithms and the experimental results. Section 2.6 concludes this chapter.

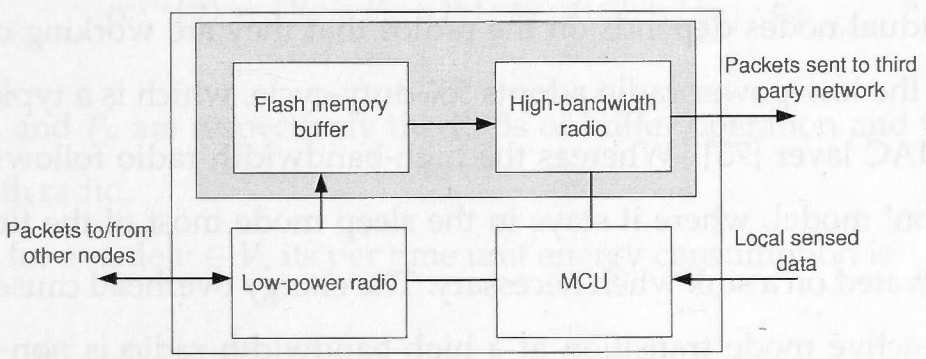
2.2 System Model and Problem Definition

We consider a wireless sensor network $G(V, E)$, where V is the set of sensor nodes with $n = |V|$, and E is the set of links between nodes. Sensor nodes have identical data generation rate r_g and their locations are stationary and known a *priori*. Each node in V is equipped with at least one low-power radio

interface, and the nodes with only low-power radios embedded are referred to as *single-radio nodes*. Some nodes (maybe all) in V are equipped with high-bandwidth radios and they are referred to as *dual-radio nodes*, which can work on either type of the radios, or both of them. The transmission range of the low-power radio is r and there is a link between two nodes if they are within r of each other.



(a) Single-radio node architecture



(b) Dual-radio node architecture

Figure 2.1: Illustration of node architectures

The architectures of single-radio nodes and dual-radio nodes are illustrated in Fig. 2.1. A single-radio node has two components, MCU for sensing and low-power radio for local communication with other nodes in the sensor network. Whereas a dual-radio node has four components including MCU, low-power radio, flash memory buffer for data temporary storage, and high-bandwidth radio for data transmission to the third party network. Dual-radio nodes with both radios on are acting as *sinks*, which collect data from all the other nodes via tree-based routing structure, buffer the data temporally, and further transmit them to the third party network.

We assume that no buffer overflow occurs during the network lifetime and there is no data aggregation at each relay node when proceeding data routing. To ensure the data freshness, we define the *data delivery delay* as the time for data to be received by the monitoring center after being generated, and the delay is required to be bounded by a user-specified threshold D, with $D > 0$. We assume that data transmission time (from a node to a sink, and from a sink to the monitoring center via the third party network) is negligible compared with the time for data buffered at a sink. We also assume the data transmission within the sensor network and through the third party network reliable.

The powers of different radios vary greatly thus the energy consumption at individual nodes depends on the radios that they are working on. In this chapter, the low-power radio adopts 5% duty-cycle, which is a typical setting in the MAC layer [73]. Whereas the high-bandwidth radio follows the 'not-always-on' model, where it stays in the sleep mode most of the time, and is only activated on a sink when necessary. The energy overhead caused by such sleep-to-active mode transition at a high-bandwidth radio is non-negligible and must be taken into account, denoted by E_o.

The energy consumption of a node working only on low-power radio, which could be either a single-radio node or a dual-radio node with only low-power radio on, is dominated by energy cost on data reception and transmission over the low-power radio, and the energy consumption per second of such a node v is

$$ec^{(s)}(v) = P_l \cdot r_g \cdot d(v), \tag{2.1}$$

where P_l is the power-per-bit (PPB) of the low-power radio, and $d(v)$ is the number of descendants of node v in the routing tree (including itself), which send their data to a sink through v.

The energy consumption of a sink (a dual-radio node with both radios

on) is constituted by four components: (i) data reception over the low-power radio; (ii) buffer operation; (iii) data transmission over the high-bandwidth radio; and (iv) high-bandwidth radio mode transition. The first three components are determined by the amount of data relayed by a sink, while the last component depends on the frequency of the sleep-to-active mode transition at the high-bandwidth radio. To meet the data delay requirement, the high-bandwidth radio at a sink only needs to be activated every D to send the buffered data away, while staying in the sleep mode rest of the time. The energy consumption per second of a sink v is

$$ec^{(d)}(v) = (P_b + P_h + P_l) \cdot r_g \cdot d(v) + \lfloor \frac{1}{D} \rfloor \cdot E_o, \tag{2.2}$$

where P_b and P_h are respectively the PPBs of buffer operation and the high-bandwidth radio.

Thus, for a node $v \in V$, its per time unit energy consumption is

$$ec(v) = \begin{cases} ec^{(d)}(v) & \text{if } v \text{ is a sink,} \\ ec^{(s)}(v) & \text{if } v \text{ is a node with only the low-power radio on,} \\ 0 & \text{if } v \text{ is not working on either radio.} \end{cases} \tag{2.3}$$

Network lifetime is defined as the time before the monitoring center is no longer able to receive a specified amount of data from each time of data transmission, and it is denoted by L. Such pre-defined amount of data required to be received by the monitoring center at each time of data transmission is referred to as *network throughput*. Applications have different throughput requirements, some of which strictly require to receive all the sensed data while some others only concern general situation of the monitored region and tolerate data loss.

Given a sensor network $G(V, E)$ with multiple sinks that have access to the third party network for remote data transfer, the network lifetime maximiza-

tion problem is considered in the following two scenarios. (1) In a homogeneous WSN, where nodes are identical and every node could become a sink during the network lifetime, data from at least α percentage of all nodes is required to be received by the monitoring center at least every period of D, with $0 < \alpha \leq 1$, and $D > 0$. Due to the fact that the power of high-bandwidth radio will drain sinks' batteries quickly and shorten the network lifetime, energy consumption among nodes needs to be balanced to prolong the network lifetime. Also, to guarantee the required network throughput, adequate number of nodes should be included in the routing trees rooted at the sinks. The *throughput guaranteed network lifetime maximization problem* is to appropriately arrange the two radios at individual nodes for the identification of a proper set of sinks out of V and the delay-constrained data forwarding from sinks to the remote monitoring center, such that the network lifetime is maximized, subject to that data from at least α percentage of nodes should be routed to the sinks. (2) In a heterogeneous WSN, where energy unconstrained sinks are to be placed at locations selected from a given set of *potential sink locations* S, all sensed data is required to be sent from the sources to the sinks within a specified number of hops h. The parameter h is introduced to upper-bound the number of hops between each pair of data source and destination to avoid long data delivery delay and low end-to-end reliability. The *h-hop constrained multiple sink placement problem* is to place the optimal number of sinks at preferable locations in S such that the number of hops between each node and its nearest sink is no more than h, meanwhile, the network lifetime is maximized under this sink deployment.

To solve the network lifetime maximization problems, we need to jointly solve the following issues: (i) determining the optimal number of sinks and their locations in G, (ii) scheduling radios at sinks to meet the data delivery latency constraint D, (iii) devising a routing forest consisting of trees rooted at the identified sinks spanning selected or all the deployed nodes, and (iv)

updating the set of sinks and the routing forest when necessary, such that the resultant network lifetime is maximized with the required network throughput or the h-hop constraint met.

2.3 Multiple Sink Assignment

We first deal with the throughput guaranteed network lifetime maximization problem in a homogeneous WSN, where every sensor node is equipped with dual radios and could become a sink during the network lifetime. There are applications which require to receive a certain amount of data to for a general understanding of the monitoring region, such as crop monitoring and environmental surveillance. Let $0 < \alpha \leq 1$ be the network throughput threshold, which indicates that data from at least α percentage of nodes need to be transmitted to the sinks. The n dual-radio nodes are classified into two categories: *active* nodes that send their data to the sinks (sinks are always active nodes), and *inactive nodes* that do not need to send data and thus are switched to the sleep mode for energy conservation. Sinks have two radios on and active nodes have only the low-power radio on. Inactive nodes do not work on any radio. The per time unit energy consumption of sinks and the two types of nodes has been formulated in Eq. (2.3).

The network lifetime in such a homogeneous network is the time before the monitoring center no longer receives data from α percentage of nodes. Referring to the energy cost models analyzed in the previous section, sinks consume more energy than the other nodes thus sinks are to be rotated among the n nodes during the network operation to distribute the energy consumption more evenly and prolong the network lifetime accordingly. We consider a periodic rotation mechanism, by assuming that the network lifetime is comprised of $R + 1$ rounds and sinks are re-selected at the beginning of every round. The first R rounds are with equal duration τ, and the last round is with duration

$\tau' < \tau$, i.e., $L = R \cdot \tau + \tau'$, shown in Fig. 2.2. We assume that the number of

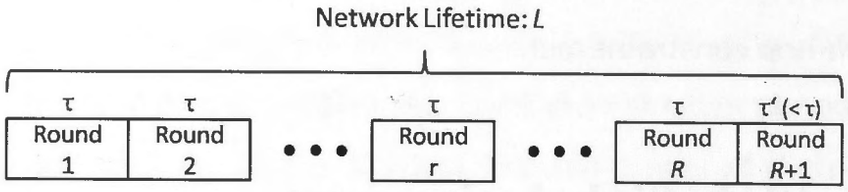


Figure 2.2: $R + 1$ rounds in network lifetime

sinks is fixed constant m in each round during the network lifetime. Given a set of sinks S, let $V_{(r)}(s)$ be the set of nodes transmitting data to a sink $s \in S$ in round r. And $\bigcup_{s \in S} V_{(r)}(s)$ is the set of active nodes in round r. Denote by $er(v)$ the residual energy of node v. In the first R rounds, each active node must have enough residual energy to survive at least the duration of τ, that is,

$$\frac{er(v)}{ec(v)} \geq \tau, \text{ for any } v \in \bigcup_{s \in S} V_{(r)}(s), 1 \leq r \leq R. \tag{2.4}$$

$\frac{er(v)}{ec(v)}$ is the *residual lifetime* of node v. Whereas in round $R + 1$, some active nodes run out of energy and the duration of last round τ' is the shortest residual lifetime among the active nodes.

$$\tau' = min\Big\{\frac{er(v)}{ec(v)} \mid v \in \bigcup_{s \in S} V_{(R+1)}(s)\Big\}, \tag{2.5}$$

and $\tau' < \tau$. Also, in order to guarantee the required network throughput throughout the network lifetime, the number of active nodes must be at least $\lceil \alpha \cdot n \rceil$ in each round.

$$\sum_{s \in S} |V_{(r)}(s)| \geq \lceil \alpha \cdot n \rceil, \ 1 \leq r \leq R + 1. \tag{2.6}$$

With given network throughput threshold α, data delivery delay bound D, the number of sinks m, and the duration of each round τ in the network lifetime, the throughput guaranteed network lifetime maximization problem

is to identify m sinks every period of τ to relay data from at least α percentage of nodes in the network to the third party network at each time interval D, such that the network lifetime is maximized. The data delay requirement can be met by activating the high-bandwidth radio at each sink every D to flush the buffered data. The problem is thus equivalent to identifying m sinks out of the n nodes and finding an energy-efficient routing forest consisting of m trees rooted at the identified sinks to route data to the monitoring center for each round such that network lifetime is maximized, subject to the network throughput requirement being guaranteed in each round.

To achieve the objective, we need to (i) identify an appropriate set of m sinks among the n nodes every period of τ to balance energy consumption throughout the network, (ii) choose active nodes within each round to meet the throughput requirement all the time, and (iii) route data from the active nodes to the m sinks energy-efficiently to conserve energy. We provide a heuristic for the throughput guaranteed network lifetime maximization problem. Recall that the network lifetime consists of a number of rounds with identical duration as well as the last round with a shorter duration. Maximizing the network lifetime is converted to finding the largest number of rounds within the lifetime and the longest duration of the last round. To this end, we first propose an energy-efficient data routing mechanism to balance energy consumption among nodes, subject to the amount of data routed to the monitoring center meeting the throughput requirement. We then determine the network lifetime by examining residual energy at nodes as the network operates.

2.3.1 Routing Forest Establishment

In this section we design the protocol for data routing within each round of the network lifetime. A routing forest is to be established to span at least α percentage of nodes in m trees rooted at sinks. We decompose the routing

forest establishing problem into three sub-problems: identifying the smallest set of active nodes to meet the network throughput requirement, partitioning the active nodes into m subsets such that the graph induced by each subset is connected, and finding the routing tree in each graph to maximize the residual energy among sensors in this graph after the duration of τ.

A set of active nodes is to be identified so that the data generated from them can be received by the base station via m sinks to meet the required network throughput. Nodes with relatively high residual energy should be chosen active so that the energy consumption is more balanced throughout the network, while the other nodes with less energy left are turned inactive. Also, in order to make sure that these active nodes are able to reach their corresponding sinks, the graph induced by the set of active nodes should contain at most m connected components. Otherwise, m sinks will not be adequate to successfully relay data from these active nodes to the third party network. This constraint is referred to as the *m-component* constraint.

Let V' be the set of active nodes to be identified. We sort the nodes in V by their residual energy in non-increasing order. Let $v'_1, v'_2, \ldots v'_n$ be the sorted node sequence, where $er(v'_i) \geq er(v'_j)$, $1 \leq i < j \leq n$, $1 \leq r \leq R+1$. We then find the smallest set of active nodes V' consisting of the first i_{min} nodes in the sequence, subject to the throughput threshold α and m-component constraints. To meet the throughput requirement, $i_{min} \geq \lceil \alpha \cdot n \rceil$. And subject to the m-component constraint, the number of connected components in the graph should not be greater than m. Let $G[V']$ be the graph induced by the nodes in V', and $CC(G[V'])$ be the number of connected components in $G[V']$. $CC(G[V'])$ should be no greater than m. The active nodes identification problem is to find the smallest set $V' = \{v'_1, \ldots, v'_{i_{min}}\}$ such that $i_{min} \geq \lceil \alpha \cdot n \rceil$ and $CC(G[V']) \leq m$. Here we adopt the binary search to efficiently locate the smallest i_{min} subject to the two constraints. After i_{min} is found, add the first i_{min} nodes from the node sequence to V' to become the active nodes.

The set of active nodes V' is then to be partitioned into m subsets, corresponding to the m sinks. Assume that there are m' connected components in $G[V']$, where $m' \leq m$. Let $\mathcal{S} = \{S_1, S_2, \ldots, S_{m'}\}$ be the collection of vertex sets of the m' connected components. If $m' = m$, the active nodes have already been in m subsets and no further partition is needed. Otherwise, we conduct the partition as follows. We select a set with the largest number of vertices, $S_l = max\{|S_i| \mid S_i \in \mathcal{S}, 1 \leq i \leq m'\}$ and remove it from $\mathcal{S}$. We partition graph $G[S_l]$ induced by S_l into two connected sub-graphs with disjoint vertex sets S_{l1} and S_{l2}, such that the difference between the vertex numbers in these two sets, denoted by $diff = abs(|S_{l1}| - |S_{l2}|)$, is minimized, where $S_{l1} \cup S_{l2} = S_l$, $S_{l1} \cap S_{l2} = \emptyset$. To this end, we find the minimum-cut in $G[S_l]$ between each pair of nodes s and $t \in S_l$, where the removing of edges in the minimum cut will partition the original graph into two connected sub-graphs with disjoint vertex sets S_{l1} and S_{l2}, and a corresponding value of $diff$. Select the cut with the smallest value of $diff$ and remove the edges in this cut from $G[S_l]$ to obtain two vertex sets S_{l1} and S_{l2}. Put S_{l1} and S_{l2} into $\mathcal{F}$ and increase the number of connected components m' by 1. This procedure continues until $m' = m$. As a result, active nodes are partitioned in m subsets $S_1, S_2, \ldots, S_m$, and m connected graphs $G_i = G[S_i]$ induced by S_i are obtained, $1 \leq i \leq m$.

For each connected graph G_i, a routing tree T_i is to be found such that after the duration τ, the minimum residual energy among nodes in this graph is maximized. Such a routing tree is referred to as the *max-min tree*. With a given a root, building a max-min tree was discussed in [57]. We adopt the method in [57] to build a max-min tree $T_i(v)$ rooted at each vertex $v \in G_i$. The lifetime of tree $T_i(v)$ equals the minimum residual lifetime of nodes in the tree, $L(T_i(v)) = min\{\frac{er(u)}{ec(u)} \mid u \in S_i\}$. The tree with the longest lifetime is selected as the routing tree for nodes in G_i, denoted by T_i, and the root is the sink. That is, $L(T_i) = max\{L(T_i(v)) \mid v \in S_i\}$. In practice, to save the computation time, we only build max-min trees rooted at $|S'|$ nodes with the highest residual energy

in G_i, where $1 \leq k < S_i$, instead of all vertices. By using the same approach, m routing trees $T_1, T_2, \ldots, T_m$ are built in $G_1, G_2, \ldots, G_m$ for data collection and the corresponding m roots are serving as sinks within this round.

2.3.2 Network Lifetime Determination

We then devise an algorithm to determine the network lifetime. The algorithm proceeds iteratively, where each iteration represents a round in the network lifetime. In each round, the routing forest establishment algorithm is first called to form m trees for data collection in the current round. Then whether or not this round is the last round of network lifetime should be determined. Denote by $l_{min} = min\{\frac{er(v)}{ec(v)} \mid v \in V(T_i), 1 \leq i \leq m\}$ the minimum residual lifetime of nodes in these trees, where $V(T_i)$ is the set of nodes in tree T_i. The value of l_{min} is compared with τ. (i) If $l_{min} \geq \tau$, it means no active node will use up the residual energy in this round and the network lifetime will not end at this stage. The network lifetime L is increased by τ and the residual energy of each node $v \in V$ is updated as follows,

$$er(v) = er(v) - \tau \cdot ec(v) - e_{\Delta}, \tag{2.7}$$

where e_{Δ} is the extra energy cost at each node per round due to the fact that the execution of the routing forest establishment algorithm is accompanied with exchanging messages between nodes and distributing the results to the network. (ii) Otherwise ($l_{min} < \tau$), it means that some active nodes in these m routing trees are no longer be able to survive the duration of τ and the current round is the last round in the network lifetime. L is increased by $\tau' = l_{min}$ and the algorithm terminates. In the end, the network lifetime L is delivered. We refer to the proposed heuristic as `Dynamic_Alg` and the detailed description of the algorithm is in **Algorithm 1**.

Algorithm 1: `Dynamic_Alg`

Input : $G(V,E)$, α, τ, m
Output: Network lifetime L

$L \leftarrow 0$; $r \leftarrow 0$; *terminate* $\leftarrow$ *false* ;
$er(v) \leftarrow IE$, for each $v \in V$;
repeat
  $r \leftarrow r+1$;
  /* stage 1: identify the set of active nodes V' */ ;
  Let $v'_1, v'_2, \ldots v'_N$ be the node sequence in V sorted in non-increasing order of residual energy;
  Binary Search for the smallest set $V' = \{v'_1, \ldots, v'_{i_{min}}\}$ such that $i_{min} \geq \lceil \alpha \cdot N \rceil$ and $CC(G[V']) \leq m$;
  /* stage 2: partition V' into m subsets */ ;
  m connected graphs induced by these subsets are obtained, G_i, with $1 \leq i \leq m$;
  /* stage 3: build max-min tree in each graph G_i */ ;
  A routing forest consisting of m trees is obtained, T_i, with $1 \leq i \leq m$;
  $l_{min} \leftarrow min\{\frac{er(v)}{ec(v)} \mid v \in V(T_i), 1 \leq i \leq m\}$;
  if $l_{min} \leq \tau$ **then**
    /* last round */ ;
    $\tau' \leftarrow l_{min}$; $L \leftarrow L + \tau'$;
    terminate $\leftarrow$ *true* ;
  else
    $L \leftarrow L + \tau$;
    Update nodes' energy according to Eq.(2.7) ;
until *terminate*;

Theorem 2.1. *For a dual-radio homogeneous wireless sensor network $G(V,E)$ with network throughput threshold α, the throughput guaranteed network lifetime maximization problem can be solved by algorithm* `Dynamic_Alg` *with complexity $O(n^3 \log n)$ within each duration of the network lifetime, where n is the number of nodes in G.*

Proof. Algorithm `Dynamic_Alg` is conducted every τ within the network lifetime, and consists of the following three parts. Finding the number of connected components in a graph is implemented in $O(|E|+n)$, using either Breadth-First Search or Depth-First Search. Partitioning active nodes takes

$O(n^3 \log n)$ time [46]. And building a max-min tree rooted at a given node takes $O(|E|n^2)$ time [57]. Thus the complexity of `Dynamic_Alg` is $O(n^3 \log n)$. □

2.4 Multiple Sink Placement

We now solve the h-hop constrained multiple sink placement problem in throughput-oriented applications, where data sensed from all nodes are required to be transmitted to the remote monitoring center. In WSNs deployed for such purpose, sinks equipped with dual radios, adequate battery supplies and storages are utilized for data transmission from the sensor network to the third party network, while the other nodes are cheap single-radio chips deployed only for data sensing and local transmission. For the sake of simplicity, the set V only consists of n single-radio nodes, excluding the sinks whose number and locations are to be determined. The set of potential sink locations S is pre-defined by users, $S = \{s_1, s_2, \ldots, s_{|S|}\}$, with $|S| \leq n$. In practice, the sink potential locations are typically determined by the terrain of the monitoring region. For instance, it is inappropriate to place a sink at a barrier that obstructs the wireless communication between nodes and the sink. Therefore, instead of assuming any spot in the monitoring region is suitable for sink placement, we define such a set of potential locations S. The hop constraint h quantifies the extent of multi-hop routing and trades off between the network lifetime and the number of sinks to be placed. A larger h may result in a smaller number of sinks but will cause higher energy consumption on data relay and a shorter network lifetime. On the contrary, the ideal situation for maximizing the network lifetime is $h = 1$ since each node can transmit its data to a sink directly and no data relay is required. Obviously, such an improvement on network lifetime is at the cost of using a prohibitively large number of sinks if the monitoring region is large and the transmission range of nodes is small.

An appropriate value of h can achieve a fine trade-off between the network lifetime and the number of sinks.

The lifetime of such a heterogeneous WSN is the time of the first node's failure due to the depletion of its energy [19]. Because the sinks are energy unconstrained, the network lifetime is determined by the energy consumption of the single-radio nodes in V. Let $S' \subseteq S$ be the set of chosen locations for sink placement and T_s be the tree rooted at sink $s \in S'$. The energy consumption of sensor nodes on wireless communication per time unit can be obtained by Eq. (2.1). Denote by $C_T(s)$ the set of children of sink s in T_s, which are within the transmission range of s. Nodes in $\{C_T(s) \mid s \in S'\}$ relay data for other nodes in the network thus consume energy faster, referred to as the *energy bottleneck nodes*. The network lifetime is calculated by the initial energy of single-radio nodes IE, and the maximum energy consumption among the bottleneck nodes per time unit.

$$L = \min\{\frac{\text{IE}}{ec(v)} \mid v \in \{C_T(s) \mid s \in S'\}\}. \tag{2.8}$$

Maximizing the network lifetime is equivalent to minimizing the maximum number of descendants of bottleneck nodes in the routing tree rooted at each $s \in S'$. In other words, the joint optimization problem is then to identify a $S' \subseteq S$ with the minimum cardinality $|S'|$ such that each node is no more than h hops from one of the sinks in S' and meanwhile the maximum number of descendants of bottleneck nodes in the routing tree rooted at each $s \in S'$ is minimized. To solve this problem, we need to jointly address the following two challenges: (i) determining the optimal number of sinks and their locations among S for placement so that any node is able to reach at least one sink within h hops, and (ii) devising a tree-based routing protocol for data transmission from nodes to sinks such that the network lifetime is maximized.

The h-hop constrained multiple sink placement problem is NP-hard. Con-

sidering one of its special cases where $h = 1$ and there is no restriction on the potential sink locations, the problem becomes the Unit Disk Covering Problem (UDCP) that aims to find the minimum number of disks to cover all sensors in the network [44]. Assume that the radius of the disk (sink) is identical to the transmission range of sensors and a sensor is covered by a disk if they are within the transmission range of each other. Since the decision version of UDCP is NP-complete [44], the problem of concern in this section is NP-complete, too.

Due to the difficulty of jointly determining the optimal number of sinks and devising a routing protocol for the maximum network lifetime, we propose a heuristic instead. We decompose the problem into two sub-problems: finding the optimal number of sinks and their locations such that each node can reach a sink with no more than h hops; and under such a sink placement, finding a load-balanced forest to maximize the network lifetime, in which each sink is the root of a routing tree with the depth no more than h. Specifically, the heuristic first calculates the set of nodes covered by a sink at each potential location subject to the given h-hop constraint. It then identifies a subset of sinks and their locations with the minimum cardinality, covering all nodes in the network. It finally constructs load-balanced routing trees rooted at the chosen sinks for energy-efficient data gathering such that the network lifetime is maximized. We detail the heuristic as follows.

2.4.1 The Optimal Number of Sinks

We first find the minimum number of locations in S to ensure that all nodes are within h hops from at least one of these locations. For a potential sink location $s \in S$, let $N_1(s) = \{u \mid (u,s) \in E, u \in V\}$ be the set of neighboring nodes of sink s and $N_h(s)$ be the set of nodes within h hops from sink s, i.e., $N_h(s) = \{v \mid \text{the number of hops from } v \text{ to } s \text{ is no greater than } h\}$. The calculation of

$N_h(s)$ for each $s \in S$ is as follows. A Breadth-First-Search (BFS) tree rooted at s is constructed, which is expanded layer by layer. The expansion will terminate when it reaches layer h. The set of nodes contained in this BFS tree is $N_h(s)$. Let $\mathcal{C} = \{N_h(s) \mid s \in S\}$ be the collection of sets derived by the set S.

The problem of placing the optimal number of sinks at locations in S such that each node can reach one of the chosen sinks with no more than h hops is equivalent to finding a sub-collection $S' \subseteq S$ such that $|S'|$ is minimized and $\bigcup_{s \in S'} N_h(s) = V$. This is a set cover problem, which is NP-complete [22]. Instead, a greedy heuristic will be employed and it delivers an approximate solution to the problem with the approximation ratio of $O(\log B)$, where $B = \max_{s \in S'}\{|N_h(s)|\} \leq n$. For convenience, a node v is referred to be *covered* by a sink s if the number of hops from v to s is no more than h; otherwise, v is *uncovered* by s. If a given node v cannot be covered by any sink in S', then the node is uncovered. The proposed algorithm proceeds iteratively. Initially, all nodes in V are uncovered and the set of chosen sinks S' is empty. The algorithm iteratively selects a sink s such that the set $N_h(s)$ from $\mathcal{C}$ covering as many uncovered nodes as possible. Once a set $N_h(s)$ is chosen, it will be removed from $\mathcal{C}$. The sink s and its current location will be added to set S'. The algorithm continues until all nodes in V are covered by the sinks in S'. The proposed algorithm for finding the optimal number of sinks is referred to as `Find_Optimal_Sink`, and its detailed description is given in **Algorithm 2**.

2.4.2 Energy-efficient Data Routing

Having placed sinks at S', we now devise an energy efficient tree-based routing protocol for data collection to maximize the network lifetime. Following Eq. (2.8), maximizing the network lifetime is equivalent to minimizing the maximum energy consumption among bottleneck nodes, while the energy consumption of each bottleneck node is proportional to the number of

Algorithm 2: `Find_Optimal_Sink`

Input : The set of nodes V, the set of potential sink locations S
Output: The set of chosen locations for sink placement S'

$U \leftarrow V$;
$\mathcal{C} \leftarrow \{N_h(s) \mid s \in S\}$;
$S' \leftarrow \emptyset$; /* the set of chosen sinks */;
while $U \neq \emptyset$ **do**
  Select a set $N_h(s) \in \mathcal{C}$ such that $|U \cap N_h(s)|$ is maximized;
  $S' \leftarrow S' \cup \{s\}$;
  $U \leftarrow U - N_h(s)$;
  $\mathcal{C} \leftarrow \mathcal{C} - \{N_h(s)\}$;
return S'.

its descendants in the routing tree rooted at a chosen sink. In other words, the optimization objective is to group nodes into different clusters headed at different sinks and make each node belong to one cluster only. For each cluster, a load-balanced routing tree rooted at the cluster head (a sink) will be built, such that (i) the number of hops from each node to its tree root is no more than h; and (ii) the maximum number of descendants among the bottleneck nodes is minimized. We refer to this clustering problem as the *load-balanced forest problem*, which can be approximately solved by the following three steps.

Partitioning n nodes into h disjoint subsets. The $|S'|$ sinks are compressed into a *virtual super node* v_s, and every neighbor of a chosen sink in the original network now becomes a neighbor of the virtual node. A BFS tree rooted at v_s in the modified network is then constructed and as a result, the nodes in the network are partitioned into h disjoint subsets, according to the number of hops from each node to v_s. Let V_i be the set of nodes in layer i, then $\bigcup_{i=1}^{h} V_i = V$ and $V_i \cap V_j = \emptyset$, where $1 \leq i, j \leq h, i \neq j$. Note that V_0 contains only the root v_s and V_1 contains only the bottleneck nodes.

Finding a load-balanced tree rooted at each sink. Finding an optimal load-balanced tree has been shown NP-complete [59]. We here adopt a heuristic proposed in [59] for the load-balanced tree construction. The heuristic is a

greedy algorithm, which expands the tree layer by layer in a top-down fashion. Assuming a partial load-balanced tree spanning the nodes from layer 0 to layer $1 \leq l < h$ has been constructed, we now expand the tree by including the nodes in layer $l+1$ as follows.

We first construct a node-weighted bipartite graph $G_l = (X, Y, E_l, w)$, where the nodes in V_l are grouped into different subsets according to their ancestors in V_1, i.e., nodes in the same subset are the descendants of the same node in V_1. Let $X = \{x_1, x_2, \ldots, x_{|X|}\} \subseteq V_1$ be the set of ancestors of the nodes in V_l that are incident to nodes in layer $l+1$, and Y be the set of nodes in layer $l+1$, i.e., $Y = V_{l+1}$. For each $x \in X$, its weight $w(x)$ is the number of descendants of x in the current tree. And each node $y \in Y$ is assigned a weight $w(y) = 1$. E_l is the set of edges consisting of (x, y) if $x \in V_1$ is the ancestor of a node $v \in V_l$ and $(v, y) \in E$. The load-balanced tree problem then is to choose a node $x \in X$ as the ancestor for every node $y \in Y$ such that the maximum number of descendants of nodes in V_1 in the resulting tree is minimized.

We then transform the problem into a maximum flow problem in an auxiliary flow network N_l by assigning its links with different capacities dynamically, where $N_l = (X \cup Y \cup \{s, t\}, E'_l \cup \{\langle s, x\rangle \mid x \in X\} \cup \{\langle s, y\rangle \mid y \in Y\} \cup \{\langle x, t\rangle \mid x \in X\}, c)$ is a directed flow network derived from G_l, assuming s is a source node and t is a destination node. Directed edges from s to $y \in Y$ and s to $x \in X$ are associated with capacity $c(s, y) = 1$ and $c(s, x) = w(x)$ respectively. The directed edge from y to x $\langle y, x\rangle \in E'_l$ has capacity $c(y, x) = 1$ if edge $(x, y) \in E_l$. The capacity of the directed edge from each $x \in X$ to t, L_m, is the maximum load among the nodes in X, that is, $c(x, t) = L_m$. The value range of L_m is within the interval $[\max_{1 \leq i \leq |X|}\{w(x_i) \mid x_i \in X\}, \max_{1 \leq i \leq |X|}\{w(x_i) \mid x_i \in X\} + |Y|]$. Given a value of L_m, we apply the maximum flow algorithm to N_l to find a flow f from s to t and check whether $|f| = \sum_{1 \leq i \leq |X|}\{w(x_i) \mid x_i \in X\} + |Y|$. If yes, it delivers a feasible solution, we will check whether it still has a feasible solution by decreasing the value of L_m; otherwise, the value of

L_m needs to be increased. The optimal value L_{opt} of L_m can be found through binary search. In the end, every node y in layer $l+1$ will be assigned an ancestor $x \in X$ if $f(y,x) = 1$. The proposed maximum flow algorithm thus can be applied at most $\lceil \log |V_{l+1}| \rceil$ times to find the optimal load L_{opt} for the current tree expansion.

The partial load-balanced tree is then expanded by including the nodes in layer $l+1$ as follows. For each node $y \in V_{l+1}$, node $v \in V_l$ becomes its parent if v is a descendant of $x \in X$, $(v,y) \in E$, and $f(y,x) = 1$. As a result, the partial load-balanced tree is expanded upto layer h. It is straightforward that the approximate load-balanced tree rooted at the virtual node v_s is no more than h layers.

Finding a load-balanced forest. The load-balanced forest consisting of $|S'|$ load-balanced trees rooted at the $|S'|$ chosen sinks is constructed as follows. A bipartite graph $G_B = (S', V_1, E')$ is constructed, where $V_1 = \{C_T(s) | s \in S'\}$ and an edge $(s,v) \in E'$ if sink $s \in S'$ is within the transmission range of node $v \in V_1$. A maximum matching in G_B is then found. For each matched $v \in V_1$, there is a matched edge with $s \in S'$ as the other endpoint. For each unmatched node $v \in V_1$, if there are multiple edges in E' incident to v, one of the edges is arbitrarily chosen and the other endpoint of the chosen edge is a sink $s \in S'$. For both cases, s is the root of a load-balanced tree and the subtree rooted at v in the original load-balanced tree will be part of this new tree. As a result, the n nodes have been partitioned into $|S'|$ load-balanced trees rooted at the $|S'|$ chosen sinks, and each node can reach its root (a sink) within h hops.

The proposed heuristic consists of algorithm `Find_Optimal_Sink` for sinks identification, and the algorithm finding load-balanced routing forest. For convenience, we refer to the heuristic for the h-hop constrained multiple sink placement problem as `Heuristic_Opt_MultiPlace`, or `HOMP` for short.

Theorem 2.2. *For a single-radio sensor network $G(V,E)$ with S as the potential sink locations, and h as the hop upper bound between each node-sink pair, the h-hop*

constrained network lifetime maximization problem can be solved by algorithm HOMP *with complexity* $O(n^3) + O(|E|n \log n)$*, where* n *is the number of nodes in* G*.*

Proof. In algorithm `Find_Optimal_Sink`, calculating the collection of sets $\mathcal{S}$ takes $O(|E||S|)$, the number of iterations is bounded by $\min(n, |S|) = |S|$, and the loop body can be implemented in time $O(n|S|)$. Thus the time complexity of algorithm `Find_Optimal_Sink` is $O(|E||S|) + O(n|S|^2) = O(n^3)$. And the time complexity of the proposed algorithm for finding a load-balanced forest is $O(|E|n \log n)$, because it takes $O(|E| + n)$ time to partition nodes, and $O(|E|n \log n)$ time to build load-balanced trees $O(|E|n \log n)$ [59]. The complexity of finding a load-balanced forest is $O(|E|n)$ [22]. In summary, the the complexity of algorithm HOMP is $O(n^3) + O(|E|n \log n) + O(|E|n) = O(n^3) + O(|E|n \log n)$. □

2.5 Performance Evaluation

In this section, we evaluate the proposed algorithms for multiple sink assignment and placement by simulation. We vary the constraint parameters and investigate their impacts on the network performance. We also compare the proposed algorithms with other comparable strategies.

2.5.1 Experiment Settings

We consider a wireless sensor network randomly deployed in a $1000m \times 1000m$ square region. The transmission range r of the low-power radio is fixed to be 100 meters and the initial energy capacity of each sensor node is $IE = 200Joules$. We adopt the energy consumption parameters of `CC2420` radio [2], a typical 3G radio `MO6012` [5] based on WCDMA 2100@24dBm standard, and the `NAND` flash memory [1] for P_l, P_h, E_o, and P_b respectively. We assume that the data generation rate of each sensor is $r_g = 1bit/s$. To avoid experimental

results being dependent on a specific network topology, for each network size n, 50 topologies are generated using the NS-2 simulator [4], and each value in figures is the mean of the results by applying the mentioned algorithm to the 50 different network topologies.

2.5.2 Evaluation of Multiple Sink Assignment Algorithm

We first evaluate the performance of the proposed multi-sink assignment algorithm `Dynamic_Alg`. Specifically, we provide a snapshot of the network lifetime by studying the energy consumption of individual nodes as the network operates. We then investigate the network lifetime by varying constraint parameters, including network throughput threshold α, round duration τ, number of sinks m, and data delay threshold D. In the default setting, the extra energy consumption for routing re-construction and control message communication is $e_\Delta = 0.2 Joules$. We finally compare network lifetime delivered by different algorithms and show the effectiveness of `Dynamic_Alg`.

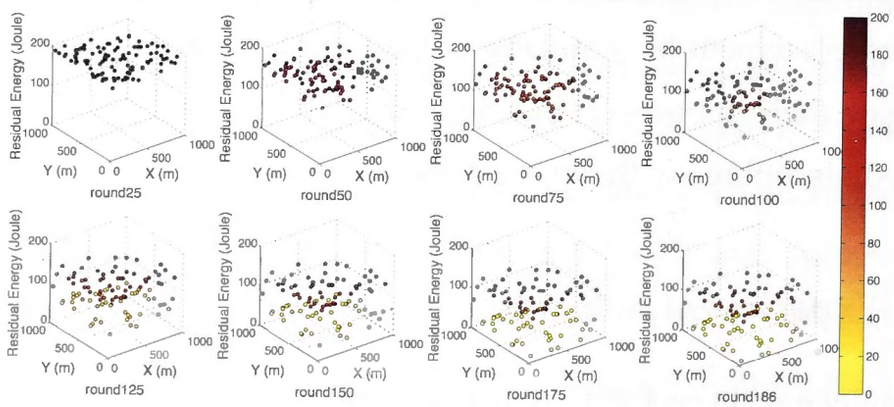


Figure 2.3: Residual energy at nodes in different rounds with $n = 100$, $m = 5$, $\alpha = 0.7$, $D = 1$ *hour* and $\tau = 2$ *hours*

We examine the residual energy at nodes after each round by fixing $n = 100$, $m = 5$, $\alpha = 0.7$, $D = 1$ *hour* and $\tau = 2$ *hours*. With this setting, there are 186 rounds in total within the network lifetime ($1.34 \times 10^6 seconds$). Fig. 2.3 shows the residual energy of individual nodes after every 25 rounds as well as

the last round. It is observed that in the first 75 rounds, all nodes have more than 50% of the initial energy left. With the increase in the number of rounds, nodes have less and less residual energy. In the 175 round, 44 nodes have less than 20% of initial energy left. In the last round, 37 nodes run out of energy, the throughput requirement is no longer met, and network lifetime ends.

We then investigate the impacts of constraint parameters on network lifetime.

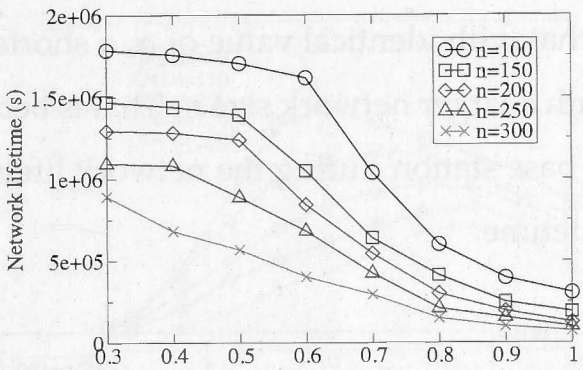


Figure 2.4: Network lifetime with different network throughput threshold α, when $m = 5$, $D = 1$ *hour*, and $\tau = 2$ *hours*

Impact of network throughput threshold α on network lifetime. We start by varying the network throughput threshold α from 0.3 to 1 with the increment of 0.1, while fixing $n = 100$, $m = 5$, $D = 1$ *hour*, and $\tau = 2$ *hours*. Fig. 2.4 illustrates that with the increase of α, the network lifetime decreases steadily before $\alpha = 0.6$, and goes down rapidly after that. This is because a small value of α does not always imply a small number of active nodes. For example, when $\alpha = 0.3$, at least $100 \times 0.3 = 30$ active nodes should be selected as active nodes but they may be in more than $m = 5$ connected components. In that case, some *extra nodes* need to be added in the active node set so that all active nodes are in at most $m = 5$ connected components (meet the m-component constraint). As such, the number of active nodes with different α (0.3, 0.4, 0.5, and 0.6 in Fig. 2.4) could be similar and the resultant network lifetime does not change much. When m is larger than 0.6, the greater the value of α, the larger

the number of active nodes required, and the larger the per time unit energy consumption at nodes, resulting in a shorter network lifetime.

Note that when the node density n increases upto 300 with the stepping of 50, the similar tendency of the network lifetime is obtained, shown in Fig. 2.4. However, when the value of n is relatively large, the curve of network lifetime L starts its dramatic decreasing from a smaller α. The reason is that with the same value of α, the higher the node density, the less extra nodes required to meet the m-component constraint, and the more distinct the impact of α on L. We can also see that with identical value of α, a shorter network lifetime L will be delivered with a larger network size n. That is because data from more nodes is sent to the base station during the network lifetime, which results in a shorter network lifetime.

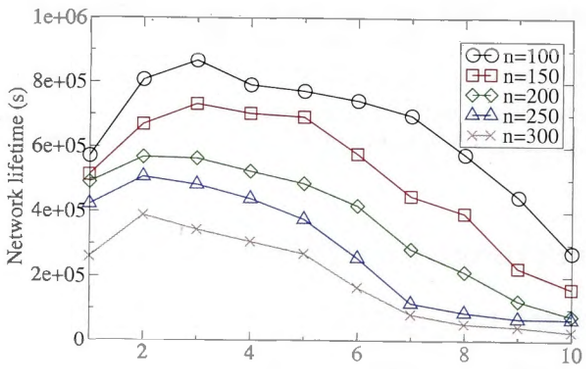


Figure 2.5: Network lifetime with different round duration τ, when $m = 5$, $\alpha = 0.7$, and $D = 1$ *hour*

Impact of round duration τ on network lifetime. We next investigate the network lifetime with different values of round duration τ. We vary τ from 1 *hour* to 10 *hours* with the increment of 1, while keeping $m = 5$, $\alpha = 0.7$, and $D = 1$ *hour*. Fig. 2.5 plots the impact of τ on L under different network sizes n. Generally, the larger the value of τ, the shorter the network lifetime L. This is due to the fact that frequent identification of sinks and active nodes results in better energy balance among nodes in the network. However, note that when τ increases from 1 *hour* to 2-3 *hours*, the value of L slightly increases. The rea-

son is that excessively frequent execution of the routing forest establishment algorithm leads to extra energy consumption and shortens the network lifetime. Thus, it indicates that the appropriate frequency of algorithm execution is significant to the network lifetime. Also, from Fig. 2.5 it can be observed that with the fixed τ, the network lifetime is smaller as the network size n goes up. That is because by using a fixed number of sinks, the energy consumption is more evenly distributed in relatively small size networks.

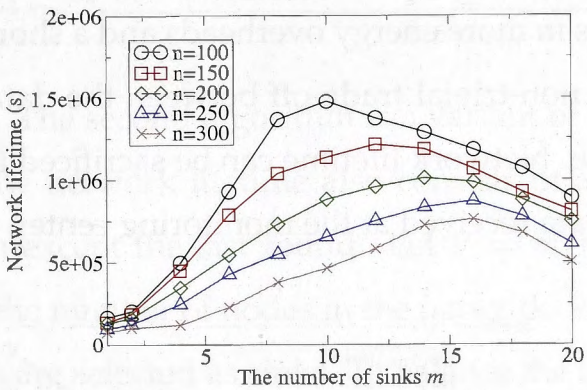


Figure 2.6: Network lifetime with different number of sinks m, when $\alpha = 0.7$, $D = 1$ *hour*, $\tau = 2$ *hours*

Impact of number of sinks m on network lifetime. We now vary the number of sinks m and evaluate its impact on network lifetime. Given the network size n, we fix $\alpha = 0.7$, $D = 1$ *hour*, $\tau = 2$ *hours*, and increase m from 2 to 20 (stepping of 2). The results are shown in Fig. 2.6, from which we can see that with a fixed number sinks, a longer network lifetime will be delivered in a smaller size network. It is interesting to notice that the increase in the number of sinks does not necessarily prolong the network lifetime. With a fixed n and increasing m, the network lifetime first increases and then decreases. Curves with different n have turning points with different values of m. Before these points, L rises dramatically as m increases, because more sinks are able to better balance the energy consumption among nodes. However, if m keeps increasing after a certain value, the network lifetime starts dropping. That is because the increase of m causes more energy consumed on high-bandwidth

radios, compromising the network lifetime. The experimental results reveal that it is not always profitable to use a large number of sinks for the purpose of a longer network lifetime.

Impact of delay threshold D on network lifetime. We then evaluate the impact of data delivery delay threshold D on network lifetime by fixing $m = 5$, $\alpha = 0.7$, $\tau = 2$ *hours*, and setting D to 10 *minutes*, 20 *minutes*, 30 *minutes*, 60 *minutes*, and 120 *minutes*. Fig. 2.7 shows that a smaller value of D leads to a shorter network lifetime because frequent on-and-off switching of the 3G radios on sinks results in more energy overheads and a shorter network lifetime. This also implies a non-trivial trade-off between the data delivery delay and the network lifetime. Network lifetime can be sacrificed for a shorter delivery delay and fresher data received at the monitoring center, and vice versa.

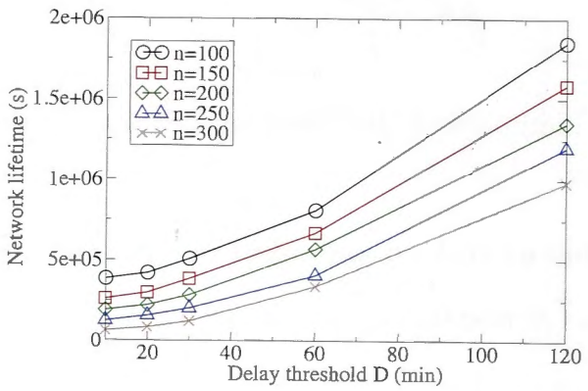


Figure 2.7: Network lifetime with different data delivery delay threshold D, when $m = 5$, $\alpha = 0.7$, $\tau = 2$ *hours*

Performance comparison. We finally compare the performance of algorithm `Dynamic_Alg` against those of another two algorithms. The first algorithm randomly selects m nodes as the sinks from all deployed nodes at the beginning of the network lifetime, and builds trees rooted at these m sinks to span other nodes using Breadth-First Search, until at least $\lceil \alpha \cdot N \rceil$ nodes are included in these trees. The m routing trees are used for data collection until the first node in the trees dies. We refer to this algorithm as `Static_Alg`. `Static_Alg` is executed only once in the network lifetime and is easy to im-

Table 2.1: Network lifetime (*second*) delivered by three algorithms with different m and τ

(a) $m = 8$, τ varies from 1 to 5 *hours*

Algorithm	$\tau = 1$ *hour*	$\tau = 2$ *hour*	$\tau = 3$ *hour*	$\tau = 4$ *hour*	$\tau = 5$ *hour*
`Static_Alg`	100250	100250	100250	100250	100250
`LEACH_Alg`	164469	184469	194469	144469	114469
`Dynamic_Alg`	1164870	1364870	1494870	1304870	1134870

(b) $\tau = 3$ *hour*, m varies from 4 to 12

Algorithm	$m = 4$	$m = 6$	$m = 8$	$m = 10$	$m = 12$
`Static_Alg`	51373	71250	100250	125250	107132
`LEACH_Alg`	118689	166333	194469	224469	184469
`Dynamic_Alg`	580936	1282191	1494870	1604870	1434870

plement in practice. The second algorithm is a variant of `LEACH` [40]. Similar to `Dynamic_Alg`, the network lifetime also consists of a number of rounds with equal duration except the last round. Let $P = m/n$ be the ratio of the number of sinks to the number of nodes in the network. Within each round, P percentage of nodes are selected as sinks. To balance the energy consumption overall the network, nodes serving as sinks in the current round cannot be selected as sinks for the next $1/P$ rounds. Denote by nodes which are qualified to become sinks the *sink candidates*. Within each round, each sink candidate has a probability P to become a sink, and m nodes will be identified as sinks. The m sinks then include at least $\lceil \alpha \cdot n \rceil - m$ nodes from the rest nodes to form m trees rooted at these sinks with the objective to minimize the total distance between nodes and corresponding sinks. The residual energy on nodes is updated as the network operates and the network lifetime ends when sinks are no longer able to transmit data from at least $\lceil \alpha \cdot n \rceil$ nodes to the third party network. This algorithm is referred to as `LEACH_Alg`.

We evaluate the network lifetime delivered by the three mentioned algorithms by fixing $n = 100$, $D = 1$ *hour* while varying the values of m and τ. We first fix the value of m to be 8 and vary the value of τ from 1 to 5 *hours* with the increment of 1 *hour*. The resultant network lifetime is listed in Table 2.1 (a). The network lifetime delivered by `Static_Alg` stays the same with different val-

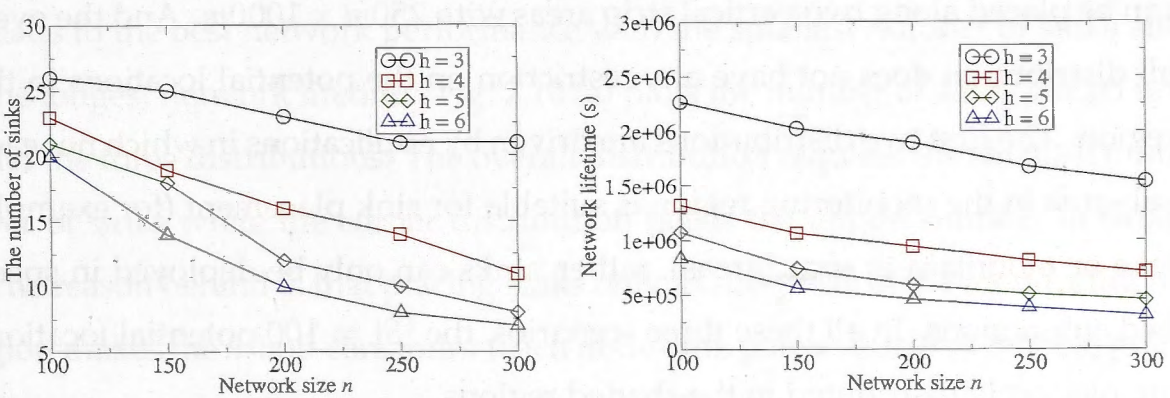
ues of τ, because `Static_Alg` is executed only once throughout the network lifetime and the network lifetime is irrelevant to the variety of τ. However, the network lifetime by both `Dynamic_Alg` and `LEACH_Alg` varies as the value of τ increases: it first rises then goes down. That is because of the impact of τ on the network lifetime, as discussed previously. We then keep the value of τ 3 *hour* and increase m from 4 to 12 with stepping of 2. The results are shown in Table 2.1 (b), from which we can see that with the increase in the value of m, the network lifetime delivered by these three algorithms first goes up (till $m = 10$) and then decreases, due to the impact of m on the network lifetime analyzed in the previous subsection.

In general, `Dynamic_Alg` always outperforms the other two algorithms while `LEACH_Alg` always outperforms `Static_Alg`. Averagely, the network lifetime delivered by `Dynamic_Alg` is about 7 times and 12 times as long as those by `LEACH_Alg` and `Static_Alg`. The superiority of algorithms `Dynamic_Alg` and `LEACH_Alg` over `Static_Alg` lies in more balanced energy consumption among nodes by periodically re-selecting sinks and active nodes. Whereas `Static_Alg` keeps the same set of sinks as well as unchanged set of active nodes throughout the network lifetime, causing energy bottleneck at sinks and ending the network lifetime much sooner. The advantages of `Dynamic_Alg` over `LEACH_Alg` include the following two aspects. The first one is its more efficient sink identification strategy. `LEACH_Alg` avoids nodes being re-selected as sinks for at least $1/P$ rounds. However, this cannot guarantee that sink candidates are all with high residual energy. If nodes with low residual energy are selected as sinks, more severe energy unbalance will be caused in the network. While `Dynamic_Alg` first identifies active nodes with relatively high residual energy, and then selects sinks from active nodes to achieve the most evenly distributed energy consumption. The second reason is the more advanced routing forest establishment approach in `Dynamic_Alg`. The total distance minimization in `LEACH_Alg` does not necessarily lead to the

minimum energy consumption. Whereas `Dynamic_Alg` builds a routing forest to maximize the minimum residual energy of active nodes, which leads to a better balance in energy consumption among nodes.

2.5.3 Evaluation of Multiple Sink Placement Algorithm

We now study the performance of the multi-sink placement algorithm `HOMP` by evaluating the impacts of the upper bound on the number of hops from each node to its nearest sink, the distribution of potential sink locations, and the size of monitoring region on the network performance. We also compare the performance of `HOMP` with another heuristic to validate its effectiveness. In the default setting, the potential sink locations in S are randomly distributed overall the monitoring region with $|S| = 100$.



(a) Impact of h on the number of sinks

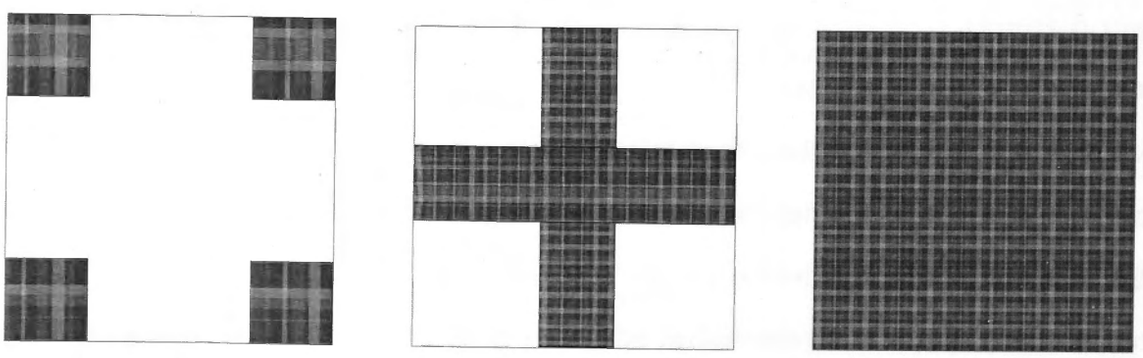
(b) Impacts of h on network lifetime

Figure 2.8: Impact of h on the network performance when $|S| = 100$

Impact of h on network performance. We first investigate the network performance by varying the values of h and n. Fig. 2.8 shows that the smaller the values of h and n, the larger the number of sinks placed, and the longer the network lifetime. With the increase in either h or n, the number of sinks placed would decrease, and so is the network lifetime. The reason is that when fixing the network size n and increasing h, each sink can cover more nodes and a smaller number of sinks will be needed to cover all the nodes. Accordingly,

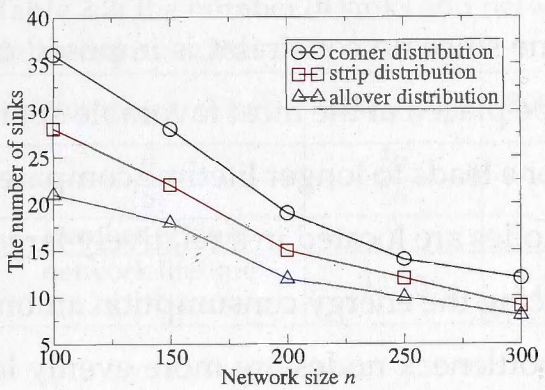
in a routing tree rooted at the sink, a heavier load is allocated at each child of a sink, and the network lifetime is shortened. Whereas with constant h and growing n, the network density increases, thus requires fewer sinks to cover all nodes, followed by a shorter network lifetime.

Impact of the distribution of potential sink locations on network performance. We next study the distribution of the potential sink locations S to the network performance by varying its distribution and size. Due to the physical limitations in the real networks, their locations may be constrained in specified areas in the monitoring region. We study the network performance under different distributions of the sink potential locations by adopting three distributions [56] shown as the shaded areas in Fig. 2.9, and fixing $h = 5$, $|S| = 100$. In the corner distribution, sinks are only allowed to be placed in four corner areas of $250m \times 250m$ in the monitoring region. In the strip distribution, sinks can be placed along two vertical strip areas with $250m \times 1000m$. And the overall distribution does not have any restriction on the potential locations in the region. The first two distributions are driven by applications in which not each sub-area in the monitoring region is suitable for sink placement (for example, lake or mountain in some areas), rather, sinks can only be deployed in specified sub-regions. In all these three scenarios, the $|S| = 100$ potential locations are randomly distributed in the shaded regions.

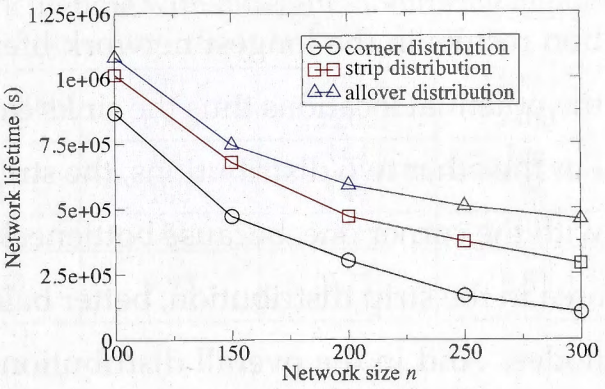


(a) Corner distribution (b) Strip distribution (c) Overall distribution

Figure 2.9: Three different distributions of potential sink locations



(a) Impact of distributions of S on the number of sinks

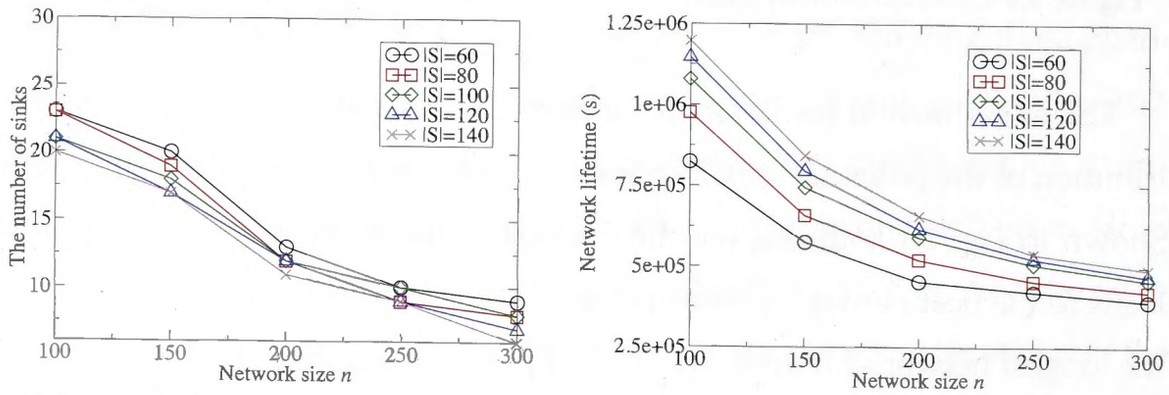


(b) Impact of distributions of S on the network lifetime

Figure 2.10: Impact of distributions of S on the network performance when $h = 5$

The experimental results indicate that less constraint imposed on the distribution of the potential sink locations provides better network performance. Shown in Fig. 2.10, among the three S distributions, the overall distribution leads to the best network performance with the smallest number of sinks and the longest network lifetime. Fig. 2.10 (a) plots the number of sinks placed under the three distributions. The overall distribution requires the smallest number of sinks while the corner distribution needs the largest number of sinks. The reason behind is that placing sinks only in sub-areas in the monitoring region makes the h-hop constraint (each node-sink pair within $h = 5$ hops) more difficult to meet. An extreme example is to distribute the $|S| = 100$ potential sink locations only in one of corner area of $250m \times 250m$ and the $n = 100$ nodes are uniformly distributed in the monitoring region. Under this setting, no solution will be found since nodes in the diagonal corner cannot reach any sink within 5 hops, no matter how many sinks are deployed. In general, the smaller the distribution area, the greater the number of sinks required. Also, notice that the gap among the three distribution is obvious when $n = 100$ yet becomes smaller with the increase in n and is tiny when $n = 300$. It verifies that in a network with high node density, the h-hop constraint is easier to meet with relatively a smaller number of sinks. Fig. 2.10(b) plots the network

lifetime under the three distributions. Not surprisingly, the overall distribution results in the longest network lifetime since no constraint is imposed on the potential locations thus the sinks can be placed at the most favorable spots. For the other two distributions, the strip one leads to longer lifetime compared with the corner one, because bottleneck nodes are located in a relatively larger area in the strip distribution, better balancing the energy consumption among nodes. And in the overall distribution, bottleneck nodes are more evenly located throughout the network, contributing to the longest lifetime.



(a) Impact of $|S|$ on the number of sinks

(b) Impact of $|S|$ on the network lifetime

Figure 2.11: Impact of the size of S on the network performance when $h = 5$

We then vary the size of S from 60 to 140 with the increment of 20, while keeping $h = 5$, and overall distribution being adopted. Shown in Fig. 2.11(a), the number of sinks needed with different sizes of S varies slightly but larger $|S|$ does lead to a longer network lifetime, as illustrated in Fig. 2.11(b), due to the larger searching space for the sink placement. Opposite to our initiative guess that more sinks would be placed with a larger size of S, the numbers of sinks delivered are almost the same with different values of $|S|$. This is because the number of sinks needed mostly depends on the nodes density, rather than the potential locations density. Also, it is interesting to observe in Fig. 2.11(b) that the superiority of a larger $|S|$ in terms of network lifetime becomes less evident as the network size n increases. The reason is that in a dense network,

Table 2.2: the number of sinks and network lifetime with different monitoring regions delivered by HOMP

monitoring area (m^2)	600×600	800×800	1000×1000	1200×1200	1400×1400
n	36	64	100	144	196
$\|S\|$	36	64	100	144	196
number of sinks	9	14	20	29	38
network lifetime ($10^5 s$)	13.20	11.70	9.93	9.21	8.81

the number of potential sink locations plays a less important role in lifetime prolongation. While in a relative sparse network (e.g., $n = 100$), more options for sink placement potentially deliver a much longer network lifetime.

Impact of monitoring region size on network performance. We now evaluate the network performance by varying the monitoring region size, while keeping the density of nodes and potential sink locations unchanged. We fix $h = 5$ and vary the monitoring area from $600m \times 600m$ to $1400m \times 1400m$, which means that the network size n and the number of potential sink locations $|S|$ will increase accordingly at the same rate.

Table 2.2 illustrates that the larger the monitoring region size, the greater the number of sinks placed. It verifies that with the same node density, more sinks are required to meet the h-hop constraint. It also shows that the network lifetime decreases with the increase of the monitoring region size. The reason behind is that despite of the increase in the number of sinks, each bottleneck node still undertakes more relay workload, which causes a shorter network lifetime.

Performance comparison. We compare the performance of the proposed heuristic against the BFS tree-based heuristic in terms of network lifetime. Recall that the results delivered by algorithm HOMP include S' for sink placement and load-balanced trees rooted at the $|S'|$ chosen sinks. For the BFS-tree based heuristic, we assume that its first two stages are identical to those of algorithm HOMP. The only difference lies in the routing protocol design: in-

stead of building a load-balanced tree, a BFS tree rooted at the virtual node will be built and $|S'|$ trees will be obtained by removing the virtual node and its incident edges from the built tree. We refer to this variant as algorithm `BFS_Heuristic_Opt_MultiPlace`, or `BFS_HOMP` for short.

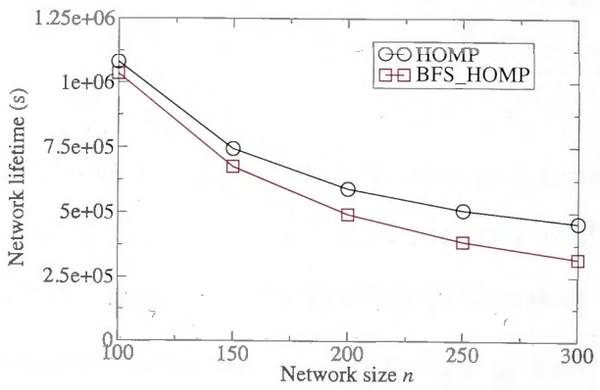


Figure 2.12: Performance evaluation between algorithms `HOMP` and `BFS_HOMP`, when $h = 5$ and $|S| = 100$

To evaluate the performance of algorithms `HOMP` and `BFS_HOMP`, we vary n from 100 to 300 while fixing $h = 5$ and $|S| = 100$. Fig. 2.12 implies that with the increase of the network size n, the network lifetime delivered by either algorithm `HOMP` or algorithm `BFS_HOMP` decreases, because the bottleneck nodes have to relay more data for other remote nodes. It is also shown that in terms of network lifetime, algorithm `HOMP` performs 13% better than algorithm `BFS_HOMP` on average since the former distributes data relay workload among bottleneck nodes more evenly. With the increase of network size n, the gap between the network lifetime delivered by these two algorithms becomes larger.

2.6 Conclusions

In this chapter, we have studied the network lifetime prolongation problem by using multiple sinks for data collection in two types of sensor networks. Specifically, we first formulated the network throughput guaranteed network

lifetime maximization problem in homogeneous WSNs, where all nodes act as sinks alternately. We provided an approach to identify the sinks dynamically and route data from selected nodes to these sinks energy-efficiently to prolong the network lifetime while guaranteeing the required network throughput. We then investigated the multiple sink placement problem in heterogeneous WSNs, assuming that sinks can only be placed at spots selected from given locations and each node is within h hops from at least one sink after the sinks are deployed. We addressed the problem by decomposing it into two sub-problems and solving them separately. We finally conducted extensive simulations to evaluate the performance of the proposed algorithms and study the impacts of constraint parameters on their performance. The experimental results showed that the proposed algorithms significantly improve network lifetime.

Chapter 3

Monitoring Quality Maximization with a Mobile Sink

3.1 Introduction

Mobile sinks are introduced to sensor networks mainly for the purpose of network lifetime prolongation. Though the mobile-sink WSN has been extensively studied and proven to be effective in balancing energy consumption among nodes, it does bring in new challenges, where data loss is one of the most urgent to be addressed, especially when high data fidelity is required. For example, in a sewage disposal system, where sensors are deployed along pipes for waste sensing and a sink moving with the sewage flow collects data from sensors, it is desirable to gather as many sensed data as possible and send them to the monitoring center for wastewater analysis. However, the non-stop moving of the sink and the limited data uploading speed of sensors result in that some data cannot be transmitted to the sink. In this chapter, we consider using a mobile sink in the WSN to collect sensed data of the monitoring region, and define monitoring quality as the criteria of how well the sink is able to collect data from the deployed WSN. We first provide system model and formulate monitoring quality. We then define the problem of communication time constrained monitoring quality maximization with NP-hardness proved, and propose a heuristic for it. We finally conduct extensive experiments by simulation to evaluate the effectiveness of the proposed algorithm.

The mobile sink used for data transmission from sensor nodes to the third party network is a dual-radio device with adequate storage and wireless antenna. It is usually installed on a shuttle, robot, or aircraft, which are petrol or electricity powered and need to be refused or recharged from time to time. The motion of the sink is typically constrained by the vehicle that it is attached to and the region which it moves in. First, the travelling distance of the mobile sink is constrained by the vehicle cruisingability (such as petrol/electricity, maneuverability). Second, the travelling of the mobile sink is constrained in a given map. In other words, the vehicle carrying the sink cannot travel freely in the monitoring region without restriction [101]. Instead, it is programmed to move with conditions, such as avoiding obstacles, e.g., ponds, bushes, buildings and fences. Third, the sink carried by a vehicle like a city commuter can only sojourn at given stops instead of any spot along the road. In some applications, specified locations must be visited by the sink at certain stages. For example, the routes of shuttle buses must include some stations as scheduled. Last, the speed of a mobile sink is bounded by multiple factors including traffic conditions, engine limitations, and speed restrictions on highways or in suburbs.

We consider a densely deployed sensor network consisting of *common nodes*, and a set of *gateways* deployed along roads in a given *road-map* in the monitoring region. Specifically, the gateways relay sensed data from common nodes to the mobile sink when the sink moves into their transmission ranges. The network is thus treated as three tiers: common nodes in the bottom tier, gateways in the middle tier, and the mobile sink in the top tier. Accordingly, data is transmitted from the bottom tier to the top tier in order. If all sensors are gateways, they upload their sensed data to the mobile sink in one-hop transmission mode. This achieves the most energy-efficient data collection by eliminating energy consumption on multi-hop routing, at the expense of a long data delivery latency because the sink has to visit individual nodes one by

one. On the other hand, if there is only one gateway in the entire network, it has to relay data for all the common nodes. Though time-efficient in data delivery it is, it becomes the traditional static sink structure which results in severe unbalanced energy consumption among sensor nodes. We adopt the hierarchical structure in this chapter for a desirable trade-off between energy conservation and data delivery latency.

The mobile sink moves without stop in the road-map, which falls into some real applications, e.g., the sink installed on an airplane following flight routes, or carried by a floater flowing with water. The sink visits selected gateways one by one for data collection. The amount of data collected by the sink from a gateway is constrained by their limited communication time: data can be uploaded from the gateway to the sink only when the sink is within the gateway transmission range. Thus, data loss occurs if the communication time between a gateway and the sink is shorter than the time required for the gateway to upload all its stored data [84,110]. And the amount of data loss depends on (1) the sink speed, (2) the length of the sink trajectory within the transmission range of the gateway, referred to as *intersection*, and (3) the data uploading rate of a gateway. A simplified scenario is illustrated by Fig. 3.1 where the road-map is a circle. As a result, not sensed data from all nodes are able to be successfully sent to the mobile sink in a single trip [98,101], and the nodes whose data are received by the sink are referred to as *active nodes*, while the others whose data are not received are *inactive nodes*. Consequently, the *communication time constrained data collection scenario* is proposed in the WSN with a mobile sink. Under this scenario, we focus on improving monitoring quality from two aspects. One is properly identifying the set of active nodes whose sensed data are the most representative among data generated from all nodes. The other is precisely estimating the unreceived data of inactive nodes by the collected ones of active nodes, based on the hypothesis that sensed data are highly spatially and temporally correlated, especially among neighboring sensors [97].

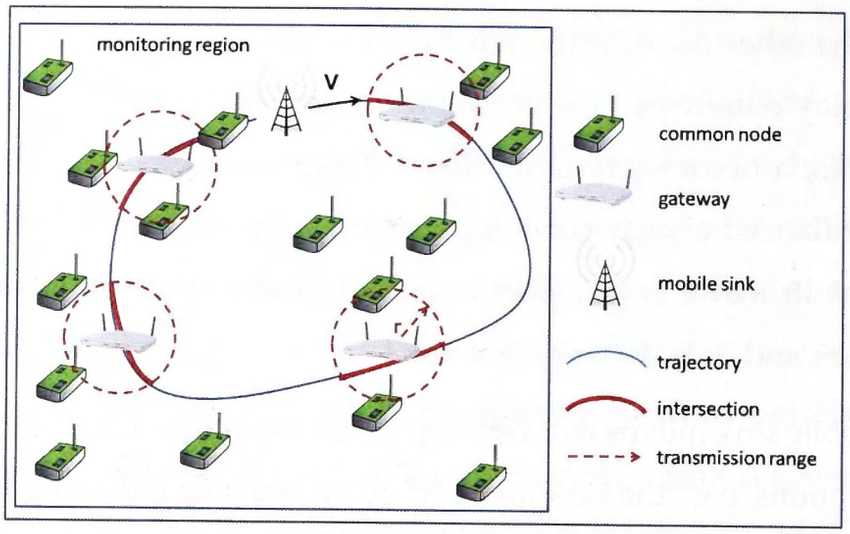


Figure 3.1: A hierarchical wireless sensor network with gateways, common nodes, and a mobile sink moving in a road-map

The remainder of this chapter is organized as follows. Section 3.2 formulates the system model and defines the problem to be solved, which is then proven to be NP-complete. Section 3.3 provides a three-stage algorithm for the problem. Section 3.4 uses extensive experimental results to evaluate the algorithm performance and indicate its effectiveness. Section 2.6 concludes this chapter.

3.2 System Model and Problem Definition

The system model of a heterogeneous WSN with a mobile sink moving in a given road-map is based on the following assumptions. (1) All common nodes are densely and randomly deployed in the monitoring region with identical initial limited energy capacities. (2) Gateways are deployed alongside the roads in the map with adequate buffer sizes for data storage. (3) Sensed data are transmitted from common nodes to the mobile sink via gateways only. (4) Gateways can always get recharged in time by the mobile sink directly or through the infrared ray, i.e., they are not energy-constrained. With the above

assumptions, sensed data are first generated by the common nodes, then transmitted to and stored temporarily at the gateways, and finally relayed to the mobile sink when gateways are individually visited. The heterogeneous network is denoted by $G(V \cup GW, E)$, where V is the set of common nodes with $n = |V|$, $GW = \{g_1, g_2, \ldots, g_m\}$ is the set of gateways with $m = |GW|$, and E is the set of links. The locations of nodes and gateways are fixed and known a *priori*, All nodes and gateways are equipped with the same omni-directional antenna, i.e., their transmission ranges are fixed and identical, denoted by r. There is a link between two nodes or a node and a gateway if they are within the transmission range of each other. The data generation rates of common nodes are identical and denoted by r_g, and the data uploading rate of gateway g_i is denoted by r_i, where $1 \leq i \leq m$. There is no data aggregation at gateways or common nodes.

The road-map is denoted by $G_M = (V_M, E_M)$, where E_M is the set of roads, $V_M = \{p_0, p_1, p_2, \ldots, p_{n_M}\}$ is the set of road crossings, and p_0 is the depot in the map. The mobile sink is not allowed to move off-road thus the trajectory to be found only contains roads in E_M. Accordingly, the sink collects data from individual roadside deployed gateways once moving into their transmission ranges, with the following two constraints imposed on its motion. (1) In each single tour, the sink starts off from the depot p_0 and returns to p_0 in the end. Thus the trajectory should be closed with depot p_0 as its starting and ending points. (2) The sink is able to travel no longer than D_{max} in each tour, where D_{max} is a pre-defined parameter, depending on the cruisingability of the mobile sink. That is, the length of the found trajectory should not be greater than D_{max}. The duration of each tour is no longer than D_{max}/v, where v is the constant speed of the sink, determined by the application tolerant data delivery latency.

Since a gateway g_i is only able to communicate with the sink when the sink is within its transmission range, the maximum amount of data that gateway g_i

can upload to the sink in one tour is

$$data(g_i) = \frac{l_i}{v} \cdot r_i,\ 1 \leq i \leq m, \tag{3.1}$$

where l_i is the length of the intersection of the sink trajectory within the transmission range of g_i. Define the *quota* of g_i as the maximum number of active nodes that can transmit their data to the sink via gateway g_i in one tour. It is denoted by $c(g_i)$ and formulated as follows.

$$c(g_i) = \lfloor \frac{data(g_i)}{\frac{L}{v} \cdot r_g} \rfloor = \lfloor \frac{l_i \cdot r_i}{v \cdot \frac{L}{v} \cdot r_g} \rfloor = \lfloor \frac{l_i \cdot r_i}{L \cdot r_g} \rfloor,\ 1 \leq i \leq m. \tag{3.2}$$

From Eq (3.2), we conclude that it is necessary to identify a subset of V as the active nodes if $\sum_{i=1}^{m} c(g_i) < n$, which means not sensed data of all nodes can be collected by the mobile sink. Let V' be the set of active nodes. After V' is identified, the sink will only collect data from them via gateways while data from inactive nodes are discarded, as shown in Fig. 3.2.

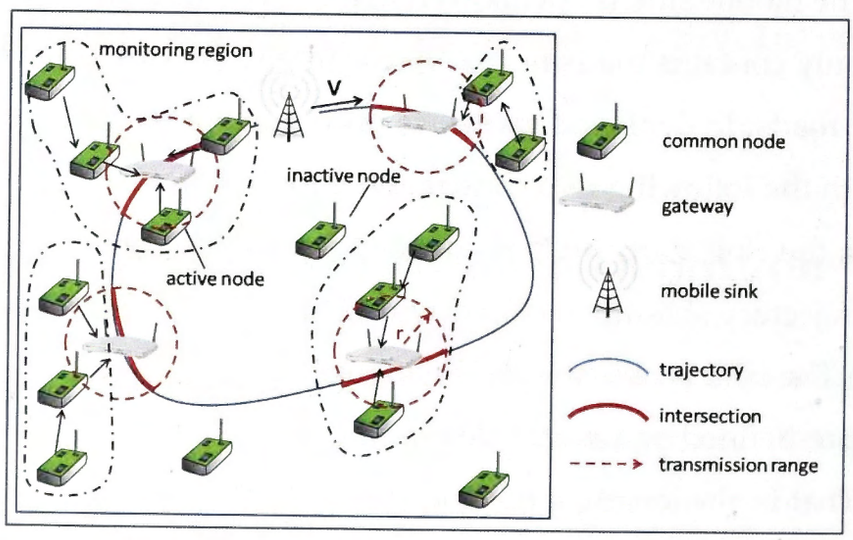


Figure 3.2: An example with $n = 15$, $m = 4$, and $c(g_1) + c(g_2) + c(g_3) + c(g_4) = 12$. Only 12 common nodes can be chosen as the active nodes, and the rest 3 nodes are inactive nodes whose sensed data are discarded.

Though data generated from inactive nodes have to be discarded, they can

be estimated by data collected from the active nodes assuming that high spatial and temporal correlations of data exist among nodes especially in WSNs for environmental surveillance. Denote by the following $T \times n$ matrix X the original data sensed by n nodes within a slotted data collection period of T.

$$\mathbf{X} = \begin{pmatrix} x_{11} & \dots & x_{1i} & \dots & x_{1n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{t1} & \dots & x_{ti} & \dots & x_{tn} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{T1} & \dots & x_{Ti} & \dots & x_{Tn} \end{pmatrix}$$

where x_{ti} represents the sensed data generated from node v_i at time slot t within T, $1 \leq t \leq T$, $1 \leq i \leq n$. Let $\widehat{x}_{ti}$ be the value of estimated data of node v_i at time slot t. The *monitoring quality loss* is formulated using the mean squared estimated error of all nodes per time unit over T,

$$mse(T) = \frac{\sum_{v_i \in V} \sum_{t=1}^{T} (x_{ti} - \widehat{x}_{ti})^2}{n \cdot T} \tag{3.3}$$

Note that the squared estimated error $(x_{ti} - \widehat{x}_{ti})^2 = 0$ for any $v_i \in V'$ at any $t \in T$. Maximizing monitoring quality is equivalent to minimizing monitoring quality loss.

Given a sensor network $G(V \cup GW, E)$ and a road-map $G_M = (V_M, E_M)$ in the monitoring region, gateways are deployed alongside roads in G_M, and a mobile sink starts off the depot p_0, moves in G_M at a constant speed v, and needs to return to p_0 before it travels up to D_{max} in one tour. The *communication time constrained monitoring quality maximization problem* is to find a closed path P with p_0 as its starting and ending points and with length no greater than D_{max}, determine the active nodes and route their data to corresponding gateways, such that the sink is able to efficiently collect sensed data of the active nodes by moving along P and the monitoring quality is maximized. To solve

the problem, the following two challenges should be addressed jointly. On the one hand, we must ensure that all sensed data of the active nodes can be successfully uploaded to corresponding gateways and then to the mobile sink to optimize the monitoring quality, which means individual gateway quota constraints need to be met during the identification of the active nodes, and the collected data are utilized to estimate the missing ones with the objective of minimizing the estimation error. On the other hand, given the road-map G_M and the deployed network G, the to-be-found path P is constrained by the given depot p_0 and D_{max} with the objective that the data collected by the mobile sink moving along P is the most representative subset among all the sensed data, corresponding to the maximum monitoring quality.

Theorem 3.1. *The decision version of the communication time constrained monitoring quality maximization problem in graph $G(V \cup GW, E)$ with a mobile sink moving in road-map $G_M = (V_M, E_M)$ is NP-complete.*

Proof. We show the claim by a reduction from the *Capacitated Minimum Spanning Tree Problem* (CMSTP) [45]. Given a graph $G = (V, E)$, the CMSTP is to find a minimal cost spanning tree rooted at a given node ensuring that all root-incident subtrees have no more than Q nodes, where Q is a pre-defined integer representing the cardinality constraint on the number of nodes in each subtree. CMSTP is NP-hard when $3 \leq Q \leq \lfloor \frac{|V|}{2} \rfloor$ [71]. Given a graph G and the value of Q, the decision version of the instance of CMSTP problem is to determine whether there is a tree spanning nodes in G such that for each subtree, the cardinality is no more than Q.

We construct an instance of the communication time constrained monitoring quality maximization problem in a sensor network $G(V \cup GW, E)$ with a mobile sink moving in road-map $G_M = (V_M, E_M)$, where $V_M = \{p_0, p_1\}$, $E_M = \{(p_0, p_1)\}$, and $|(p_0, p_1)| = \frac{D_{max}}{2}$. Assume that each gateway is assigned with identical quota Q corresponding to the subtrees in CMSTP, where

$m \cdot Q \geq |V|$. The decision version of this special case of the communication time constrained monitoring quality maximization problem is to determine whether there are m disjoint trees rooted at the m gateways spanning nodes in V, subject to the size of each tree being no greater than Q, such that the sink moving back and forth on the road (p_0, p_1) collects data from these trees. Clearly, the sink closed tour is a return trip on road (p_0, p_1) once and the total travelling distance is D_{max}, meeting the distance constraint.

If there is a solution to the instance of the special communication time constrained monitoring quality maximization problem, there is a solution to the instance of the CMSTP problem. Since the CMSTP is NP-hard when $3 \leq Q \leq \lfloor \frac{|V|}{2} \rfloor$ and the reduction is polynomial, the decision version of the communication time constrained monitoring quality maximization problem is thus NP-hard. Meanwhile, the problem of concern is in NP class because it is easy to verify whether a given solution delivers a forest consisting of m trees with no more than Q nodes in each of them. Therefore, the communication time constrained monitoring quality maximization problem is NP-complete.

□

3.3 Algorithm

In the following we propose a novel heuristic for the communication time constrained monitoring quality maximization problem by decomposing the problem into three sub-problems and solving them separately. We also discuss the algorithm implementation in this section.

3.3.1 Overview

First, for the purpose of monitoring quality maximization, we need to study the relationship between monitoring quality and the sink's moving along an

individual road in G_M. We refer to the quality benefit by collecting data through moving along a road as *individual gain*. Clearly, the individual gain of road $e_M \in E_M$ depends on the set of nodes which are selected to transmit their data to the sink via gateways deployed alongside e_M, and such nodes are referred to as *active candidates*. Identifying active candidates in the network and allocating them to gateways are crucial to the calculation of individual gains. To this end, the algorithm explores the sensed data correlation, determines the set of active candidates from V, allocates them to roads in G_M, and calculates individual gains for roads. Second, having individual gains of roads, the maximum optimization problem (monitoring quality maximization problem) is converted into a minimum optimization problem (distance-constrained shortest path problem), and a feasible a path is delivered satisfying the following three constraints: (1) the length of the path should not be greater than D_{max}, (2) the path must start from and end at the depot p_0 in each tour, and (3) the path should include as many roads with high individual gains as possible. And accordingly, the set of active nodes is identified. Third, an energy efficient routing protocol needs to be devised for data transmission from the active nodes to corresponding gateways. The algorithm reduces the routing finding problem to the Steiner Tree Problem and builds trees rooted at individual gateways spanning the identified active nodes.

3.3.2 Nodes Identification and Allocation

Identifying active candidates. To study the monitoring quality benefit of travelling each road $e_M \in E_M$, we relax the distance constraint D_{max} by assuming that D_{max} is large enough and the mobile sink is able to move along each road in the road-map. Then the problem is converted to identifying a set of active candidates $V_c \subset V$ whose sensed data can represent those of other (inactive) nodes with the minimum estimation error, subject to $|V_c| \leq \sum_{i=1}^{m} c(g_i)$, and

allocating the active candidates to each road $e_M \in E_M$ while the number of the allocated active candidates at a gateway is not greater than its quota.

To identify the active candidates, we need to investigate sensed data correlation and explore how data can recover each other. For the purpose of monitoring quality maximization, any single node should be represented by at least one active candidate. Specifically, the monitoring quality loss becomes zero when $\sum_{i=1}^{m} c(g_i) = n$ and thus $V_c = V$. Otherwise, the loss depends on how data of active candidates relate to those of other nodes. Obviously, the loss decreases if the selected active candidates are more data-related to the others.

To explore such correlation, we set up an *initial training phase* with length T_1, during which each node sends sensed data to its neighboring nodes. Data generated from a node v_i at time slot t can be used to estimate those of its neighboring node v_j at t with the minimum error as follows [52],

$$\widehat{x_{tj}} = a_{i,j} \cdot x_{ti} + b_{i,j}, \tag{3.4}$$

where $a_{i,j}$ together with $b_{i,j}$ is referred to as the *estimation model* and can be calculated as follows:

$$a_{i,j} = \frac{n \times \sum_{t=1}^{T_1} x_{ti}x_{tj} - \sum_{t=1}^{T_1} x_{ti} \sum_{t=1}^{T_1} x_{tj}}{n \times \sum_{t=1}^{T_1} (x_{ti})^2 - (\sum_{t=1}^{T_1} x_{ti})^2}, \tag{3.5}$$

and

$$b_{i,j} = \frac{\sum_{t=1}^{T_1} x_{tj} - a_{i,j} \cdot \sum_{t=1}^{T_1} x_{ti}}{T_1}. \tag{3.6}$$

We define that v_i is qualified to *represent* its neighboring sensor v_j (including itself) if and only if $\frac{\sum_{t=1}^{T_1}(x_{tj} - \widehat{x_{tj}})^2}{T_1} \leq \varepsilon$, where ε is a pre-defined tolerant error. With a given ε, each common node builds a set of nodes that it can represent, denoted by Cd_v_i, where $1 \leq i \leq n$, and data generated within T_1 by each

node in Cd_v_i can be estimated by those of v_i with the mean error no greater than ε. Note that $v_i \in Cd\_v_i$ always holds. Let $\mathcal{C} = \{Cd\_v_i | v_i \in V\}$ be the collection of sets derived by the set of common nodes. We aim to find $V_c \subseteq V$ with $|V_c| = \sum_{i=1}^{m} c(g_i)$ such that $\bigcup_{v_i \in V_c} Cd\_v_i = V$.

The algorithm proceeds iteratively. ε is initially assigned to zero and its value is increased by a constant Δ_ε at the beginning of each iteration. With a given ε, a V_c will be identified to represent all common nodes. and the size of V_c will be checked. When the value of ε is very small (close to 0), every Cd_v_i only contains v_i itself, and thus all common nodes will be chosen as active candidates. With the increase of ε, more nodes are included in Cd_v_i while the size of the resultant V_c decreases. The iteration continues until $|V_c| \leq \sum_{i=1}^{m} c(g_i)$. If $|V_c| < \sum_{i=1}^{m} c(g_i)$, $\sum_{i=1}^{m} c(g_i) - |V_c|$ nodes are randomly chosen from $V - V_c$ to become active candidates. The current value of ε is an upper bound of data estimation error per time unit over the initial training phase. The algorithm is referred to as `Find_Active_Candidates` and its detailed description is in **Algorithm 3**.

Algorithm 3: `Find_Active_Candidates`

Input : The set of nodes V, the increment Δ_ε
Output: The set of active candidates V_c

$V_c \leftarrow V$; $\varepsilon \leftarrow 0$;
while $|V_c| > \sum_{i=1}^{m} c(g_i)$ **do**
  $\varepsilon \leftarrow \varepsilon + \Delta_\varepsilon$;
  Build Cd_v_i for each $v_i \in V$;
  $\mathcal{C} \leftarrow \{Cd\_v_i | v_i \in V\}$;
  $V_c \leftarrow$`Set_Cover`$(V, \mathcal{C})$;
if $|V_c| < \sum_{i=1}^{m} c(g_i)$ **then**
  Randomly choose $\sum_{i=1}^{m} c(g_i) - |V_c|$ nodes from $V - V_c$ as active candidates, and add them into V_c;
return V_c.

Algorithm `Find_Active_Candidates` delivers a set of active candidates V_c with $|V_c| = \sum_{i=1}^{m} c(g_i)$, and the estimation model can be calculated by

Algorithm 4: `Set_Cover`

Input : The set of nodes V, the collection of sets F
Output: The set of active candidates S'

$U \leftarrow V$;
$\mathcal{S} \leftarrow F$;
$S' \leftarrow \emptyset$; /* the set of active candidates */;
while $U \neq \emptyset$ **do**
  Select a set $Cd\_v_i \in \mathcal{S}$ such that $|U \cap Cd\_v_i|$ is maximized;
  $S' \leftarrow S' \cup \{v_i\}$;
  $U \leftarrow U - Cd\_v_i$;
  $\mathcal{S} \leftarrow \mathcal{S} - \{Cd\_v_i\}$;
return S'.

Eq. (3.6) and Eq. (3.5). Using the model trained in T_1, the monitoring quality is mainly compromised by the dynamic variance in data correlation over time, which leads to unstable data estimation error. To deal with this issue, we re-calibrate the estimation model regularly during the network lifetime for controlling monitoring quality loss. Particularly, the model trained in T_1 is used only for the following period of T_2, referred to as the *operation phase*, and then it will be re-trained. We analyze the impact of T_1/T_2 on the monitoring quality in Section 3.4.

Allocating active candidates to roads. After V_c is identified, our milestone followed becomes allocating the active candidates to roads in G_M. To this end, we first allocate them to individual gateways, and then to roads.

For active candidates allocation to gateways, we aim to minimize the energy consumption on node-gateway data transmission, which is distance dependent. Accordingly, we formulate the problem as partitioning $V_c{}'$ into m disjoint subsets $S_1, S_2, \ldots, S_m$, such that $\bigcup_{i=1}^{m} S_i = V_c, S_i \cap S_j = \emptyset, 1 \leq i, j \leq m, i \neq j$, subject to (1) the m subsets correspond to the m gateways, i.e., $|S_i| = c(g_i), 1 \leq i \leq m$; and (2) a gateway and its active candidates subset should be as close as possible in distance, i.e., $\sum_{i=1}^{m} \sum_{u \in S_i, g_i \in GW} dist(u, g_i)$ is to be minimized, where $dist(u, g_i)$ is the Euclidean distance between u and g_i in G.

To solve the above problem, we first construct a weighted bipartite graph $G' = (V_c \cup GW', E', d)$, where $GW' = \bigcup_{i=1}^{m} g_{aux_i}$ is the collection of m sets of auxiliary gateways, $E' = \{\langle u, g\rangle | u \in V_c, g \in GW'\}$, and $d(u,g) = dist(u,g)$, the Euclidean distance between u and g. Let $g_{aux_i} = \{g_{i1}, g_{i2}, \ldots, g_{ic(g_i)}\}$ be a set containing $c(g_i)$ copies of g_i and $dist(u, g_{ij}) = dist(u, g_i)$, for $u \in V_c$ and $1 \leq j \leq c(g_i)$. We next find a perfect matching of G' by the Hungarian algorithm [54], aiming to minimize $\sum_{e \in M} d(e)$, where M is the set of matched edges with $|M| = |V_c|$. In the resultant matching, all $g \in GW'$ and $u \in V_c$ are involved in the $|M|$ matched edges. For each matched $u \in V_c$, there is a matched edge with $g \in GW'$ as the other endpoint, representing that each active candidate is assigned to an auxiliary gateway. We then merge auxiliary gateways $g_{i1}, g_{i2}, \ldots, g_{ic(g_i)}$ to their original node g_i and put their matched nodes into S_i. Since each auxiliary gateway $g \in GW'$ is associated with one active candidate $u \in V_c$, the number of matched nodes added into S_i is exactly $c(g_i)$. That is, $|S_i| = c(g_i)$, meeting the quota constraint. We finally assign each road $e_M \in E_M$ with corresponding set of active candidates, denoted by $S(e_M) = \{S_i | \text{if } g_i \text{ is on the road } e_M \in E_M\}$. As a result, the set of active candidate V_c is partitioned into $|E_M|$ disjoint sets, $S(e_1), S(e_2), \ldots, S(e_{|E_M|})$. And the individual gain of road $e_M \in E_M$ is obtained by using Eq.(3.3), denoted by $q(e_M)$.

3.3.3 Distance-constrained Trajectory

The closed path to be found in the road-map G_M should include roads with high individual gains subject to the distance constraint D_{max}. We approach the problem by referring to the methodology in [58], that is, converting the maximum optimization problem into a classic *distance-constrained shortest path* problem, which has been extensively studied [21,39].

We first construct a weighted, directed graph $G_d = (V_M, E_M, \omega, d)$. Re-

call that $V_M = \{p_0, p_1, p_2, \ldots, p_{n_M}\}$ represents the set of road crossings where p_0 should be the starting/ending point of the closed path, and E_M is known as the set of road edges in which an edge $e_M \in E_M$ can be also expressed as $e_M = \langle p_i, p_j \rangle$ where $p_i, p_j \in V_M$ are the two endpoints of road e_M in G_M. The weight of each edge $e_M = \langle p_i, p_j \rangle \in E_M$ is $\omega(p_i, p_j) = q(\langle p_i, p_j \rangle)$ and $d(p_i, p_j)$ is the Euclidean distance between p_i and p_j, in other words, the geometric length of e_M. Accordingly, finding an optimal tour in G_M is then reduced to finding a closed path in G_d, starting from and ending at p_0, such that the total weight of the edges in the path is maximized, while D_{max} is not violated by the sum of the edge length in the path. We refer to this problem as *the distance-constrained longest path problem*, which is NP-hard, since the well known NP-hard Hamiltonian problem [33] is a special case of the problem where no distance constraint is imposed.

We transform the problem to a *distance-constrained shortest path problem* in another auxiliary directed graph $G'_d = (V_M, E_M, \omega', d)$ constructed as follows. For each directed edge $\langle p_i, p_j \rangle \in E_M$,

$$\omega'(p_i, p_j) = \begin{cases} \Phi & \text{if } q(\langle p_i, p_j \rangle) = 0, \\ \frac{1}{q(\langle p_i, p_j \rangle)} - \rho & \text{otherwise,} \end{cases} \tag{3.7}$$

where Φ and ρ are non-negative constants, $\Phi \geq q_{max}$, $0 < \rho \leq \frac{1}{q_{max}}$, and $q_{max} = \max_{e_M \in E_M}\{q(e_M)\}$. The introducing of term ρ breaks the tie of two shortest paths between a pair of nodes with equal length by favoring the one with the larger individual gain. Let E_P be the set of edges in the identified path P. Consequently, the original *distance-constrained longest path problem* in G_d is well transferred to the *distance-constrained shortest path problem* in G'_d, which is defined as finding such a closed path P consisting of vertices in V_M, with p_0 as both its starting point and end point, s.t., $\sum_{\langle p_i, p_j \rangle \in E_P} \omega'(p_i, p_j)$ is minimized while $L(P) = \sum_{\langle p_i, p_j \rangle \in E_P} d(p_i, p_j) \leq D_{max}$.

We modify an approximation algorithm by Chen *et al.* [21] for the distance-constrained shortest path problem, which finds a feasible tour and then locally improves the found tour. We calculate all pairs of shortest paths for all pairs of nodes $u, v \in V_M$ in G'_d, denoted by $D(u,v)$. $P = \langle p_0 \rangle$ initially. Let $P = \langle p_0, p_1, \ldots, p_i, p_0 \rangle$ be the currently constructed path. We extend P by adding a next vertex u into P, which minimizes the total weight of the edges in the path $\sum_{j=0}^{i-1} \omega'(p_j, p_{j+1}) + \sum_{e_k \in D(p_i,u)} \omega'(e_k) + \sum_{e_l \in D(u,p_0)} \omega'(e_l)$, subject to the constraint D_{max}

$$\sum_{j=0}^{i-1} d(p_j, p_{j+1}) + |D(p_i, u)| + |D(u, p_0)| \leq D_{max}. \tag{3.8}$$

All vertices in $D(p_i, u)$ and $D(u, p_0)$ should be added into P in order. The process continues until the distance constraint is no longer met. Note that $D(p_i, u)$ and $D(u, p_0)$ may contain vertices that have already been visited and included in P. Adding them again to P is, however, necessary. Otherwise, the mobile sink may not be guaranteed to return to p_0 after visiting all other vertices in P. We here use an example to illustrate such tour extension as shown in Fig. 3.3. Assume that $P = \langle p_0, p_1, p_3, p_5, p_0 \rangle$ is the path to be extended, the shortest paths from p_5 to p_7, and from p_7 to p_0 are respectively $D(p_5, p_7) = \langle p_5, p_1, p_7 \rangle$ and $D(p_7, p_0) = \langle p_7, p_4, p_0 \rangle$. We now check whether adding p_7 to P still meets the D_{max} constraint. If we only consider to add un-visited vertices into P, the updated path will be $P = \langle p_0, p_1, p_3, p_5, p_7, p_4, p_0 \rangle$. Assume $L(P) \leq D_{max}$. Note that there is no edge directly connecting p_5 and p_7, or p_7 and p_0 in G'_M. However, after visiting p_5, the sink is not able to visit p_7 by following P since $\langle p_5, p_7 \rangle \notin E'$. In order to reach p_7, the mobile sink has to visit other vertices (p_1 in this example). Since $|D(p_5, p_7)| = d(p_5, p_1) + d(p_1, p_7) \geq d(p_5, p_7)$, the actual tour length may exceed D_{max}. Therefore, we need to follow Eq.(3.8) to check whether the distance constraint is met. If it is still met, the updated path should be $P = \langle p_0, p_1, p_3, p_5, p_1, p_7, p_4, p_0 \rangle$.

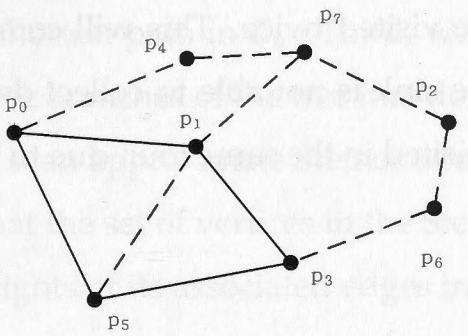


Figure 3.3: Tour extension in trajectory finding for the mobile sink

We now improve the found tour further. Assume that a feasible tour $P = \langle p_0, p_1, \ldots, p_k, p_0 \rangle$ has been found with $L(P) = \sum_{e \in E_P} dist(e) \leq D_{max}$, where $dist(e)$ is the Euclidean distance between two endpoints of e. We perform a local improvement on the feasible solution by adding as many vertices to P as possible until the distance constraint is violated. To do so, we iteratively check whether there exists a vertex $p_j \notin P$ meeting the conditions: (i) $\langle p_i, p_j \rangle \in E_M$ and $\langle p_j, p_{i+1} \rangle \in E_M$, where $\langle p_i, p_{i+1} \rangle \in E_P$, $i \neq 0$, and (ii) $L(P) + d(p_i, p_j) + d(p_j, p_{i+1}) - d(p_i, p_{i+1}) \leq D_{max}$. If yes, add p_j into P since a better path $P = \langle p_0, p_1, \ldots, p_i, p_j, p_{i+1}, \ldots, p_k, p_0 \rangle$ is found. If there are multiple vertices meeting the distance constraint, the one leading to the maximum individual gain will be added. This procedure continues until the D_{max} constraint is no longer met.

Once P is found, the set of active nodes can be determined, which is $V' = \bigcup_{e \in E_P} S(e)$ ($S(e)$ is the set of active candidates allocated to road $e \in E_M$), and the set of gateways GW_P which will upload data to the sink are those deployed on edges in E_P. The mobile sink moves along E_P and collects the sensed data of active nodes by visiting gateways in GW_P.

There may exist duplicate nodes or edges in the tour. In the above example, p_1 is visited twice in $P = \langle p_0, p_1, p_3, p_5, p_1, p_7, p_4, p_0 \rangle$. Now assume the shortest path between p_7 and p_0 is $SP(p_7, p_0) = \langle p_7, p_1, p_0 \rangle$, the updated tour $P = \langle p_0, p_1, p_3, p_5, p_1, p_7, p_1, p_0 \rangle$, where p_1 is visited three times and edges

$\langle p_0, p_1 \rangle$ and $\langle p_1, p_7 \rangle$ are visited twice. This will compromise the monitoring quality since the mobile sink is not able to collect data when it moves along an edge that has been visited in the same tour, due to the constrained gateway quotas.

3.3.4 Data Routing Protocol

Having allocated active nodes V' to gateways in GW_P, we next devise an energy-efficient routing protocol for the node-gateway data transmission in the operation phase. Assume an active node $v_i \in V'$ is allocated to gateway g_j on road $e \in E_P$, with $dist(v_i, g_j) > r$. In that case, v_i cannot send its sensed data to g_j directly, rather, it needs at least one node acting as a relay node between them. Note that such relay nodes could be active nodes allocated to the same gateway, those allocated to other gateways, or inactive nodes.

We aim to construct a routing structure $\mathcal{F}$ consisting of $|GW_P|$ trees, $\mathcal{F} = \{T_i \mid T_i \text{ is tree rooted at } g_i \text{ spanning all active nodes in } S_i, 1 \leq i \leq |GW_P|\}$ (recall that S_i is the subset of active nodes which send their data to the mobile sink via gateway g_i). For a tree T_i rooted at g_i, the Euclidean distance between any two connected nodes in T_i is to be minimized, s.t., $\sum_{\langle u,v \rangle \in T_i} dist(u, v)$ is minimized, which is then reduced to the Steiner Tree Problem (STP), with S_i as the *terminal set*. STP is NP-hard, and we find a solution for it by modifying the approximation algorithm by Kou *et al.* [53] as follows.

First, we construct a weighted graph $G_w = (V \cup GW, E, \eta)$, where $\eta(u, v) = dist(u, v)$ for each $\langle u, v \rangle \in E$. And we compute all pairs of shortest paths in G_w. An auxiliary complete weighted graph G_{S_i} consisting of only S_i is constructed. The weight assigned to each edge in G_{S_i} is the length of the shortest path between the two endpoints of the edge in G_w. Second, we find a minimum spanning tree (MST) in G_{S_i}. Let E_{OPT} be the set of edges in the MST. A subgraph of G_w, G_i, is induced by the edges in E_{OPT}. Note that each edge in

E_{OPT} corresponds to a shortest path in G_w. Third, we find a MST in the subgraph G_i, and prune those branches of the tree that do not contain the node in S_i. The resultant tree T_i is an approximate Steiner tree with S_i as the terminal set. Finally, assuming that the set of vertices in the Steiner tree is V_{S_i}. for each $u \in V_{S_i}$, we increase weights of its associated edges in G_w by a small constant Δ_w. As a result, the opportunity of the same node involved in more than one Steiner trees is reduced and the energy consumption among sensors can be further balanced. The found Steiner tree T_i is used to route data from nodes in S_i to gateway g_i, where $1 \leq i \leq |GW_P|$. And the trees form the routing structure $\mathcal{F}$.

Note that T_i rooted at g_i may include nodes not in S_i for the purpose of data relay, and in that case, the number of descendants of g_i, $|V_{S_i}|$, will exceed the quota $c(g_i)$. However, gateway g_i only receives data from active nodes in S_i, and sensed data of the other nodes in tree T_i will not be forwarded to g_i.

The heuristic algorithm is referred to as `Monitor_Quality_Max`, or `MQM` for short.

3.3.5 Algorithm Implementation

We now discuss the implementation of algorithm `MQM`, illustrated in Fig. 3.4.

The network lifetime L, which is defined as the time of the first sensor's failure due to the depletion of its energy [19], consists of multiple training phases and operation phases. The length of a training phase is T_1 while the length of an operation phase is N times of the duration of a tour ($\frac{L(P)}{v}$), i.e., $T_2 = N \cdot \frac{L(P)}{v}$, where N is a positive integer, to ensure the mobile sink returns to the depot after each operation phase. At the beginning of the network lifetime, a minimum spanning tree spanning all common nodes in G is built for data collection within the training phase T_1. Then the energy consumption of each node within T_1 is calculated, denoted by $E_{c1}(v_i)$. Let $E_r(v_i)$ be the resid-

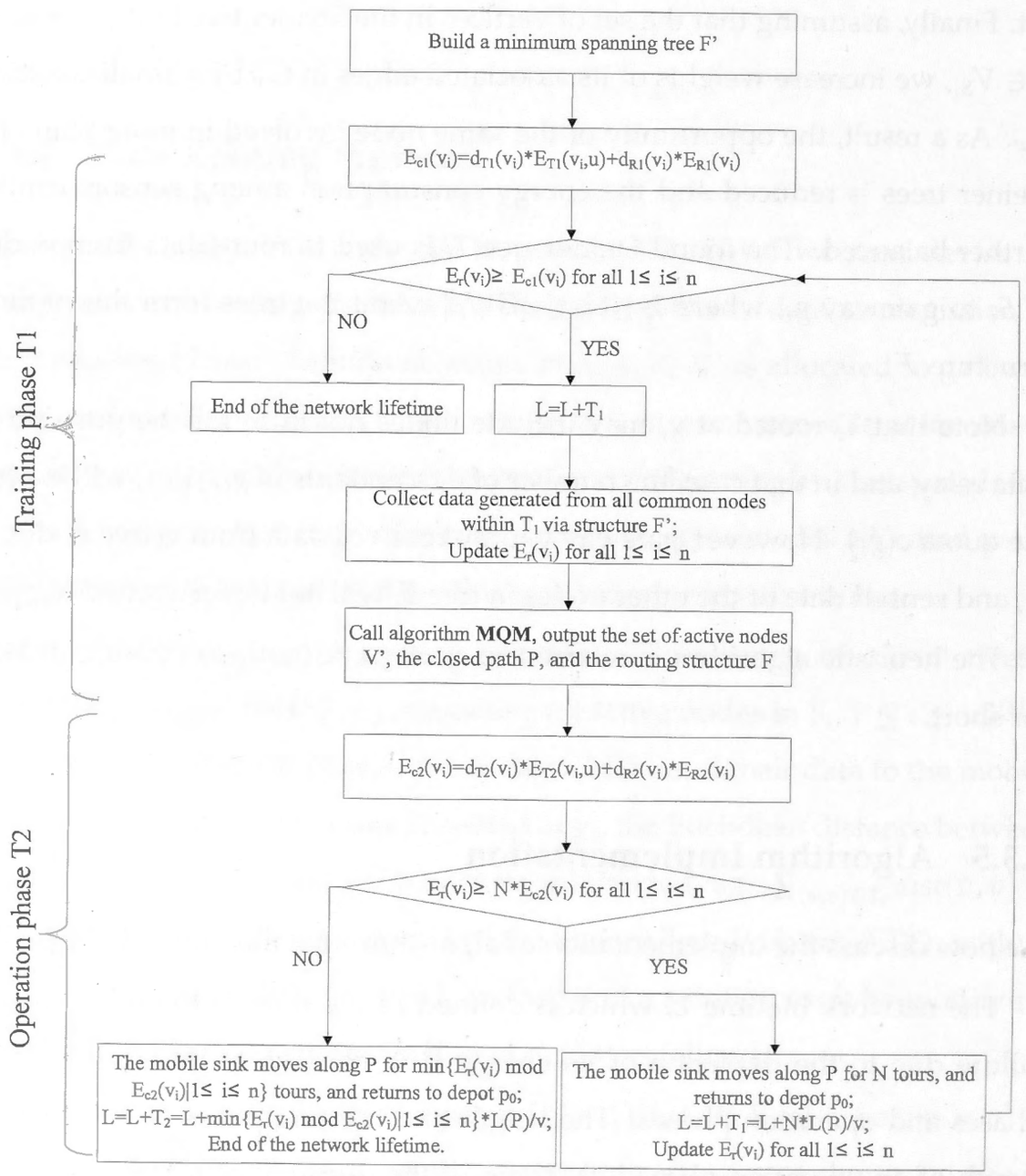


Figure 3.4: Flow chart of the network lifetime calculation

ual energy at node v_i. If $E_r(v_i) \geq E_{c1}(v_i)$ for $1 \leq i \leq n$, no node will deplete its energy in the current training phase, and network lifetime L is increased by T_1. Otherwise, at least one node will use up its energy within the following T_1 and the network lifetime ends. At the end of each training phase, algorithm MQM is implemented and delivers the set of active nodes V', a closed path P, and a routing structure $\mathcal{F}$. At the beginning of each operation phase, the energy consumption of each node in one tour's duration ($\frac{L(P)}{v}$) is calculated and denoted by $E_{c2}(v_i)$. If the residual energy of any node is smaller than the amount of energy to be consumed within N tours ($N \cdot E_{c2}(v_i)$), the network lifetime ends when the node depletes its energy, and L is increased by $min\{E_r(v_i) \mod E_{c2}(v_i) \mid 1 \leq i \leq n\} \cdot \frac{L(P)}{v}$. Otherwise, the whole operation phase ends with all nodes alive, L is increased by $T_2 = N \cdot \frac{L(P)}{v}$, and the above procedure is repeated with a new training phase. That is, algorithm *MQM* is implemented every $(T_1 + T_2)$, with the trajectory and the routing structure re-delivered accordingly.

The energy consumption of nodes in training phase and operation phase is calculated as follows. Considering only the energy consumption on wireless communication including data transmission and reception [74], we obtain the following equation.

$$E_{c1}(v_i) = d_{T1}(v_i) \cdot E_T(v_i, u') + d_{R1}(v_i) \cdot E_R(v_i), \tag{3.9}$$

assuming v_i transmits data to node u' in $\mathcal{F}'$, where $E_R(v_i)$ and $E_T(v_i, u')$ are the amounts of energy consumed by v_i on receiving 1 bit of data and transmitting 1 bit of data to u'; $d_{T1}(v_i)$ and $d_{R1}(v_i)$ are respectively the amounts of data transmitted from and received by v_i in the training phase T_1.

$$E_R(v_i) = \epsilon_{elec}, \tag{3.10}$$

and

$$E_T(v_i, u') = \epsilon_{elec} + \epsilon_{amp} \cdot dist^2(v_i, u'), \tag{3.11}$$

where ϵ_{elec} is the energy cost of processing 1 bit data and ϵ_{amp} is the transmitter amplifier, $dist(v_i, u')$ is the Euclidean distance between v_i and u'. Let $c(v_i)$ be the number of nodes using v_i as the relay node in the routing structure (either $\mathcal{F}'$ or $\mathcal{F}$), then

$$d_{T1}(v_i) = T_1 \cdot r_g \cdot (c(v_i) + 1), \tag{3.12}$$

and

$$d_{R1}(v_i) = T_1 \cdot r_g \cdot c(v_i). \tag{3.13}$$

Assuming v_i transmits data to node u in $\mathcal{F}$,

$$E_{c2}(v_i) = d_{T2}(v_i) \cdot E_T(v_i, u) + d_{R2}(v_i) \cdot E_R(v_i), \tag{3.14}$$

where $d_{T2}(v_i)$ and $d_{R2}(v_i)$ are respectively the amounts of data transmitted from and received by v_i in one tour.

$$d_{T2}(v_i) = \begin{cases} \frac{L(P)}{v} \cdot r_g \cdot (c(v_i) + 1) & \text{if } v_i \in V', \\ \frac{L(P)}{v} \cdot r_g \cdot c(v_i) & \text{otherwise,} \end{cases} \tag{3.15}$$

and

$$d_{R2}(v_i) = \frac{L(P)}{v} \cdot r_g \cdot c(v_i). \tag{3.16}$$

Theorem 3.2. *For a wireless sensor network $G(V \cup GW, E)$ with a mobile sink moving in a given road-map, the communication time constrained monitoring quality maximization problem can be solved by the proposed algorithm* MQM *with complexity $O(n^4)$, where n is the number of sensor nodes in G.*

Proof. The complexity of algorithm MQM depends on the following four stages. In the first stage, the `Find_Active_Candidates` takes $O(n^4)$ with algorithm `Set_Cover` being implemented in time $O(n^3)$ and the number of iterations in `Find_Active_Candidates` being bounded by n. In the second one, allocating active candidates to gateways takes $O(n^4)$ because the computation complexity of the Hungarian algorithm is $O(|V'|^4) = O(n^4)$. In the third one, the distance-constrained tour P is found within $O(n^2)$ [58]. In the last stage, as shown in [53], the data routing structure $\mathcal{F}$ is built in $O(n^3)$. Therefore, it is safe to conclude that the overall complexity of algorithm MQM is $O(n^4)$. □

3.4 Performance Evaluation

In this section, we evaluate the performance of the proposed algorithm MQM through extensive experiments by simulation. Specifically, we vary network topologies with different numbers of gateways and common nodes, and evaluate the monitoring quality in terms of data estimation error, and the energy consumption on data collection.

3.4.1 Experiment Settings

We consider a wireless sensor network deployed in a $150m \times 150m$ square region. The road-map in the monitoring region is shown in Fig. 3.5, in which there are 13 roads and 8 road crossings A, B, C, D, E, F, G, and H. The crossing A is the depot, where the sink starts from and returns to during each tour.

The parameter settings in the simulation are as follows. The speed of the mobile sink $v = 2\ m/s$, the data transmission rates of gateways are identical to be 100 $bits/s$, the data generation rate of sensor nodes is $r_g = 1\ bit/s$, the transmission range of sensors and gateways is $r = 10\ m$, the initial energy capacity of sensors $IE = 100$ *Joules*, $\epsilon_{amp} = 100\ pJ/bit/m^2$, $\epsilon_{elec} = 50\ nJ/bit$,

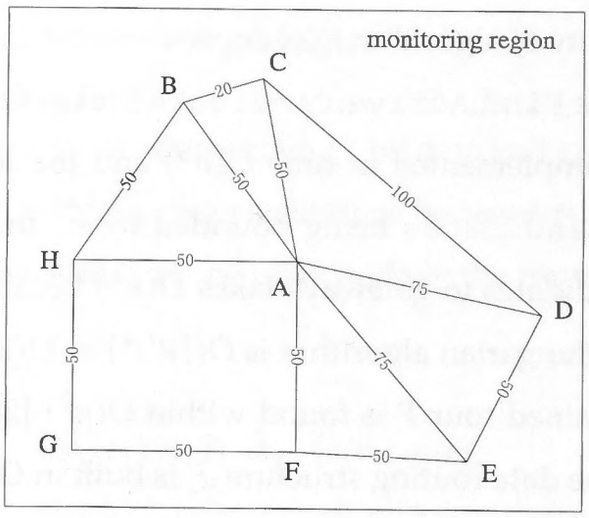


Figure 3.5: A road-map in the monitoring region

adopting the energy consumption model in [40]. All experiment results are the mean of the results delivered by the algorithm with 50 different network topologies and identical parameter settings.

The synthetic data used for simulation is generated following a random walking pattern [26, 52, 68]. Firstly, the initial value for each common node is chosen uniformly in the range $[0, 100]$. Then, all common nodes are partitioned into K classes, and the values of nodes belonging to the same class are further changed by making random step with a certain probability, which is uniformly chosen in the range $[0.2, 1]$. In that case, the n common nodes have K classified behaviors in terms of their sensed data and nodes in the same class have the same behavior. For example, data correlation of common nodes is at its maximum when $K = 1$ (i.e., all common nodes have the same behavior), and it decreases when the value of K increases.

We generate a set of synthetic data for $T_1 + T_2$ time units, following the above strategy. We use data generated within the first T_1 units to train the estimation model, and adopt data generated in the last T_2 units to evaluate the effectiveness of the model by calculating monitoring quality loss using Eq.(3.3). Define the ratio of T_1 to $T_1 + T_2$ as the initial training phase ratio,

denoted by $R = T_1/T_2$. Specifically, in our default simulation setting, T_2 is 8 times of $\frac{L(P)}{v}$ and R is set to be 10%.

3.4.2 Impact of Constraint Parameters on Network Performance

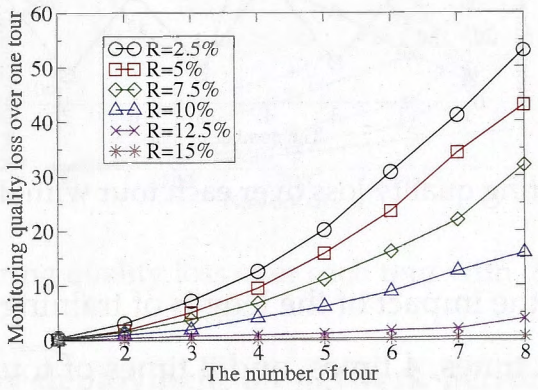


Figure 3.6: Monitoring quality loss over each tour with different initial training phase ratios

Impact of the training data set on monitoring quality. We first evaluate the impact of the synthetic training data on the network performance, by studying the influence of the initial training phase ratio R on the monitoring quality loss over each tour. As shown in Fig. 3.6, we vary R from 2.5% to 15% with the increment of 2.5% while fixing $n = 100$, $m = 50$, $D_{max} = 1000m$. Two main observations are obtained accordingly. The first one is that with a fixed R, the loss goes up as the number of tour increases, which is due to the reason mentioned in Section 3.3 that the estimation model is outdated and unable to reflect the sensed data correlation accurately after a while. The second one is that the loss becomes smaller with a greater R within any single tour. Taking the 4^{th} tour as an example, the loss is over 12 with $R = 2.5\%$ while the loss is lowered to about 7 when R increases to 7.5%. The reason behind is that a longer initial training phase leads to more accurate and stable estimation model, which is valid for a larger number of tours. From another point of view, estimation model with a smaller R is required to be updated more

frequently.

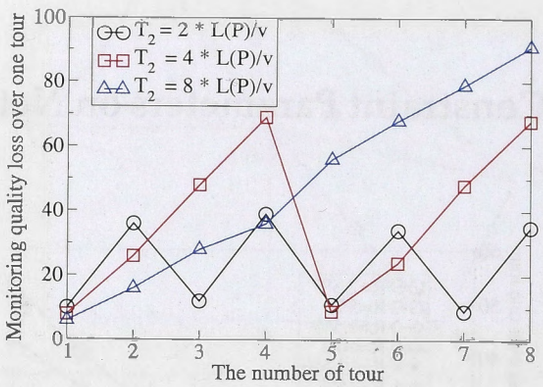


Figure 3.7: Monitoring quality loss over each tour with different values of T_2

We then evaluate the impact of the length of training and operation phases by varying T_2 from 2 times, 4 times, and 8 times of tour duration $\frac{L(P)}{v}$, while fixing $D_{max} = 200m$ and $R = 10\%$. Fig. 3.7 shows that with $T_2 = 2 \times \frac{L(P)}{v}$, the quality loss could be controlled below 40 over tours whereas with the growth in T_2, the highest loss increases. However, with a larger value of T_2, the loss over the first few tours becomes smaller. Taking the first two tours as an example, the loss with $T_2 = 8 \times \frac{L(P)}{v}$ is the lowest compared with those with $T_2 = 2 \times \frac{L(P)}{v}$ and $T_2 = 4 \times \frac{L(P)}{v}$, and within the first four tours, the loss with $T_2 = 8 \times \frac{L(P)}{v}$ is lower than that with $T_2 = 4 \times \frac{L(P)}{v}$. The results verify that re-building the data estimation model is essential to control the monitoring quality loss. Also it indicates that with the same ratio of $R = T_1/T_2$, the larger the T_2, the more accurate the data estimation at the beginning in the following operation phase, yet potentially the greater the quality loss in a later stage of the operation phase.

We finally investigate the number of common node classes K by varying K from 1 to 9 with the increment of 2. From Fig. 3.8, we observe that the monitoring quality loss is relatively small when $K = 1$ and it goes up as K increases. The result verifies that when the value of K is small, data correlation of nodes is high and data estimation of inactive nodes is accurate. When K is

large, sensed data among nodes are not highly correlated and data estimation causes a relative greater loss.

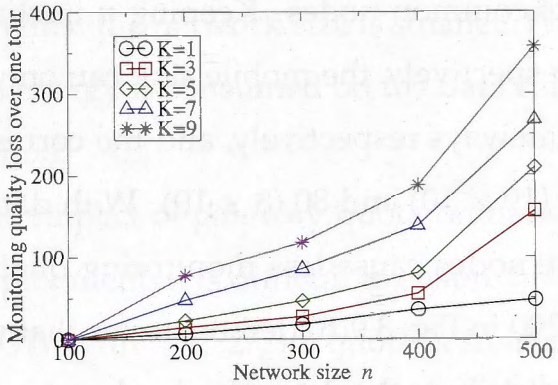
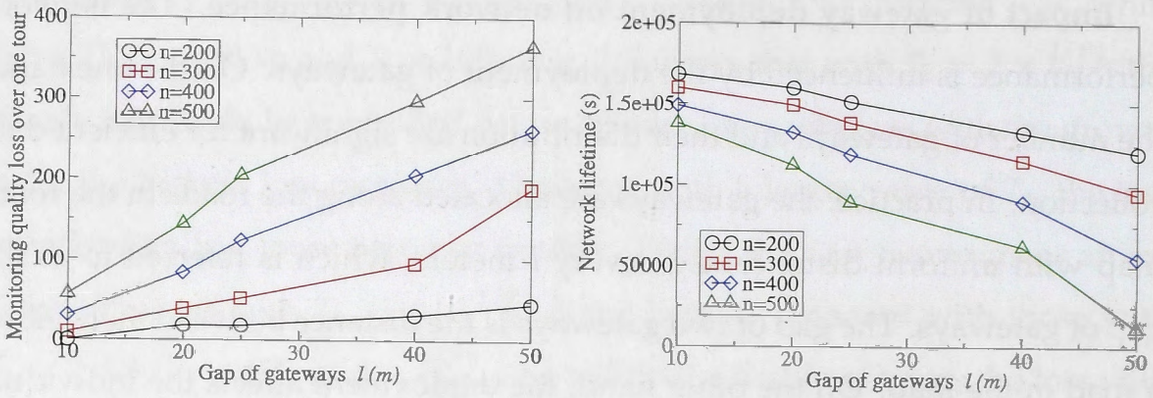


Figure 3.8: Monitoring quality loss over each tour with different values of K

Impact of gateway deployment on network performance. The network performance is influenced by the deployment of gateways. On the one hand, the number of gateways and their distribution are significant for efficient data collection. In practice, the gateways are allocated along the roads in the road-map with uniform distance, e.g., every l meters, which is referred to as the gap of gateways. The gap of two gateways is the distance between them along a road in the map. On the other hand, the deployment affects the individual quotas of gateways and determine their data relay capabilities. As mentioned before, the quota of a gateway $g_i \in GW$ depends on its transmission range and path intersection length l_i, which is dependent on the vertical distance between gateway g_i and its deployed road.

We first evaluate monitoring quality and network lifetime by setting l to 10m, 20m, 25m, 40m, and 50m respectively, while fixing $n = 200$ and $D_{max} = 200m$. The quota of each gateway is assigned to the maximum 10 by Eq. (3.2), when $l_i = 2r$ for each gateway $g_i \in GW$. In Fig. 3.9(a), the curve with $n = 200$ shows the monitoring quality loss with different values of l. It indicates that the more densely deployed the gateways (the smaller the l), the smaller the monitoring quality loss. This is because more gateways will be able to

relay data from more sensors to the mobile sink. For example, when $l = 10m$, there is no monitoring quality loss, since the mobile sink can visit 20 gateways ($200m/10m = 20$) and thus collect data from 200 active nodes (20×10), which is the complete set of common nodes. Keeping n unchanged, when we set l to be 20m and 25m respectively, the mobile sink can only visit 10 ($200m/20m$) and 8 ($200m/25m$) gateways respectively, and the corresponding numbers of active nodes are 100 (10×10) and 80 (8×10). With data loss unavoidable in both cases, 100 active nodes cause less monitoring quality loss than 80 ones. The curve with $n = 200$ in Fig. 3.9(b) demonstrates that with a smaller l, more gateways are able to distribute the data relay load more evenly among sensors, resulting in a longer network lifetime.



(a) Monitoring quality loss with different gaps of gateways

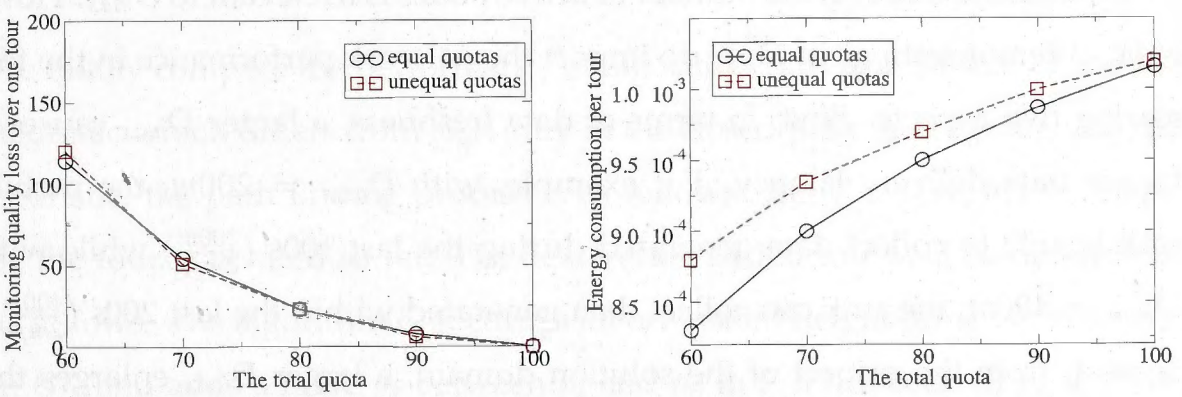
(b) Network lifetime with different gaps of gateways

Figure 3.9: Performance evaluation of algorithm MQM with different gaps of gateways

The value of l is determined by both the user requirement on the monitoring quality and the application budget. The number of gateways is $\lfloor L/l \rfloor$ and the gateway quota is $\frac{l_i \cdot r_l}{L \cdot r_g} = \frac{2r \cdot r_l}{L \cdot r_g}$ referring to Eq.(3.2) when $l_i = 2r$. If the data transmission rates of all gateways are identical, the total number of active nodes is $\frac{L}{l} \cdot \frac{2r \cdot r_l}{L \cdot r_g} = \frac{2r \cdot r_l}{l \cdot r_g}$. That is, the number of active nodes only depends on l. If quality loss is not acceptable, gateways should be deployed every $\frac{2r \cdot r_l}{n \cdot r_g}$ meters on roads. If the application budget is limited for a large number of gateways, we should increase l at the cost of lower monitoring quality. Also

note that when the network size n increases from 200 to 500 while l keeps a constant, the quality loss increases and the network lifetime decreases, shown in Fig. 3.9. This is because the same number of active nodes can better represent the other nodes when the network size is smaller. With the increase in the network size n, more energy is consumed on the data collection, resulting in a shorter network lifetime.

We now study the impact of gateway quota on network performance. In practical network deployment, it is difficult to ensure $l_i = 2r$ for each gateway. For a gateway $g_i \in GW$ with $l_i < 2r$, its quota is smaller than 10. Under the deployment with $D_{max} = 200m$, $l = 20m$, $n = 200$, we vary the total gateway quota from 60 to 100 with the increment of 10 and evaluate the network performance. We analyze both *equal quota* and *unequal quota* scenarios. In the equal scenario, quotas of the 10 gateways are assigned equally, while in the unequal scenario, quotas are randomly generated with their sum fixed as a given total quota and then assigned to the 10 gateways. For example, with a total quota of 60, the quota of every gateway is 6 in the equal quota scenario, whereas the value varies from each other in the unequal quota scenario with the sum fixed as 60.



(a) Impact of the gateway quota on the monitoring quality loss

(b) Impact of the gateway quota on the energy consumption

Figure 3.10: Impact of the gateway quota on the network performance

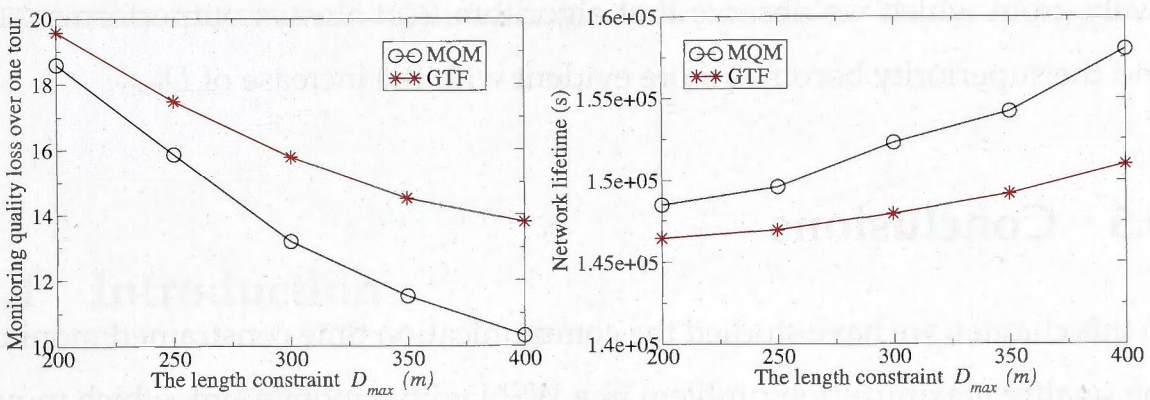
Fig. 3.10(a) shows monitoring quality loss over each tour with different to-

tal quotas. It is observed that the larger the total quota, the smaller the loss delivered because more active nodes are chosen. Also, with the same total quota, the monitoring quality loss in the two scenarios is almost identical, indicating that the loss does not depend on the individual quota assignment but the total quota value. Fig. 3.10(b) plots energy consumption on data collection per tour with different total quotas. The consumed energy goes up with the increase of the total quota since data from more common nodes needs to be transmitted. We also notice that the unequal quota distribution results in higher energy consumption because data from active nodes has to be relayed to distant nodes to reach the corresponding gateways. The two curves intersect at total quota=100, since all common nodes are active nodes and the routing structures are the same. And the gap becomes larger as the total quota decreases.

Impact of road-map on network performance. We then evaluate various road-maps on the network performance. We adopt the road-map shown in Fig. 3.5 and vary the distance constraint D_{max} from 200m to 400m with the increment of 50 while fixing $l = 25m$ and $n = 200$. Fig. 3.11 shows that a larger D_{max} results in better network performance.

As discussed above, the number of active nodes is irrelevant to D_{max}. However, different settings of D_{max} do impact the network performance in the following two aspects. First, in terms of data freshness, a larger D_{max} causes a longer data delivery latency. For example, with $D_{max} = 200m$, the mobile sink is able to collect data generated during the last 100s ($\frac{200m}{2m/s}$) while with $D_{max} = 400m$, the sink can collect data generated within the last 200s ($\frac{400m}{2m/s}$). Second, from the respect of the solution domain, a larger D_{max} enlarges the domain for a better solution with smaller monitoring quality loss and more energy-efficient data collection. With $l = 25m$, the number of active nodes is 80 regardless of the value of D_{max}. Setting D_{max} to be 200m and 400m, e.g., the numbers of gateways the mobile sink can visit are 8 and 16, with quotas 10 and

5 respectively ($8 \times 10 = 16 \times 5 = 80$). However, compared with $D_{max} = 200m$, in the scenario with $D_{max} = 400m$ we find path P in the road-map with relaxing length constraint. Accordingly, edges with higher monitoring quality are more likely to be involved in P and the data relay load is distributed more evenly among nodes.



(a) Monitoring quality loss with different D_{max}

(b) Network lifetime with different D_{max}

Figure 3.11: Performance evaluation of algorithm MQM and GTF with different distance constraints

3.4.3 Performance Comparison

We finally compare the performance of algorithm MQM against that of another heuristic which differs from MQM only in the closed path finding stage. In this heuristic, the path finding process is as follows. Let $P = \langle p_0, p_1, \ldots, p_l, p_0 \rangle$ be the tour generated so far. The next vertex added to P will be determined as follows. The algorithm first checks all un-visited neighboring vertices of p_l in G_M and adds a node p_k between p_l and p_0 in P if it meets: (i) $p_k \notin P$; (ii) $q(\langle l,k \rangle) = max\{q(\langle l,i \rangle) | \langle l,i \rangle \in E_M\}$; and (iii) $|D(p_k, p_0)| + d(p_l, p_k) + L(P) - d(p_l, p_0) \leq D_{max}$. If no such a vertex is found, the algorithm checks all visited neighboring vertices of p_l, finds $p_k \in P$ with the minimum $|D(p_k, p_0)| + d(p_l, p_k) + L(P) - d(p_l, p_0)$ ($\leq D_{max}$), and adds it between p_l and p_0 in P. The

algorithm terminates if the D_{max} constraint is no longer met. We refer to this heuristic as algorithm `Greedy_Tour_Finding`, or `GTF` for short.

We evaluate the network performance delivered by algorithms `MQM` and `GTF` through fixing $n = 200$, $l = 25m$, and varying D_{max} from 200 to 400 with the increment of 50. The monitoring quality loss and the network lifetime delivered by two algorithms are shown in Fig. 3.11(a) and Fig. 3.11(b) respectively, from which we observe that algorithm `MQM` always outperforms `GTF`, and the superiority becomes more evident with the increase of D_{max}.

3.5 Conclusions

In this chapter, we have studied the communication time constrained monitoring quality maximization problem in a WSN with a mobile sink which moves in a given road-map. We formulated the problem as a joint optimization problem consisting of identifying the set of active nodes, finding a closed distance-constrained path, and devising an energy-efficient routing protocol, such that the mobile sink can efficiently collect data from active nodes by moving along the path with monitoring quality maximized. We proved its NP-hardness and provided a heuristic instead. We finally conducted extensive experiments by simulation to evaluate the effectiveness of the proposed algorithms. Through the experiments, we studied the impacts of training data set, gateway deployment, and road-map shape on the network performance. The experimental results also showed the superiority of the proposed algorithms to a greedy heuristic algorithm.

Chapter 4

Optimization of Network Throughput and Service Cost

4.1 Introduction

Data transfer in remote monitoring is evaluated by two metrics: network throughput and service cost. Network throughput is defined as the volume of data received by the monitoring center from the WSN within a specified period. Data transfer from each individual sensor node to the remote monitoring center undergoes unreliable wireless transmission in the WSN, and the paid reliable transmission through the third party network. Thus throughput depends on data loss during transmission within the WSN. Service cost is the payment charged for the third party communication service. It is determined by the amount of data carried by the third party network, as well as the charging strategy of the service provider, such as charging rate, charging period. In this chapter, we formulate the network throughput and model the service cost, based on which we study the optimization of these two performance metrics.

Maximizing network throughput and minimizing service cost at the same time is desirable, however, can not be achieved simultaneously. Network throughput is the profit gained from remote monitoring whereas service cost is the expense for the operation of remote monitoring, and intuitively, the more the profit, the higher the expense. For example, if no data is sent through

the third party network, the service cost is zero (at its minimum) yet with nil network throughput. Or, when the network throughput is at its maximum (all sensed data is received at the monitoring center), the incurred service cost would be very high. The trade-off between the two metrics can be found with specified objectives and constraints. Budgeting-oriented applications are with the objective of minimizing service cost provided a required amount of data can be received. Whereas throughput-targeting applications aim at maximizing network throughput with relative relaxing restriction on the service cost. We define optimization problems under these two scenarios, namely service cost minimization with guaranteed network throughput, and network throughput maximization with minimal service cost. We then prove that they are NP-complete and develop algorithms for them separately. We finally provide analytical and experimental results to validate the performance of the proposed algorithms.

The problem of finding a trade-off between the network throughput and the service cost is related to the load-balanced routing problem [24, 42, 57, 78] in that data received at sinks should be balanced to avoid heavy penalty or much waste of pre-paid fixed cost. Hsiao *et al.* [42] introduced dynamic load-balanced tree for grid-topology wireless access networks and proposed a distributed algorithm for it. Dai *et al.* [24] developed a hierarchy-balanced tree and utilized the Chebyshev sum as a metric for evaluating top-level load-balanced trees. Liang *et al.* [57] proposed an algorithm which dynamically builds a spanning tree to balance the data transmission load among nodes and prolong network lifetime. Shan *et al.* [78] studied the load-balanced tree problem by incorporating data delivery delay, and provided a novel top-down heuristic for it. Different from these studies, we consider both service cost and network throughput and aim to design a routing protocol to maximize or maintain the network throughput with the minimal service cost.

Other related problems include the *Capacitated Minimum Spanning Tree prob-*

lem (CMST) [45, 71], the *K-Rooted Min-max Spanning Forest problem* [104], and the *Maximum Concurrent Flow problem* [34], which are separately defined as follows. Given an edge-weighted graph, a root node, a set of *source nodes*, and a given demand $Q > 2$, the CMST problem is to construct a minimum-cost spanning or steiner tree rooted at the root node spanning all source nodes subject to the sum of the demands in the subtree of each child of the root being no more than Q. The K-Rooted Min-max Spanning Forest problem is to find a spanning forest with K given root nodes in an undirected graph such that the maximum tree cost is minimized. The Maximum Concurrent Flow problem is defined in a directed graph, where there are multiple pairs of source and destination nodes with each source having a given demand to be sent to its destination. It is to route the same fractional of the demand of each source to its destination such that the proportion is as large as possible. Intuitively, the problem of concern can be reduced to one of the above mentioned problems. For example, we may reduce the problem to the CMST problem or the K-Rooted Min-max Spanning Forest problem, where the data quota at each sink is treated as the capacity, and a routing tree rooted at the sink subject to its capacity constraint is then built, or a forest is found such that the cost of the maximum cost tree is minimized. Alternatively, we may reduce the problem to the Maximum Concurrent Flow problem, where the volume of data generated by each node for a specified period is treated as the demand of the node and the destinations of all nodes are sinks. However, the problem studied in this chapter is essentially different from these mentioned problems: (i) In the CMST problem, the load allocated to each sink is no more than its capacity, while in our problem the load of some sinks is allowed to exceed the data quotas (capacities). (ii) The K-Rooted Min-max Spanning Forest problem only aims to minimize the maximum cost of trees in a forest, which is not equivalent to our problem where the load of each tree root is determined by not only its edge weights (link reliability) but also the distance of each node in the tree to the tree root (end-to-end

reliability). (iii) We here consider a link-unreliable wireless sensor network where data loss is unavoidable during data transfer, which means that the volume of data collected at each destination node is usually less than the volume of data sent at its source node. Thus, the traditional multi-commodity flow technique is not applicable as the flow conservation property does not hold any more. Overall, existing algorithms are not applicable to our problem, and new algorithms need to be developed.

The remainder of this chapter is organized as follows. Section 4.2 models the unreliable sensor network and formulates the network throughput and service cost. It also proposes two novel optimization problems and proves their NP-hardness. Section 4.3 and 4.4 develop approximation algorithms for service cost minimization and network throughput maximization. Section 4.5 includes experimental results to evaluate the proposed algorithms. Section 4.6 concludes this chapter.

4.2 System Model and Problem Definition

We consider a link-unreliable wireless sensor network $G = (V, E)$, where V is the set of sensor nodes and E is the set of unreliable links, $n = |V|$. Each sensor node is equipped with at least a low-power radio and there is a link between two sensor nodes if they are within the low-power radio transmission range of each other. We have the following assumptions on the network model. (1) Sensor nodes have identical data generation rates r_g and their locations are stationary and known a *priori*. (2) We consider a periodic environmental monitoring application scenario, in which nodes have low data generation rates and thus data burst and bandwidth capacity constraint are not considered in this chapter. (3) The *link reliability* of a link $e \in E$, denoted by p_e with $0 < p_e \leq 1$, is determined by the path loss, concurrent transmission interference, and ambient noise on wireless channels. We assume that the successful probabilities

of any two data transmissions over the same link e are independent, either of which only depends on p_e. (4) Denote by S the set of sinks with $m = |S|$, and each sink has a high-bandwidth radio onboard to transfer data to the third party network. (5) We adopt the tree-based routing structure for data transmission. Let T_i be the tree rooted at sink $s_i \in S$ and $V(T_i)$ be the set of nodes in tree T_i, $1 \leq i \leq m$.

The volume of data received by the monitoring center within a specified period τ is defined as the *network throughput*. Sensed data from a source node must undergo three stages to reach the remote monitoring center. It is first transmitted from its source node to a sink (omitted if it is generated by the sink itself) along a path in the routing tree rooted at the sink, then sent out of the sensor network by the sink, and finally forwarded to the monitoring center by the third party network. The data transmission over wireless links in the WSN is unreliable while the data forwarding over the third party network is considered reliable (paid service and beyond the control of the application user). Thus, the network throughput is the sum volume of data received by all sinks during the period τ. The volume of data received at a sink s_i via T_i within τ is denoted by $D^{(\tau)}(i)$, and the network throughput is $\sum_{i=1}^{m} D^{(\tau)}(i)$. Let $e_1, e_2, \ldots, e_h$ be the link sequence in the path of T_i from a sensor node $v \in V(T_i)$ to sink s_i. Denote by p_{v,s_i} the *end-to-end reliability* between v and s_i, thus $p_{v,s_i} = \prod_{i=1}^{h} p(e_i)$. We treat each attempt of node v sending its data to s_i as *one trial* and each trial as an independent and identically distributed (i.i.d) event. Denote by $D(v, s_i)$ a 0-1 variable to represent whether one trial is successful,

$$D(v, s_i) = \begin{cases} 1 & \text{successful,} \\ 0 & \text{otherwise.} \end{cases} \tag{4.1}$$

Then, $Pr[D(v, s_i) = 1] = p_{v,s_i}$. The expectation of $D(v, s_i)$ is $E[D(v, s_i)] = Pr[D(v, s_i) = 1] \cdot 1 + Pr[D(v, s_i) = 0] \cdot 0 = p_{v,s_i}$. Let $E[D^{(\tau)}(i)]$ be the expected

volume of data received by sink s_i within a charging period τ. Following the definition of Poisson trials [67], the expected volume of data collected by sink s_i in T_i within τ is

$$\begin{aligned} L(s_i) &= E[D^{(\tau)}(i)] = E[\sum_{v \in V(T_i)} (r_g \cdot \tau \cdot D(v, s_i))] \\ &= r_g \cdot \tau \cdot \sum_{v \in V(T_i)} E[D(v, s_i)] = r_g \cdot \tau \cdot \sum_{v \in V(T_i)} p_{v,s_i}. \end{aligned} \tag{4.2}$$

Network throughput $\sum_{i=1}^{m} D^{(\tau)}(i)$ is not deterministic too, and its expectation denoted by $D^{(\tau)}$ is

$$D^{(\tau)} = \sum_{i=1}^{m} E(D^{(\tau)}(i)) = \sum_{i=1}^{m} r_g \cdot \tau \cdot \sum_{v \in V(T_i)} p_{v,s_i} = r_g \cdot \tau \cdot \sum_{v \in V} p_{v,s_i}. \tag{4.3}$$

The remote communication service leased from the third party telecommunication company is charged at each individual sink according to the volume of data it receives and sends. For each sink, the cost within a regular charging period τ is fixed C_f if the amount of data sent via it is no greater than a specified quota Q, otherwise, extra cost is applied with a penalty rate c_p for every MB data usage exceeding Q during τ. Let C_p be the penalty cost of a data quota Q, i.e., $C_p = c_p \cdot Q$. The combinations of different C_f, Q, and c_p comprise optional charging plans, and in most plans, the penalty rate is higher than the rate for data quota, i.e., $c_p > C_f/Q$, and $C_p > C_f$. Thus, if a sink cannot consume the data quota of its current plan within the charging period, it is under-utilizing the current plan and wasting money in some sense, and it should choose another plan with a lower fixed cost for a smaller data quota. On the contrary, if a sink always exceeds the quota, it is wise to choose a plan with a higher fixed cost for a larger data quota to reduce prohibitive penalty.

The service cost of remote monitoring is determined by the number of sinks and the volume of data exceeding the data quota at each sink. In this chapter,

we assume that all the sinks are applied with the same data plan and investigate the total costs of all the sinks within τ. Therefore, we yield the formulated expression for the service cost, denoted by C, as follows.

$$C = m \cdot C_f + \sum_{i=1}^{m} max\{0, (E[D^{(\tau)}(i)] - Q) \cdot c_p\}, \tag{4.4}$$

On the right hand side of Eq. (4.4), the first term is the fixed cost of the m sinks, the second term is the total penalties incurred, either 0 or $(E[D^{(\tau)}(i)] - Q) \cdot c_p$ at sink s_i depending on whether the quota is overused at s_i. From Eq. (4.4), the service cost is determined by the number of sinks, the data plan adopted at individual sinks, and the volume of data sent through each sink.

Given a link-unreliable sensor network $G(V, E)$ for remote monitoring with a specified data plan adopted by each sink for a charging period of τ, the trade-off between network throughput and service cost is considered in the following two scenarios. (1) Budget-oriented remote monitoring applications with homogeneous cheap nodes deployed in the WSN have service cost as their top concern, while still requiring a certain amount of data to be received by the monitoring center. In these applications, the service cost is to be minimized while the specified network throughput requirement has to be met. We define the network throughput requirement as $D_{req}^{(\tau)} = \alpha \cdot n \cdot r_g \cdot \tau$, where $0 < \alpha \leq 1$ is a pre-defined parameter, referred to as the *network throughput threshold*, a lower bound on the percentage of all sensed data that must be received by the monitoring center during a charging period τ. To meet the specified throughput requirement, $D^{(\tau)} \geq D_{req}^{(\tau)}$ should hold. (2) High-precision applications aim to acquire detailed information of the monitoring region thus require to receive as much real-time data as possible from the monitoring region. Clearly, the network throughput is the first objective of these applications, which would not spare to deploy a number of resource-unconstrained sinks at pre-defined locations in the monitoring region with the hope that they can be more capable

to undertake crucial role in remote data transfer to improve network throughput. Under this circumstance, the service cost is not the foremost concern yet is desirable to be as low as possible provided the throughput is the maximum.

Two performance optimization problems are defined accordingly as follows: (1) In homogeneous dual-radio WSNs which require the monitoring center to receive at least α percentage of all sensed data within τ, the *throughput guaranteed service cost minimization problem* is to identify a set of sinks for each period of τ and find a forest of routing trees to transmit the data generated within τ to the sinks, such that the incurred service cost is minimized, subject to the throughput requirement. (2) In heterogeneous WSNs with a given set of dual-radio sinks S with $m = |S|$, the *network throughput maximization with minimal service cost problem* is to design a data routing strategy such that the accumulative volume of data received by the m sinks is maximized while the service cost of transferring this amount of data to the remote monitoring center is minimized. Different from the first problem, where the throughput is a constraint imposed on the objective of minimizing the service cost, the second problem has throughput maximization as the foremost aim, a precondition for minimizing service cost.

Providing efficient solutions to the above constrained optimization problems is challenging. The core difficulty lies in jointly determining the set of sinks and find a routing forest consisting of trees rooted at the chosen or pre-identified sinks, via which (i) the total volume of data received at all roots is maximized or meets the throughput requirement (ii) the volumes received at roots are balanced to eliminate penalty due to quota exceeding, or minimize penalty at any sink if the quota exceeding occurs. (iii) adequate data can be received at each root to avoid the waste of pre-paid fixed cost, which happens if the received volume is always below the quota.

Theorem 4.1. *The decision versions of the two performance optimization problems in network $G(V, E)$ are NP-complete.*

Proof. We first prove that the throughput guaranteed service cost minimization problem is NP-complete. We show a special case of the problem is NP-complete through a reduction from a NP-complete problem, the subset sum problem [22]. Given a set of positive integers $U = \{a_1, a_2, \ldots, a_n\}$, the subset sum problem is to partition the set into two disjoint subsets U_1 and U_2 such that $\sum_{a_i \in U_1} a_i = \sum_{a_j \in U_2} a_j$. The decision version of an instance of the subset sum problem is to determine whether there is a set partition U' and $U'' = U - U'$ such that $\sum_{a_i \in U'} a_i = \sum_{a_j \in U''} a_j = A$, where $A = \sum_{a_i \in U} a_i/2$.

Given an instance of the subset sum problem, we now construct an instance of a special maximizing network throughput with minimal remote data transfer cost problem in a sensor network $G(V \cup \{s_1, s_2\}, E)$ with two sinks s_1 and s_2 as follows. V is the set of sensors and each sensor $v_i \in V$ corresponds to an element in U, and sinks s_1 and s_2 correspond to sets U' and U'' respectively. There is an edge in E between each sensor node and either sink or between two sensors if they are within the transmission range of each other. Assume that the link reliability of a link between a sensor v_i and either sink is $p_i = a_i/\tau$, where $\tau = max\{\, a_i \mid 1 \leq i \leq n\} + 1$ is the duration of a monitoring period, while the link reliability between any two sensors is 1. We further assume that the data generation rate is $r = 1$. The amount of data from a sensor $v_i \in V$ to either sink during this period is $p_i \cdot \tau \cdot r$. Clearly, the maximum network throughput of this network is $\sum_{v_i \in V_1}(p_i \cdot \tau \cdot r) + \sum_{v_j \in V_2}(p_j \cdot \tau \cdot r) = \sum_{a_i \in U} a_i = 2A$. Let the data quota Q be A, i.e., $Q = A$, and the fixed cost is C_f. The decision version of this special case of the problem is to determine whether there are two routing trees rooted at the sinks such that the network throughput is maximized while the remote data transfer cost is minimized, i.e., $2C_f$. Note that only when the volume of data received at each root of the two trees is A, the remote data transfer cost is $2C_f$, otherwise, the cost will be larger than $2C_f$. Obviously, if there is a solution to the problem, there is a corresponding solution to the subset sum problem. Since the subset sum problem is NP-

complete and this reduction is in polynomial time, the problem of concern thus is NP-hard. Meanwhile, it is easy to verify whether a given solution has the the maximum network throughput $2A$ with remote data transfer cost $2C_f$ in polynomial time, thus the problem is in NP class. As one of its special case is NP-complete, the problem is NP-complete.

The NP-completeness proof of the network throughput maximization with minimal service cost problem is similar to that of the throughput guaranteed service cost minimization problem, omitted. □

4.3 Service Cost Minimization

We first deal with the throughput guaranteed service cost minimization problem. Due to its NP-completeness, we propose a heuristic instead. Given a specific data plan for sinks, the service cost in a charging period is determined by the number of sinks, and penalties for quota exceeding at individual sinks. Intuitively, if there is a solution to the problem, the minimum service cost can be achieved when no fixed cost is wasted and no penalty is applied. That is, the amount of data relayed by each sink within the charging period is exactly equal to its data quota. However, in reality, the volumes of data relayed by sinks may not be balanced, which will result in money waste at some sinks that relay data less than their quotas, and penalties at some others that relay data more than their quotas. In order to minimize the service cost while maintaining the throughput requirement, the proposed heuristic needs to identify an appropriate number of sinks, and design routing trees rooted at these sinks spanning the rest of nodes such that the sum of costs for relaying data collected from individual trees is minimized while the expected total volume of data collected from all these trees meets the throughput requirement.

The number of sinks has a great impact on service cost, and we aim to find the one corresponding with the minimum cost. On the one hand, a small

number of sinks means the total fixed cost is relatively low, however, the quotas might be severely overused at some sinks and expensive penalties will be applied. If the penalty at a sink is greater than the fixed cost, it is worth employing another one or more sinks to share the workload, thereby reducing the chance of quota exceeding at sinks and decreasing the service cost. On the other hand, a large number of sinks means that each sink has less data to relay and small penalties or no penalty will be applied. However, the data quotas at some sinks may be severely underutilized, and a large fraction of the fixed cost charged at these sinks will be wasted. If the volume of data relayed by a sink can be redistributed to some other sinks without causing any quota exceeding among the involved sinks, this sink can be removed and the fixed cost will be saved accordingly. Therefore, an appropriate number of sinks is to be found to fully utilize data quotas at sinks and avoid penalties to the largest extent.

When a profitable number of sinks is determined, we need to address which nodes should become the sinks. The choice of sinks is guided by the following rationale. Sinks should have relatively high residual energy, because they relay data over the high-bandwidth radios and thus consume much more energy than the other nodes. If a node with low residual energy serves as a sink, the energy imbalance will be aggravated and the network lifetime will be significantly shortened. Once the sinks are identified, we construct routing trees rooted at the sinks spanning all the other nodes in the network to balance the energy consumption overall the network.

4.3.1 Routing Trees Establishment

For the purpose of convenience, we now assume that the number of sinks is given as m with $1 \leq m \leq n$, and deal with the problem of identifying m sinks from all sensor nodes and finding the m routing trees rooted at the identified sinks. We will remove this assumption later.

To select m sinks from n nodes in V, we first sort the n nodes by their residual energy in non-increasing order. Denote by $er(v)$ the current residual energy of node v. Let $v'_1, v'_2, \ldots v'_n$ be the sorted node sequence, where $er(v'_i) \geq er(v'_j), 1 \leq i < j \leq n$. Instead of directly choosing the first m nodes from the sequence, we select the first $m' = \lceil n \cdot \beta \rceil > m$ nodes and then randomly pick up m nodes from the m' ones to form the set of sinks S, where $0 < \beta \leq 1$ is a pre-defined parameter referred to as the *search space percentage*. The rationale behind is that choosing sinks only according to the residual energy regardless of their locations in the monitoring region may result in poor network throughput no matter how data routing is designed. For example, if the first m nodes in the sequence are all located close to each other in the corner in the monitoring region (which is likely to happen in a WSN), the data generated from nodes in the diagonal corner, no matter how to be routed, have to go through a long distance wireless transmission, causing high probability of data loss, compromising the network throughput. A greater value of β increases the probability of a better distribution yet increases the possibility of choosing a node with less residual energy as a sink too.

What follows is to find a forest that consists of routing trees rooted at the m sinks in S, spanning all the other nodes in $V \setminus S$. To this end, we first construct a weighted, directed graph $G_d = (V', E', \omega)$, where $V' = V \cup \{s_v\}$ and node s_v is a virtual super sink, $E' = \{\langle v, u\rangle, \langle u, v\rangle \mid (v, u) \in E\} \cup \{\langle s_v, s_i\rangle \mid s \in S\}$. That is, the virtual super sink s_v is only connected to the m sinks and each of such links is assigned a weight of $\omega(\langle s_v, s_i\rangle) = 0$ for any $s_i \in S$. For the weights of other edges in E', we incorporate both the link reliability and the residual energy of sensor nodes into consideration. Specifically, for a directed edge $\langle v, u\rangle$,

$$\omega(\langle v, u\rangle) = IE \cdot a^{1-er(v)/IE}/p(v, u) \text{ [57]}, \tag{4.5}$$

where IE is the initial energy of each node, $p(u, v)$ is the link probability of its corresponding edge $(v, u) \in E$, and $a > 1$ is a positive constant determining the impact of residual energy on the weight, referred to as the *weight adjustment parameter*. Note that all outgoing edges from a node v have identical weights, and for each edge $(v, u) \in E$, there are two directed edges $\langle v, u \rangle$ and $\langle u, v \rangle$ in E' with asymmetric weights. Initially, $er(v) = IE$ at each node $v \in V$. As nodes consume their energy, the residual energy of each node becomes smaller and the weights of outgoing edges from the node increase. The less the energy left at node v, the higher the weights of its outgoing edges. Having the auxiliary graph G_d, we now describe the construction of routing trees in G_d.

A single-source shortest path tree F in G_d rooted at s_v is constructed, using Dijkstra's algorithm [22]. Let $L(v, u)$ be the shortest path from node v to node u in graph G_d. In path $L(s_v, v)$ from the virtual super sink s_v to any node $v \in V$, the first and second vertex are respectively s_v and a sink in S. That is because the super sink can only access to sinks thus the path from s_v to any other node must be via a sink in S. After the removal of s_v and its incident edges from tree F, a forest $\mathcal{F} = \{T_i \mid 1 \leq i \leq m\}$ consisting of routing trees rooted at the m sinks then follows. For any node $v \in V(T_i)$, it sends its sensed data to sink s_i along the reverse path $L(s_i, v)$ in T_i. The expected network throughput is calculated by Eq. (4.3) and the service cost is calculated by Eq. (4.4). The algorithm is referred to as `Iden_Sink` and its detailed description is given in **Algorithm 5**.

4.3.2 Determination of the Optimal Number of Sinks

So far we have assumed that the number of sinks m is given, we now remove this assumption and propose an algorithm for the problem of concern as follows. We focus on finding an appropriate value of m to minimize the service cost. Consider a scenario in which the total volume of generated data is evenly

Algorithm 5: `Iden_Sink`

Input : $G(V,E)$, m, τ, β, r_g, C_f, c_p, Q
Output: The network throughput, the routing forest $\mathcal{F}$, and the service cost C

/* Stage 1: identify m sinks */
Let $v'_1, v'_2, \ldots v'_n$ be the sorted node sequence in V in non-increasing order of residual energy;
$m' \leftarrow \lceil n \cdot \beta \rceil$ $(> m)$;
Select m nodes from the first m' nodes randomly as the sinks in S;
/* Stage 2: building routing trees rooted at the m sinks */
Construct a weighted directed graph $G_d(V', E', \omega)$, with $V' = V \cup \{s_v\}$ and $E' = \{\langle v,u\rangle, \langle u,v\rangle | (v,u) \in E\} \cup \{\langle s_v, s\rangle \mid s \in S\}$;
Find a single-source shortest path tree F in G_d rooted at s_v;
Remove node s_v and all edges incident to it from F and obtain m routing trees rooted at the sinks in S, T_i with $1 \leq i \leq m$;
Calculate the network throughput $\sum_{i=1}^{m} E[D^{(\tau)}(i)]$ according to Eq.(4.3);
Calculate the service cost C according to Eq.(4.4);
return $\sum_{i=1}^{m} E[D^{(\tau)}(i)]$, $\mathcal{F} = \{T_i \mid 1 \leq i \leq m\}$, *and* C.

distributed to (and relayed by) m_0 sinks, where $m_0 = \lfloor \frac{\alpha \cdot n \cdot \tau \cdot r_g}{Q} \rfloor$, and the expected volume of data collected by the m_0 sinks meets the throughput requirement. This will lead to the minimum service cost $C_{opt} = m_0 \cdot C_f$ because the data quota at every sink is fully utilized and no data exceeding occurs. However, such a solution may never exist in real remote monitoring applications because the volume of data relayed by the m_0 sinks may never be balanced due to the network topology and link reliability. Thus, a larger or smaller number of sinks than m_0 may result in a lower service cost. In the following, we develop a greedy heuristic to deliver a solution such that its corresponding service cost is the minimum among the found solutions while the throughput requirement is met.

The proposed heuristic proceeds iteratively in the two intervals $[1, m_0]$ and $[m_0 + 1, n]$ separately. We first search the appropriate value of m in the interval $[1, m_0]$ by setting $m = m_0$ and decreasing its value by one in each iteration. Within each iteration, the algorithm `Iden_Sink` is first called with the current

value of m as the input, and a solution with a corresponding service cost will be obtained. Then the cost will be checked to see whether it is the minimum one among all found solutions so far. If not, the procedure terminates and a solution is found, otherwise, the value of m is decreased by 1 and the procedure continues. We then start from $m = m_0 + 1$, increase the value of m by one in each iteration, and repeat the above process. In the end, a feasible solution to the problem with a service cost will be delivered. The detailed description of algorithm `Min_Cost` is in **Algorithm 6**.

Theorem 4.2. *Given a homogeneous dual-radio sensor network $G(V, E)$ with unreliable link reliability, algorithm* `Min_Cost` *solves the throughput guaranteed service cost minimization problem with complexity $O(n^3)$, where $n = |V|$ is the number of nodes in G.*

Proof. With a given number of sinks, identifying them from the n nodes takes $O(|E|)$ time [37], while finding routing trees rooted at the sinks takes $O(|E| + |V| \log |V|) = O(n^2)$ time [22]. The number of iterations for searching the appropriate number of sinks is no more than n. Thus, the computational complexity of algorithm `Min_Cost` is $O(n^3)$. □

4.3.3 Theoretical Analysis

Since wireless communications in the sensor network are unreliable, the actual volume of data received by each sink may not always be equal to its expected data volume. And this would cause the actual network throughput below the requirement, or the actual service cost beyond the one delivered by algorithm `Min_Cost`. In the following we analyze the probabilities that such events happen in the solution delivered by the proposed algorithm.

We first analyze the probability that the actual volume of data received by all sinks in the forest $\mathcal{F}$, $\sum_{i=1}^{m} D^{(\tau)}(i)$, is less than the throughput requirement $D_{req}^{(\tau)} = \alpha \cdot r_g \cdot \tau \cdot n$. We refer to this probability as the *throughput failure*

Algorithm 6: `Min_Cost`

Input : $G(V,E)$, τ, β, r_g, C_f, c_p, Q, $D_{req}^{(\tau)}$
Output: Routing forest and service cost

$m_0 \leftarrow \lfloor \frac{\alpha \cdot n \cdot \tau \cdot r_g}{Q} \rfloor$; $C \leftarrow \infty$ /* the initial service cost */ ;
/* search the interval $[1, m_0]$ */;
$m \leftarrow m_0$; $loop \leftarrow' true'$;
while ($m \geq 1$ *and* $loop$) **do**
  call algorithm `Iden_Sink` $(G, m, \tau, \beta, r_g, C_f, c_p, Q)$;
  $E(D(m)) \leftarrow E(D^{(\tau)})$; $\mathcal{F}(m) \leftarrow \mathcal{F}$; $C(m) \leftarrow C$;
  if $E(D(m)) \geq D_{req}^{(\tau)}$ **then**
    if $C(m) < C$ **then**
      $C \leftarrow C(m)$; $\mathcal{F} \leftarrow \mathcal{F}(m)$;
    else
      $loop \leftarrow' false'$;
  $m \leftarrow m - 1$;
/* search the interval $[m_0 + 1, n]$ */;
$m \leftarrow m_0 + 1$; $loop \leftarrow' true'$;
while ($m \leq n$ *and* $loop$) **do**
  call algorithm `Iden_Sink` $(G, m, \tau, \beta, r_g, C_f, c_p, Q)$;
  $E(D(m)) \leftarrow E(D^{(\tau)})$; $\mathcal{F}(m) \leftarrow \mathcal{F}$; $C(m) \leftarrow C$;
  if $E(D(m)) \geq D_{req}^{(\tau)}$ **then**
    if $C(m) < C$ **then**
      $C \leftarrow C(m)$; $\mathcal{F} \leftarrow \mathcal{F}(m)$;
    else
      $loop \leftarrow' false'$;
  $m \leftarrow m + 1$;
return $\mathcal{F}$ *and* C.

probability, denoted by $Pr[\sum_{i=1}^{m} D^{(\tau)}(i) < D_{req}^{(\tau)}]$. We assume that the expected volume of data received by all sinks in $\mathcal{F}$ is greater than the network throughput requirement, i.e., $E[\sum_{i=1}^{m} D^{(\tau)}(i)] > D_{req}^{(\tau)}$. This assumption indicates that the network throughput requirement is met most of the time, and we have the following lemma.

Lemma 4.3. *Given the network throughput requirement $D_{req}^{(\tau)}$ in any period of τ, the*

probability that the volume of data collected by all sinks within τ, $\sum_{i=1}^{m} D^{(\tau)}(i)$, *is less than* $D_{req}^{(\tau)}$ *is no more than* $e^{-\mu_{\mathcal{F}}(1-D_{req}^{(\tau)}/\mu_{\mathcal{F}})^2/2}$, *where* $\mu_{\mathcal{F}} = E[\sum_{i=1}^{m} D^{(\tau)}(i)]$.

Proof. Let $\delta = 1 - \frac{D_{req}^{(\tau)}}{\mu_{\mathcal{F}}}$. We have

$$\begin{aligned} Pr[\sum_{i=1}^{m} D^{(\tau)}(i) < D_{req}^{(\tau)}] &= Pr[\sum_{i=1}^{m} D^{(\tau)}(i) < (1-\delta)\mu_{\mathcal{F}}] \\ &< \left(\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}}\right)^{\mu_{\mathcal{F}}} \text{ (by the Chenorff bound [67])} \\ &< e^{\frac{-\mu_{\mathcal{F}}\delta^2}{2}}. \end{aligned} \tag{4.6}$$

By substituting δ in Eq.(4.6) with $1 - \frac{D_{req}^{\tau}}{\mu_{\mathcal{F}}}$, we have

$$Pr[\sum_{i=1}^{m} D^{(\tau)}(i) < D_{req}^{(\tau)}] < e^{-\mu_{\mathcal{F}}(1-D_{req}^{(\tau)}/\mu_{\mathcal{F}})^2/2}.$$

□

Note that the throughput failure probability only depends on the value of $D_{req}^{(\tau)}$, because once the routing trees are identified, the end-to-end reliability between any node to its sink is determined. The larger the value of $D_{req}^{(\tau)}$, the greater the throughput failure probability.

We then analyze the probability that the actual service cost is greater than the service cost in the solution delivered by algorithm `Min_Cost`. We refer to this probability as *the cost exceeding probability*, denoted by $Pr[C^{(\tau)} \geq (1+\theta)C]$ where $\theta > 0$ and $C^{(\tau)}$ is the actual service cost, $C^{(\tau)} = m \cdot C_f + \sum_{i=1}^{m} max\{0, (D^{(\tau)}(i) - Q) \cdot c_p\}$. The actual service cost $C^{(\tau)}$ depends on the value of $D^{(\tau)}(i)$, the actual volume of data received by each sink $s_i \in S$ within a period of τ, while the value of $D^{(\tau)}(i)$ depends only on the structure of the tree $T_i \in \mathcal{F}$. Any two values of $D^{(\tau)}(i)$ and $D^{(\tau)}(j)$ are independent of each other when $i \neq j$. Therefore, the actual service cost of each sink can be considered as an *i.i.d* random variable. Following the Chenorff bound [67], the probabil-

ity that the actual service cost $C^{(\tau)}$ is greater than the service cost C is no more than $(\frac{e^{\theta}}{(1+\theta)^{(1+\theta)}})^{C}$, and two special cases of this general setting can be derived: (i) when $0 < \theta \leq 1$, $Pr[C^{(\tau)} \geq (1+\theta) \cdot C] \leq e^{-C \cdot \theta^2/3}$; (ii)when $\theta \geq 5$, $Pr[C^{(\tau)} \geq (1+\theta) \cdot C] \leq 2^{-(1+\theta) \cdot C}$. We therefore conclude that a large value of θ will lead to a relatively small cost exceeding probability, which is consistent with the intuition.

From the above analysis, we have shown that with the solution delivered by algorithm `Min_Cost`, both the throughput failure probability and cost exceeding probability are bounded.

4.4 Network Throughput Maximization

We now solve the network throughput maximization with minimal service cost problem. Specifically, we first consider a special case of networks with uniform link reliability and devise an algorithm to achieve the maximum network throughput with bounded service cost. We then remove the uniform reliability assumption and deal with the general WSNs with non-uniform link reliability by revising and expanding the algorithm for the special case.

4.4.1 Algorithm for Special Networks with Uniform Link Reliability

In a special network with uniform reliability on links, every link has identical reliability p with $0 < p \leq 1$, for which we devise a novel approximation algorithm that can maximize the network throughput with bounded service cost.

We start with some notations. Given the network $G(V \cup S, E)$, we construct another network $G' = (V \cup S \cup \{s_v\}, E')$ as follows. A virtual super sink s_v and an edge between s_v and each sink $s \in S$ are added to G', i.e., $E' = E \cup$

$\{(s_i, s_v) \mid s_i \in S\}$. Let T^{BFS} be a Breadth-First Search tree in G' rooted at the super sink s_v with the depth h, where the virtual sink s_v is in layer 0 and all sinks are in layer 1 of T^{BFS}. Let V_l be the set of nodes of T^{BFS} in layer l for all l with $0 \leq l \leq h$. Then, the nodes in $V \cup S$ are partitioned into h disjoint subsets $V_1, V_2, \ldots, V_h$ such that $V_1 = S$, $\cup_{l=2}^{h} V_l = V$, and $V_i \cap V_j = \emptyset$ if $i \neq j$ and $1 \leq i, j \leq h$. Note that $l-1$ is the minimum number of hops in G from $v \in V_l$ to its nearest sink in S, with $2 \leq l \leq h$.

Lemma 4.4. *Given an unreliable sensor network $G(V \cup S, E)$ with identical link reliability p, $0 < p \leq 1$, (i) for each node $v \in V$, let $h(v) - 1$ be the minimum number of hops from v to its nearest sink in G, then the end-to-end reliability of the most reliable path in G from v to a sink is equal to $p^{h(v)-1}$. (ii) The maximum network throughput is $D_{max} = \sum_{l=2}^{h} \sum_{v \in V_l} p^{l-1} \cdot (\tau \cdot r_g) = \sum_{l=2}^{h} (|V_l| \cdot p^{l-1} \cdot \tau \cdot r_g)$.*

Proof. (i) Since the reliability of each link is p, let P_v be the most reliable path from a sensor $v \in V$ to a sink and $|P_v| = l - 1$, then, the reliability of path P_v is p^{l-1}. Maximizing the end-to-end reliability of path P_v, p^{l-1}, is equivalent to minimizing the value of l, while the length of a shortest path from v to any sink is no less than $h(v) - 1$, following the definition of $h(v)$. Thus the end-to-end reliability of the most reliable path in G from v to a sink is $p^{h(v)-1}$.

(ii) For each node $v \in V$ with $h(v) = l$, its contribution to the maximum network throughput for a period of τ is identical, which is $d_l = d_{max}^v = p^{l-1} \cdot (\tau \cdot r_g)$. Maximizing d_l is equivalent to minimizing l, while $l \geq h(v)$, l is equal to $h(v)$. Thus $D_{max} = \sum_{v \in V} d_{max}^v = \sum_{l=2}^{h} (|V_l| \cdot p^{l-1} \cdot \tau \cdot r_g)$. □

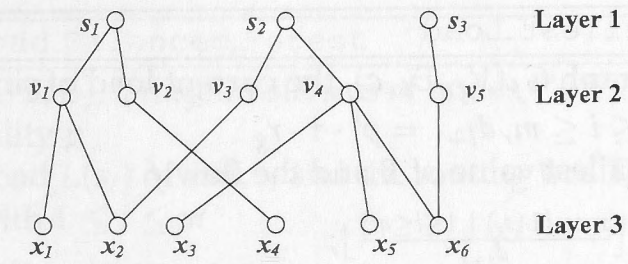
From Lemma 4.4, it can be seen that the forest consisting of routing trees rooted at the sinks, derived by the removal of the virtual sink s_v and its adjacent edges from T^{BFS}, can deliver the maximum network throughput. However, the service cost for this amount of network throughput may not be the minimum. To minimize the service cost, the sinks should fully utilize their data quotas to avoid data exceeding penalties, while keeping the maximum

throughput unchanged. To this end, we first construct a forest consisting of load-balanced shortest path trees by employing the maximum flow technique, where the length of a 'shortest' path is the minimum number of hops from a source node to its the sink. We then readjust the load among the sinks through modifying the routing trees to further reduce the service cost.

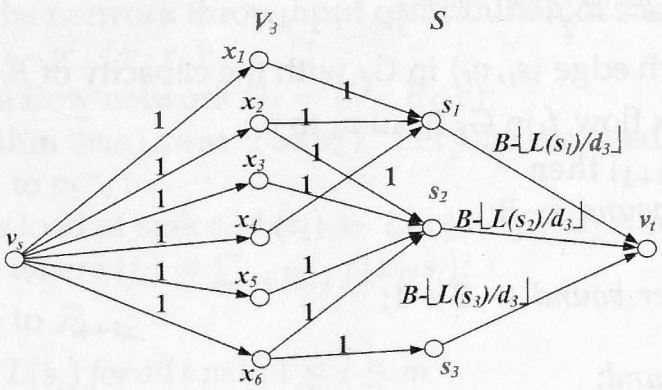
Load-balanced forest establishment. We first construct a forest $\mathcal{F}$ consisting of m routing trees rooted at m sinks such that load of sinks is well balanced while the maximum network throughput is maintained.

The proposed algorithm proceeds the forest construction iteratively. Within each iteration, a level expansion of each tree is conducted. Let $\mathcal{F}_l$ be the forest which spans the nodes in the first l layers. Initially, $\mathcal{F}_1$ spans all sinks $s_1, s_2, \ldots, s_m \in S$ and the load of each sink $L(s_i)$ is 0, for all i with $1 \leq i \leq m$. Assuming that the forest $\mathcal{F}_l$ containing the nodes in the first l layers has been built, we now expand the forest from $\mathcal{F}_l$ to $\mathcal{F}_{l+1}$ by including the nodes in V_{l+1} with the objective that the maximum load among the m sinks in the resulting forest $\mathcal{F}_{l+1}$ is minimized. To this end, we adopt the maximum flow technique as follows. We construct a flow network $G_f = (V_f, E_f, c)$, where $V_f = \{v_s, v_t\} \cup \{s_1, s_2, \ldots, s_m\} \cup V_{l+1}$, v_s and v_t are the source node and the destination node, and c is the link capacity function. Let $V_{l+1} = \{x_1, x_2, \ldots, x_{|V_{l+1}|}\}$. There is a directed edge in E_f from v_s to each node $x_j \in V_{l+1}$ with capacity of 1, and there is a directed edge from each sink s_i to the destination node v_t with capacity of non-negative integer $B_i = B - \lfloor \frac{L(s_i)}{d_{l+1}} \rfloor$, where $d_{l+1} = p^l \cdot \tau \cdot r_g$ is the amount of data that can be collected at a sink from any node in V_{l+1} and B will be defined later. If a node $x_j \in V_{l+1}$ has at least one edge with the nodes in the tree rooted at s_i, there is an edge in E_f from node x_j to sink s_i with capacity of 1. Fig. 4.1 illustrates the construction of G_f through an example.

Minimizing the maximum load among the sinks is equal to minimizing the value of B for a maximum flow in G_f from v_s to v_t with value of $|V_{l+1}|$ (i.e., each node in V_{l+1} will be attached to one of the m routing trees), where the



(a) A forest containing nodes in the first two layers, and edges in G between nodes in layer 2 and 3



(b) A flow network G_f is constructed to include the nodes in layer 3

Figure 4.1: The construction of $G_f(V_f, E_f, c)$

value of B is in the range between $\lfloor \frac{\max\{L(s_i)|1 \le i \le m\}}{d_{l+1}} \rfloor$ and $\lfloor \frac{\max\{L(s_i)|1 \le i \le m\}+|V_{l+1}|\cdot d_{l+1}}{d_{l+1}} \rfloor = \lfloor \frac{\max\{L(s_i)|1 \le i \le m\}}{d_{l+1}} \rfloor + |V_{l+1}|$. The smallest B can be found by algorithm `Smallest_Load`, detailed in **Algorithm 7**.

With the smallest value of B, a flow f of value $|f| = |V_{l+1}|$ from v_s to v_t is obtained. Let $|f_i| = \sum_{x_j \in V_{l+1}} f(x_j, g_i)$ be the total amount of flow entering sink s_i, then the corresponding nodes in V_{l+1} will join tree T_i and the load $L(s_i)$ of T_i is updated to $L(g_i) + |f_i| \cdot d_{l+1}$ for all i with $1 \le i \le m$. Forest $\mathcal{F}_l$ now has been expanded to $\mathcal{F}_{l+1}$. The algorithm `Load_Balanced_Forest` for constructing a forest consisting of load-balanced routing trees is described in **Algorithm 8**.

Lemma 4.5. *Given an unreliable sensor network $G(V \cup S, E)$, the layer expansion from forest $\mathcal{F}_l$ to forest $\mathcal{F}_{l+1}$ by algorithm* `Load_Balanced_Forest` *is optimal in*

Algorithm 7: `Smallest_Load`

Input : Flow graph $G_f(V_f, E_f, c)$, the current load of sink $L(s_i)$ for all i with $1 \leq i \leq m$, $d_{l+1} = p^l \cdot \tau \cdot r_g$
Output: The smallest value of B and the flow f

$lower\_bound \leftarrow \lfloor \frac{\max\{L(s_i) \mid 1 \leq i \leq m\}}{d_{l+1}} \rfloor$;
$upper\_bound \leftarrow \lfloor \frac{\max\{L(s_i) \mid 1 \leq i \leq m\}}{d_{l+1}} \rfloor + |V_{l+1}|$;
while $(lower\_bound \neq upper\_bound)$ **do**
  $B \leftarrow \lfloor \frac{lower\_bound + upper\_bound}{2} \rfloor$;
  Assign each edge $\langle s_i, v_t \rangle$ in G_f with the capacity of $B_i = B - \lfloor \frac{L(s_i)}{d_{l+1}} \rfloor$;
  Find a max flow f in G_f from v_s to v_t;
  if $|f| = |V_{l+1}|$ **then**
    $upper\_bound \leftarrow B$;
    else
      $lower\_bound \leftarrow B + 1$;
$B \leftarrow upper\_bound$;
return B *and* f.

terms of minimizing the maximum data load among the m sinks, $1 \leq l \leq h-1$, *i.e., the maximum flow* f *with* $|f| = |V_{l+1}|$ *in the auxiliary graph* G_f *from* v_s *to* v_t *with the minimum* B *is an optimal solution.*

Proof. It is obvious that the new maximum load among the m sinks is between $\max\{L(s_i) \mid 1 \leq i \leq m\}$ and $\max\{L(s_i) \mid 1 \leq i \leq m\} + d_{l+1} \cdot |V_{l+1}|$. As each sensor node $v \in V_{l+1}$ will contribute exactly the same volume of data d_{l+1} to a sink, finding a maximum flow with value $d_{l+1} \cdot |V_{l+1}|$ is equivalent to finding a maximum flow in G_f from v_s to v_t with value $|V_{l+1}|$ under the capacity B. □

Dynamic load readjustment. The constructed forest consisting of load-balanced trees by algorithm `Load_Balanced_Forest` can be improved in terms of the cost, through further balancing the load among the routing trees. Fig. 4.2 shows an example of load-unbalanced trees constructed by algorithm `Load_Balanced_Forest`. There are two sinks s_1 and s_2, and each of the nodes v_1, v_2, v_3, v_4 in the second layer is within the transmission ranges of s_1 and s_2.

Algorithm 8: `Load_Balanced_Forest`

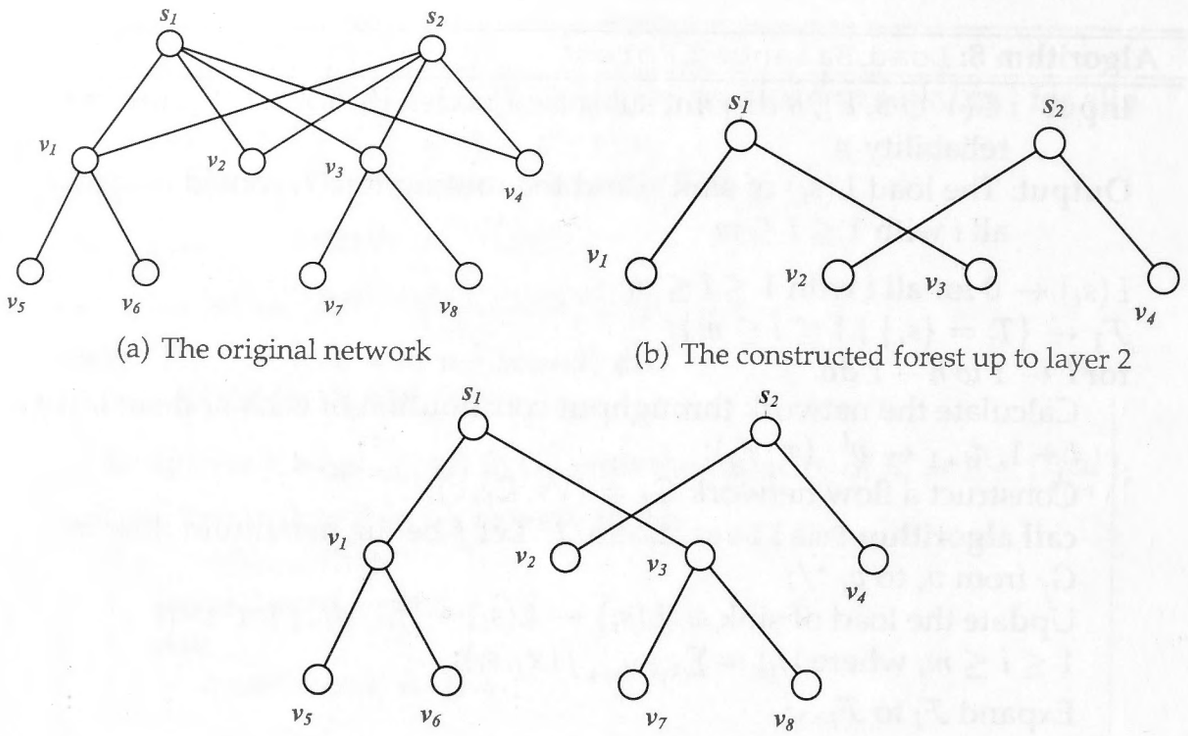
Input : $G(V \cup S, E)$, h disjoint subsets of nodes $V_1, V_2, \ldots, V_h$, the link reliability p

Output: The load $L(s_i)$ of sink s_i and the routing tree T_i rooted at s_i for all i with $1 \leq i \leq m$

$L(s_i) \leftarrow 0$ for all i with $1 \leq i \leq m$;
$\mathcal{F}_1 \leftarrow \{T_i = \{s_i\} \mid 1 \leq i \leq m\}$;
for $l \leftarrow 1$ *to* $h-1$ **do**
  Calculate the network throughput contribution of each node in layer $l+1$, $d_{l+1} \leftarrow p^l \cdot (\tau \cdot r_g)$;
  Construct a flow network $G_f = (V_f, E_f, c)$;
  call algorithm `Smallest_Load`, /* Let f be the maximum flow in G_f from v_s to v_t */;
  Update the load of sink s_i, $L(s_i) \leftarrow L(s_i) + |f_i| \cdot d_{l+1}$ for each $1 \leq i \leq m$, where $|f_i| = \sum_{x_j \in V_{l+1}} f(x_j, s_i)$;
  Expand $\mathcal{F}_l$ to $\mathcal{F}_{l+1}$;
return $\mathcal{F}_h$ *and* $L(s_i)$ *for all* i *with* $1 \leq i \leq m$.

Fig. 4.2(b) shows the load is balanced among the sinks when the forest spans the nodes in the first two layers. However, when it further expands to include nodes v_5, v_6, v_7, v_8 in the third layer, all of these nodes are included in the tree rooted at sink s_1 by algorithm `Load_Balanced_Forest`. This causes severe load imbalance between sinks s_1 and s_2, where $L(s_1) = (2 \cdot p + 4 \cdot p^2) \cdot \tau \cdot r_g$ while $L(s_2) = 2 \cdot p \cdot \tau \cdot r_g$. The reason behind is that no more information below the current layer is available when expanding a new layer.

Having the forest delivered by algorithm `Load_Balanced_Forest`, we then readjust the load among the sinks dynamically to further reduce the service cost. To this end, we consider the following cases. If the load of each sink is no greater than or no less than the data quota Q, we do nothing because the service cost is already the minimum one. Otherwise, the cost can be improved through readjusting the load among the m sinks through a series of *edge swapping* that replaces a tree edge by a non-tree edge while the network throughput is still unchanged. To this end, not all but only certain types of non-tree edges



(a) The original network

(b) The constructed forest up to layer 2

(c) The found forest consists of nodes in 3 layers

Figure 4.2: An example of a forest to be improved in terms of the service cost

can be swapped with the tree edges, which is stated by the following lemma.

Lemma 4.6. *Given an unreliable sensor network $G(V \cup S, E)$ with uniform link reliability p and a forest consisting of load-balanced routing trees rooted at the sinks, the load adjustment among the sinks can be achieved through a series of edge swapping between tree edges and non-tree edges. Specifically, for all l with $1 \leq l \leq h-1$, the use of any edge in G between V_l and V_{l+1} will not change the network throughput obtained by the current forest; otherwise the use of any other edge will reduce the obtained network throughput.*

Proof. Let (u, v) be a tree edge with $v \in V_l$ and $u \in V_{l+1}$. Let $L(u)$ be the volume of data collected at node u from all its descendant nodes in the subtree rooted at u, and this volume of data is then forwarded to the sink that u is attached, the resulting volume of collected data at the sink derived from node u is $p^l \cdot L(u)$. Now, if we remove this tree edge (u, v) from the tree and instead

add another non-tree edge (u, v') to form a new tree edge where $v' \in V_l$, then the volume of collected data of a sink derived from the nodes in the subtree rooted at u is still $p^l \cdot L(u)$, which implies that the accumulative volume of collected data at all sinks does not change through this type of edge swapping.

Now, consider a non-neighboring non-tree edge (u, v'') with $v'' \notin V_l$ if it does exist, then the depth $h(v'')$ of node v'' must be equal to or greater than $l+1$ because its depth is obtained through the BFS traversal, i.e., either $h(v'') = l+1$ or $h(v'') > l+1$, and the amount of data collected by a reachable sink from u is $p^{h(v'')} \cdot L(u) \leq p^{l+1} \cdot L(u) \leq p^l \cdot L(u)$ because $p \leq 1$. This implies that the network throughout will be reduced if performing the edge swapping between the tree edge (u, v) and the non-tree edge (u, v''). The lemma then follows. □

To maintain the maximum network throughput, following Lemma 4.6, we only make use of the non-tree edges between neighboring layers to adjust the load among the sinks.

Given the load-balanced trees in forest $\mathcal{F}$, the dynamic load readjustment algorithm examines the tree edges in the forest from the lower layer to the higher layer. Let $E_{l,l+1} = (V_l \times V_{l+1}) \cap E$ be the edge set between two neighboring layers l and $l+1$ for all l with $1 \leq l \leq h-1$. Let $e_1 = (u, v) \in E_{l,l+1}$ be a tree edge considered at this moment and nodes v and u are in layers l and $l+1$, respectively. Let $L(u)$ be the volume of data collected at node u in T_i from all its descendant nodes in the subtree rooted at u, and the volume of data received at the sink derived from node u is $p^l \cdot L(u)$. We will remove this tree edge and add another non-tree edge $e_2 = (u, v') \in E_{l,l+1}$ to form a new forest if this leads to a lower service cost. Assume v and v' are in trees T_i and T_j rooted at sinks s_i and s_j. Perform the swapping only if all the following conditions are met: (i) T_i and T_j are not the same tree, (ii) $L(s_i) > Q$ while $L(s_j) < Q$, and (iii) $L(s_i) - p^l \cdot L(u) > L(s_j)$. This procedure continues

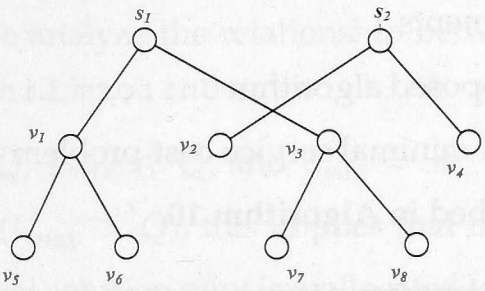
until all tree edges have been examined. It can be shown that after a series of swapping, the network throughput has not been changed but the service cost will be reduced. We refer to this dynamic load readjustment procedure as algorithm `Refine_Cost` and detail its description in **Algorithm 9**.

Algorithm 9: `Refine_Cost`

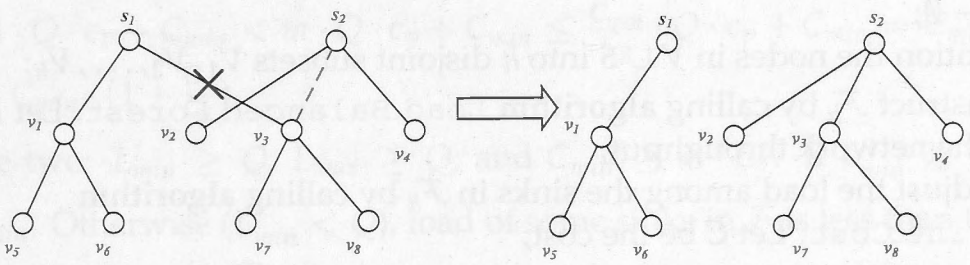
Input : $G(V \cup S, E)$, The routing forest $\mathcal{F}$, data quota Q
Output: The refined cost C, the updated routing forest $\mathcal{F}$

Calculate the cost of the forest $\mathcal{F}$:
$C = m \cdot C_f + \sum_{i=1}^{m} \max\{0, (L(s_i) - Q) \cdot c_p\}$;
if $L(s_i) \leq Q$ *or* $L(s_i) \geq Q$ *for all* i *with* $1 \leq i \leq m$ **then**
  /* the load of each sink is either greater or smaller than the quota Q */ **EXIT**;
  else
    for $l \leftarrow h - 1$ *downto* 1 **do**
      $E_{l,l+1} \leftarrow (V_l \times V_{l+1}) \cap E$;
      while *all tree edges have not been examined* **do**
        For any two edges $e_1 = (u, v)$ (a tree edge) and $e_2 = (u, v')$ (an non-tree edge) in $E_{l,l+1}$, where nodes v and v' in layer l and $v \neq v'$, and node u in layer $l + 1$, perform swapping if e_1 and e_2 are in two different trees $T(e_1)$ and $T(e_2)$ in $\mathcal{F}$, $L(T(e_1)) > Q$ and $L(T(e_2)) < Q$, and $L(T(e_1)) - p^l \cdot L(u) > L(T(e_2))$;
        Update the related trees in the forest and their load;
    Recalculate the cost C of the resulting forest $\mathcal{F}$ according to Eq. (4.4);
return $\mathcal{F}$ *and* C.

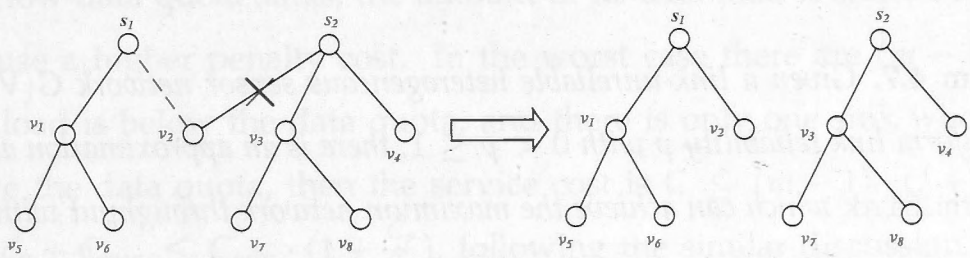
An example. Fig. 4.3 uses an example to illustrate the dynamic load readjustment to the trees in the forest shown in Fig. 4.2. Assume that the uniform link reliability p is 0.7 and $Q = 2 \cdot \tau \cdot r_g$. The load of the two sinks s_1 and s_2 is $L(s_1) = 3.36 \cdot \tau \cdot r_g > Q$ and $L(s_2) = 1.4 \cdot \tau \cdot r_g < Q$, respectively. And the service cost before the readjustment is $2 \cdot C_f + (3.36 - 2) \cdot \tau \cdot r_g \cdot c_p = 2 \cdot C_f + 1.36 \cdot \tau \cdot r_g \cdot c_p$. Now, consider node v_3 with $L(v_3) = 2.4 \cdot \tau \cdot r_g$ in layer 2. As $L(s_1) - 0.7 \cdot L(v_3) > L(s_2)$, we remove v_3 from T_1 and add it into T_2 by connecting v_3 and s_2. Consequently, the load of the two sinks is updated to



(a) The load-balanced routing trees rooted at s_1 and s_2 with $L(s_1) = 3.36 \cdot \tau \cdot r_g$, $L(s_2) = 1.42 \cdot \tau \cdot r_g$ and $C = 2 \cdot C_f + 1.36 \cdot \tau \cdot r_g \cdot c_p$.



(b) After a pair of edge swapping: $L(s_1) = 1.68 \cdot \tau \cdot r_g$, $L(s_2) = 3.08 \cdot \tau \cdot r_g$, and $C = 2 \cdot C_f + 1.08 \cdot \tau \cdot r_g \cdot c_p$.



(c) After the second pair of edge swapping: $L(s_1) = 2.38 \cdot \tau \cdot r_g$, $L(s_2) = 2.38 \cdot \tau \cdot r_g$, and $C = 2 \cdot C_f + 0.76 \cdot \tau \cdot r_g \cdot c_p$.

Figure 4.3: An example of refinement for further cost reduction

$L(s_1) = 1.68 \cdot \tau \cdot r_g < Q$ and $L(s_2) = 3.08 \cdot \tau \cdot r_g > Q$, and the service cost is reduced to $2 \cdot C_f + (3.08 - 2) \cdot \tau \cdot r_g \cdot c_p = 2 \cdot C_f + 1.08 \cdot \tau \cdot r_g \cdot c_p$. We then consider node v_2 in layer 2 with $L(v_2) = \tau \cdot r_g$. Since $L(s_2) - 0.7 \cdot L(v_2) > L(s_1)$, we remove v_2 from T_2 and add it to T_1 by removing edge (v_2, s_2) and connecting v_2 and s_1. The load of the two sinks is updated to $L(s_1) = 2.38 \cdot \tau \cdot r_g > Q$ and $L(s_2) = 2.38 \cdot \tau \cdot r_g > Q$. The routing trees in the updated forest now are perfectly balanced, and the service cost is $2 \cdot C_f + 2 \cdot 0.38 \cdot \tau \cdot r_g \cdot c_p = 2 \cdot C_f + 0.76 \cdot \tau \cdot r_g \cdot c_p$, which is reduced by $0.6 \cdot \tau \cdot r_g \cdot c_p$ after a series of

dynamic load readjustments.

Algorithm. The proposed algorithm `Uniform_Link` for the network throughput maximization with minimal service cost problem in a WSN with uniform link reliability is described in **Algorithm 10**.

Algorithm 10: `Uniform_Link`

Input : $G(V \cup S, E)$, monitoring period of τ
Output: The network throughput $D^{(\tau)}$ and the cost C

$\mathcal{F} \leftarrow \emptyset$;
Partition the nodes in $V \cup S$ into h disjoint subsets $V_1, V_2, \ldots, V_h$;
Construct $\mathcal{F}_h$ by calling **algorithm** `Load_Balanced_Forest`. Let $D^{(\tau)}$ be the network throughput;
Readjust the load among the sinks in $\mathcal{F}_h$ by calling **algorithm** `Refine_Cost`. Let C be the cost;
return $D^{(\tau)}$ *and* C.

Theorem 4.7. *Given a link-unreliable heterogeneous sensor network $G(V \cup S, E)$ with uniform link reliability p with $0 < p \leq 1$, there is an approximation algorithm* `Uniform_Link` *which can achieve the maximum network throughput with at most $(1 + \frac{C_p}{C_f})$ times of the minimal service cost, where $C_p = Q \cdot c_p$. The time complexity of the proposed algorithm is $O(|V||E|^2)$, assuming that $|S| << |V|$.*

Proof. Following our analysis, if each node sends its data along a shortest path (in terms of the number of hops) to a sink, it makes the maximum contribution to the network throughput. In algorithm `Uniform_Link`, each node routes its data to a sink along a shortest path, thus, the accumulative volume of data received by all sinks is the maximum one.

We then show that the service cost of the solution is no more than $(1 + \frac{C_p}{C_f})$ times of the minimal one. Assuming that in the optimal solution, the minimum and maximum loads among sinks are L_{min} and L_{max}, respectively, with the minimal service cost C_{min}. Let L'_{min} and L'_{max} be the minimum and maximum loads among the sinks in the forest $\mathcal{F}$ delivered by algorithm `Uniform_Link`,

with service cost C. We analyze the relationship between C and C_{min} by the following three cases.

Case one: $L_{min} \leq Q$, $L_{max} \leq Q$, and $C_{min} = m \cdot C_f$. If $L'_{max} \leq Q$, then $C = C_{min}$. Otherwise ($L'_{max} > Q$), this implies that the below-quota load of some sinks in the optimal solution now is reallocated to the other above-quota sinks in $\mathcal{F}$. Since the load of each sink is less than Q in the optimal solution, the maximum amount of shifted load is no more than $(m-1) \cdot Q$. Thus, $C \leq (m-1) \cdot Q \cdot c_p + C_{min} < m \cdot Q \cdot c_p + C_{min} \leq \frac{C_{min}}{C_f} \cdot Q \cdot c_p + C_{min} = C_{min} \cdot (1 + \frac{Q \cdot c_p}{C_f}) = C_{min} \cdot (1 + \frac{C_p}{C_f})$.

Case two: $L_{min} \geq Q$, $L_{max} \geq Q$, and $C_{min} \geq m \cdot C_f$. If $L'_{min} \geq Q$, then $C = C_{min}$. Otherwise ($L'_{min} < Q$), load of some sinks in $\mathcal{F}$ is less than the data quota Q while the load of the rest of sinks is far more than data quota. For each below-data quota sinks, the amount of its data load is shifted to others that cause a higher penalty cost. In the worst case there are $(m-1)$ sinks whose load is below the data quota, and there is only one sink whose load is above the data quota, then the service cost is $C \leq (m-1) \cdot Q + C_{min} < m \cdot Q \cdot c_p + C_{min} \leq C_{min} \cdot (1 + \frac{C_p}{C_f})$, following the similar discussion in Case one, omitted.

Case three: $L_{min} \leq Q$, $L_{max} \geq Q$, and $C_{min} \geq m \cdot C_f$. If $L'_{min} < L_{min}$ and $L'_{max} > Q$, then $C < C_{min} \cdot (1 + \frac{C_p}{C_f})$, following the similar arguments as for cases one and two, omitted. Otherwise, it can be shown that $C < C_{min} \cdot (1 + \frac{C_p}{C_f})$ as well.

Thus, the service cost of the delivered solution is no more than $(1 + \frac{C_p}{C_f})$ times of the minimal cost.

The time complexity of algorithm `Uniform_Link` is analyzed as follows. Partitioning the nodes into layers by the BFS transversal to graph G takes $O(|V| + |E|)$ time. In the construction of the forest of load-balanced routing trees, the main routine is the layer expansion by invoking a maximum flow algorithm, taking $O(|E_{l,l+1}| \cdot (|V_{l+1} + K)) = O(|E_{l,l+1}| \cdot |V|)$ time. Since it

takes $O(\log |V_{l+1}|)$ time to find the minimum value of the maximum capacity B for the flow network, the time spent on layer expansion is $O(|E_{l,l+1}| \cdot |V| \log |V_{l+1}|)$, and finding the forest takes $\sum_{l=1}^{h-1} O(|E_{l,l+1}| \cdot |V| \log |V_{l+1}|) = O(|E| \cdot |V| \log |V|)$. The time complexity of dynamic load readjustment is analyzed as follows. Routine `Refine_Cost` proceeds layer by layer. The number of pairs of edge swapping between layers l and $l+1$ is no more than $|E_{l,l+1}|^2$, while each edge swapping takes $O(|V|)$ time by updating at most two routing trees in the forest. Thus, the time spent on dynamic load readjustment is $\sum_{l=1}^{h-1} O(|E_{l,l+1}|^2 \cdot |V|) = O(|V|(\sum_{l=1}^{h-1} |E_{l,l+1}|)^2) = O(|V| \cdot |E|^2)$. The time complexity of `Uniform_Link` therefore is $\sum_{l=1}^{h-1} O(|E_{l,l+1}| \cdot |V| \log |V_{l+1}|) + \sum_{l=1}^{h-1} O(|E_{l,l+1}|^2 \cdot |V|) = O(|V||E|^2)$. □

4.4.2 Algorithm for General Networks with Non-uniform Link Reliability

We now remove the assumption of uniform link reliability and consider the problem of maximizing network throughput with minimal service cost in the general network with non-uniform link reliability. We first propose a simple heuristic that maximizes the network throughput without optimizing the service cost. We then provide improved heuristics that trade off between the network throughput and the service cost effectively.

A simple heuristic

Recall that $T_1, T_2, \ldots, T_m$ are the m routing trees rooted at the m sinks. We now construct the m routing trees such that the network throughput is maximized. Initially, each tree T_i contains only sink $s_i \in S$ for each i with $1 \leq i \leq m$. Let $V' \subseteq V$ be the set of nodes that are not included in these trees but one hop neighbors of the nodes in the trees $\cup_{i=1}^{m} V(T_i)$, i.e., $V' = \{v \mid (u,v) \in E, u \in \cup_{i=1}^{m} V(T_i), v \notin \cup_{i=1}^{m} V(T_i)\}$. The proposed algorithm proceeds itera-

tively. Within each iteration, only one node in V' is added to one of the m trees. Consider a node $v \in V'$. If there are multiple edges between v and the nodes in T_i, we choose one endpoint $u \in V(T_i)$ such that the reliability of the unique path from v to s_i, $p_{v,s_i} = p_{v,u} \cdot p_{u,s_i}$, is maximized. Thus, the contribution of node v to the network throughput via sink s_i within a period of τ is $d^{(\tau)}_{v,s_i} = p_{v,s_i} \cdot (\tau \cdot r_g)$. If there are multiple trees in which v has edges connected to the nodes, the maximum contribution made by v to the network throughput is $d^v_{max} = \max\{d^{(\tau)}_{v,s_i} | 1 \leq i \leq m\}$. Having calculated the maximum contribution of each node $v \in V'$, we add a node $v' \in V'$ to a tree T_j in the current iteration if the maximum contribution made by v' via sink s_j is the maximum one among the nodes in V', i.e., $d^{v'}_{max} = \max\{d^v_{max} | v \in V'\}$. Node v' is then removed from V'. This procedure continues until $V' = \emptyset$. The service cost then can be calculated by Eq. (4.4). We refer to this iterative algorithm as `Simple_Alg`.

Theorem 4.8. *Given a link-unreliable heterogeneous sensor network $G(V \cup S, E)$ with link reliability p_e for link $e \in E$, algorithm* `Simple_Alg` *delivers a solution with the maximum network throughput but the service cost is not optimized.*

Proof. Let p^v_{max} be the end-to-end reliability of the most reliable routing path from v to one of its reachable sinks, i.e., $p^v_{max} = \max\{p_{v,s_i} \mid s_i$ is a reachable sink from v in $G\}$. The maximum network throughput contribution by node v is $d^v_{max} = p^v_{max} \cdot \tau \cdot r_g$, and the maximum network throughput contributed by all nodes during the period of τ is $D^{(\tau)}_{max} = \sum_{v \in V}(p^v_{max} \cdot \tau \cdot r_g)$.

In the following we show that the accumulative volume of data collected by all sinks in the solution delivered by algorithm `Simple_Alg` is exactly $D^{(\tau)}_{max}$ by contradiction. Let $v_1, v_2, \ldots, v_i$ be the node sequence added to the m routing trees, where node v_j is added to the forest prior to v_{j+1} for all j with $1 \leq j \leq i-1 < |V|$. Assume that v_i is added at iteration i. Let $N(v_i)$ be the set of one hop neighbors of v_i in the network G and $N_T(v_i)$ a subset of $N(v_i)$ in which

all nodes have been included in the forest. We claim that connecting v_i to one of its neighbors $u \in N_T(v_i)$ will result in the maximum network throughput contribution by v_i. That is, $p_{max}^{v_i} = \max\{p_{v_i,u} \cdot p_{max}^{u} | u \in N_T(v_i)\}$. Otherwise, assume that the most reliable routing path from v_i to a sink is first via its neighbor $u \notin \cup_{i=1}^{m} V(T_i)$, and then via a node $v' \in N_T(v_i)$ which is connected to u, i.e., $p_{max}^{v_i} = \max\{p_{v_i,u} \cdot p_{u,v'} \cdot p_{max}^{v'} \mid u \in N(v_i) \setminus N_T(v_i) = v' \in \cup_{j=1}^{m} V(T_j)\}$. In that case, the maximum network throughput contributions made by nodes u and v_i are $d_{max}^{u} = p_{u,v'} \cdot p_{max}^{v'} \cdot (\tau \cdot r_g)$ and $d_{max}^{v_i} = p_{v_i,u} \cdot p_{u,v'} \cdot p_{max}^{v'} \cdot (\tau \cdot r_g) = p_{v_i,u} \cdot d_{max}^{u} < d_{max}^{u}$, respectively. Following the forest construction, u should have already been added to the forest prior to v_i, which contradicts the assumption that $u \in N(v_i) \setminus N_T(v_i)$ is not yet in the forest when adding v_i to the forest. Thus, every node $v \in V$ is added to the forest via the most reliable path and has the maximum network throughput contribution $p_{max}^{v} \cdot \tau \cdot r_g$. The volume of data collected by all sinks is $D_{max}^{(\tau)}$.

Since the service cost is not taken into account in algorithm `Simple_Alg`, it has not been optimized. □

Approximation algorithms

Although the network throughput delivered by algorithm `Simple_Alg` is the maximum one, the service cost is not optimized. We now take into account service cost in the design of approximation algorithms to strive for a non-trivial trade-off between network throughput and service cost. For the convenience of discussion, we assume that the reliability of each link in $G(V \cup S, E)$ is within the range of $[p, (1+\delta)p]$, where $p > 0$ and $(1+\delta)p \leq 1$. We start with a basic algorithm and then make a further improvement based on it.

The basic heuristic is extended by algorithm `Uniform_Link` and proceeds as follows. It first calls algorithm `Load_Balanced_Forest` by assuming that the link reliability of each link is $0 < p \leq 1$. Let $\mathcal{F} = \{T_1, T_2, \ldots, T_m\}$ be the forest obtained, which is a feasible solution to the problem. It then per-

forms dynamic load readjustment on the trees in $\mathcal{F}$ to further improve the network throughput while optimizing the service cost too. Note that the dynamic load readjustment algorithm for uniform link reliability in the previous section cannot be applicable anymore since that edge swapping between neighboring layers may reduce the network throughput as each link has a different link reliability, we thus need to modify algorithm `Refine_Cost` for dynamic load readjustment before applying it to the general case. Similar to the discussions in the previous section, we only consider edge swapping in the same neighboring layers, otherwise it will make the problem much more complicated and intractable. Consider swapping a pair of edges: a tree edge $e_1 = (u, v)$ and a non-tree edge $e_2 = (u, v')$, where node u is in layer $l + 1$ and nodes v, v' are in layer l of trees T_i and T_j rooted at sink s_i, s_j respectively. There are five cases to be considered.

Case one: $T_i = T_j$ and $L(s_i) > Q$. Perform the swapping only if $\Delta d = (p_{u,v'} \cdot p_{v',s_i} - p_{u,v} \cdot p_{v,s_i}) \cdot L(u) > 0$. The swapping increases the network throughput and the corresponding cost throughput increment will be paid.

Case two: $T_i = T_j$ and $L(s_i) < Q$. Perform the swapping only if $\Delta d = (p_{u,v'} \cdot p_{v',s_i} - p_{u,v} \cdot p_{v,s_i}) \cdot L(u) > 0$. Although the net increase in the network throughput is Δd, the extra charge for the amount of data $\Delta d - (Q - L(s_i))$ is paid only if $\Delta d - (Q - L(s_i)) > 0$.

Case three: $T_i \neq T_j$, $L(s_i) > Q$ and $L(s_j) < Q$. Perform the swapping only if both of the following conditions are met: (i) $\Delta d = (p_{u,v'} \cdot p_{v',s_j} - p_{u,v} \cdot p_{v,s_i}) \cdot L(u) > 0$; and (ii) either $L'(s_i) = L(s_i) - p_{u,v} \cdot p_{v,s_i} \cdot L(u) \geq Q$ or $Q - L(s_j) > Q - L'(s_i)$. We now analyze condition (ii). Assume that we perform the swapping. As a result, the loads at sink s_i and s_j become $L'(s_i) = L(s_i) - p_{u,v} \cdot p_{v,s_i} \cdot L(u)$ and $L'(s_j) = L(s_j) + p_{u,v'} \cdot p_{v',s_j} \cdot L(u)$ respectively. Now, if $L'(s_i) \geq Q$, the extra increase on the network throughput is at the expense of less data transfer cost, because only the amount of data $p_{u,v'} \cdot p_{v',s_j} \cdot L(u) - (Q - L(s_j))$ does incur any extra charges if $p_{u,v'} \cdot p_{v',s_j} \cdot L(u) - (Q - L(s_j)) > 0$,

as $L(s_j) < Q$ by our assumption. Otherwise ($L'(s_i) < Q$), the amount of data quota at sink s_i, $Q - L'(s_i)$, will not be used (wasting the paid money), while the unused data quota at sink s_j is $Q - L(s_j)$. In this case, only if $Q - L(s_j) > Q - L'(s_i)$, the swapping can proceed and the result is economical.

Case four: $T_i \neq T_j$, $L(s_i) < Q$ and $L(s_j) < Q$. The discussion is similar to Case three. Perform the swapping only if the following conditions are met: (i) $\Delta d = (p_{u,v'} \cdot p_{v',s_j} - p_{u,v} \cdot p_{v,s_i}) \cdot L(u) > 0$; and (ii) either $L'(s_j) \leq Q$ or $Q - L(s_j) \geq p_{u,v} \cdot p_{v,s_i} \cdot L(u)$.

Case five: $T_i \neq T_j$, $L(s_i) > Q$ and $L(s_j) > Q$. The discussion is similar to Case three. Perform the swapping only if $\Delta d = (p_{u,v'} \cdot p_{v',s_j} - p_{u,v} \cdot p_{v,s_i}) \cdot L(u) > 0$ and $L'(s_i) = L(s_i) - p_{u,v} \cdot p_{v,s_i} \cdot L(u) \geq Q$ are met.

We refer to this basic algorithm as `Appro`. It can be shown that after a series of edge swapping, the network throughput is increased while the service cost is also further optimized. We have the following theorem.

Theorem 4.9. *Given a link-unreliable heterogeneous wireless sensor network $G(V \cup S, E)$ with link reliability in $[p, (1+\delta)p]$ and $0 \leq \delta < \frac{1}{p} - 1$, there is an algorithm* `Appro` *for the maximizing network throughput with minimal service cost problem, which delivers a solution with no less than $\frac{1}{(1+\delta)^h}$ times of the maximum network throughput while the service cost is no more than $(1 + \frac{C_p}{C_f})$ times of the minimal one, where h is the maximum number of hops from any node to its nearest sink. The algorithm takes $O(|V| \cdot |E|^2)$ time.*

Proof. Let P_{v,s_i} be the most reliable path consisting of l links in $G(V \cup S, E)$ from node $v \in V$ to one of the m sinks, e.g., sink s_i, then the maximum volume of data can be collected at s_i for a period of τ is $d^v_{max} = \Pi_{e \in P(v,s_i)} p_e \cdot (\tau \cdot r_g)$, with $l \geq h(v)$ where $h(v)$ is the height of node v in the BFS tree in G' (G' is defined in Section 4.4.1). Assume that there is another shortest path P_v from v to another sink, if each link in P_v is the most reliable one with reliability $(1+\delta)p$, then the volume of received data at the sink is $d^v_U = p^{h(v)}(1+\delta)^{h(v)} \cdot (\tau \cdot r_g)$, which is

an upper bound on the maximum network throughput contribution of v. It is easy to see that $d_U^v \geq d_{max}^v$ because $h(v) \leq l$. On the other hand, if all links in P_v are the least reliable, then the amount of data received at the sink is no less than $d_L^v = p^{h(v)} \cdot (\tau \cdot r_g)$. Let node v be in a tree rooted at sink s_i and $d_v^{s_i}$ the volume of data generated from v and received at s_i, following the proposed algorithm, we have $\frac{d_v^{s_i}}{d_{max}^v} \geq \frac{d_L^v}{d_{max}^v} \geq \frac{d_L^v}{d_U^v} = \frac{1}{(1+\delta)^{h(v)}}$. Let $\gamma = \frac{1}{1+\delta}$, then, $d_v^{s_i} \geq d_L^v = \gamma^{h(v)} \cdot d_U^v$. Let $d_U^l = \sum_{v \in V_l} d_U^v$ be the upper bound on the accumulative volume of data collected by all sinks from nodes in V_l. Then, if the shortest routing path from each node to sinks is used to route its sensed data to a sink, the amount of data collected at the sink is no less than $\gamma^{h(v)}$ times the maximum volume of data from node v to any sink in the network.

Let X_l and Y_l be the sums of data received by the sinks from the nodes in V_l, using the most reliable path and the least reliable shortest path, respectively, then $X_l = \sum_{v \in V_l} \prod_{e \in P(v,s_i)} p_e \cdot (\tau \cdot r_g)$ and $Y_l = \sum_{v \in V_l} p^l \cdot (\tau \cdot r_g)$. The approximation ratio of the network throughput *Approx* by the algorithm `Appro` to the optimal one *OPT*, ξ, is

$$\begin{aligned} \xi &= \frac{Approx}{OPT} \geq \frac{\sum_{l=2}^{h} Y_l}{\sum_{l=2}^{h} X_l} \geq \frac{\sum_{l=2}^{h} \sum_{v \in V_l} d_L^v}{\sum_{l=2}^{h} \sum_{v \in V_l} d_{max}^v} \geq \frac{\sum_{l=2}^{h} \sum_{v \in V_l} d_L^v}{\sum_{l=2}^{h} \sum_{v \in V_l} d_U^v} \\ &\geq \frac{\sum_{l=2}^{h} \gamma^l \cdot d_U^l}{\sum_{l=2}^{h} d_U^l} \geq \frac{\gamma^h \cdot \sum_{l=2}^{h} d_U^l}{\sum_{l=2}^{h} d_U^l} = \gamma^h = \frac{1}{(1+\delta)^h}. \end{aligned} \tag{4.7}$$

The upper bound on the service cost has been proven in Theorem 4.7, omitted. The analysis of time complexity is almost identical to the one in Theorem 4.7, omitted. □

Note that the analytical estimation on the approximation ratio is very conservative, which yet has been confirmed from the later empirical results that the actual network throughout is no less than 78% of the maximum throughput, while the service cost is no more than 116% of the minimal cost.

In algorithm `Appro`, its analytical ratio $\frac{1}{(1+\delta)^h}$ is determined by parameters

h and δ. If link reliability vary significantly, the value of δ becomes large, and the approximation ratio on the network throughput will be small. We now further improve the network throughput while minimizing the service cost by proposing an improved approximate algorithm.

Assume that each link reliability is in range $[p_{min}, p_{max}]$, with $0 < p_{min} \leq p_{max} \leq 1$. We classify the links in $G(V \cup S, E)$ into $\lceil \log \frac{p_{max}}{p_{min}} \rceil$ different groups according to their reliability. Note that a link can join multiple groups. Then, a subgraph $G_i = (V \cup S, E_i)$ of G can be induced from the network G for each group i, where $E_i = \{e \mid e \in E \text{ and } p_e \geq p_i\}$ and $p_i = \min\{2^i \cdot p_{min}, p_{max} | 0 \leq i \leq \lceil \log \frac{p_{max}}{p_{min}} \rceil\}$. The algorithm proceeds as follows. Starting from $i = \lceil \log \frac{p_{max}}{p_{min}} \rceil$, for each i, a graph $G_{s_v,i} = (V \cup S \cup \{s_v\}, E_i \cup \{(s_v, s_j) \mid s_j \in S\})$ is constructed by adding a virtual sink s_v and the edges between s_v and every sink in S. If $G_{s_v,i}$ is disconnected, there is no solution for the problem, which is shown by Lemma 4.10. The value of i is decreased by one and the procedure continues until $G_{s_v,i}$ is connected. It then applies algorithm `Appro` on graph $G_{s_v,i}$ to find a solution. We refer to this improved algorithm as `Impro_Appro`, described in **Algorithm 11.**

Lemma 4.10. *In algorithm* `Impro_Appro`, *if* $G_{s_v,i}$ *is disconnected, there is no solution for graph* $G_{s_v,j}$, *with* $0 < i \leq j \leq \lceil \log \frac{p_{max}}{p_{min}} \rceil$. *Algorithm* `Impro_Appro` *can deliver an approximate solution to the concerned problem with the complexity of* $O(|V| \cdot |E|^2 + |E| \cdot \lceil \log \frac{p_{max}}{p_{min}} \rceil)$.

Proof. As the set of links in $G_{s_v,j}$ is a subset of the set of links in $G_{s_v,i}$ when $j > i$, graph $G_{s_v,j}$ is a subgraph of $G_{s_v,i}$. If $G_{s_v,i}$ is not connected, graph $G_{s_v,j}$ is not connected either, with $0 < i \leq j \leq \lceil \log \frac{p_{max}}{p_{min}} \rceil$. For any disconnected graph $G_{s_v,j}$, performing a BFS traversal on it will not obtain a tree since at least one node $v \in V$ is not in the connected component that virtual sink s_v and the sinks belong to, which implies that there is not any routing path from node v to any sink. Thus, there is not a forest in $G_{s_v,j}$ spanning all nodes in

Algorithm 11: `Impro_Appro`

Input : $G(V \cup S, E)$, the data quota Q, the link reliability range $[p_{min}, p_{max}]$

Output: The network throughput $D^{(\tau)}$, the forest $\mathcal{F}$, and the service cost C

$D^{(\tau)} \leftarrow 0$; $C \leftarrow m \cdot C_f$; $\mathcal{F} \leftarrow \emptyset$;

$i \leftarrow \lceil \log \frac{p_{max}}{p_{min}} \rceil$;

while $i \geq 0$ **do**

  $p_i = \min\{2^i \cdot p_{min}, p_{max}\}$;

  Construct a sub-network $G_i = (V \cup S, E_i)$, where $E_i = \{e \mid e \in E$ and $p_e \geq p_i\}$;

  Construct a graph $G_{s_v,i} = (V \cup S \cup \{s_v\}, E_i \cup \{(s_v, s_j) \mid s_j \in S\})$ by adding a virtual sink s_v and connecting s_v to all sinks in S;

  if *graph* $G_{s_v,i}$ *is disconnected* **then**

    $i \leftarrow i - 1$;

  else

    Perform a BFS traversal on $G_{s_v,i}$ starting from s_v to obtain a tree T_i^{BFS} with depth h;

    Partition the nodes in $G_{s_v,i}$ into h layers;

    Call algorithm `Appro` which consists of applying **Algorithm 8** to the layered nodes with uniform link reliability p_i, followed by calling the dynamic load readjustment procedure;

    Obtain the network throughput $D^{(\tau)}$, the service cost C, and the routing forest $\mathcal{F}$;

    EXIT;

return $D^{(\tau)}$, $\mathcal{F}$, C.

V and no solution can be obtained. There is an i with $0 \leq i \leq \lceil \log \frac{p_{max}}{p_{min}} \rceil$ such that $G_{s_v,i}$ is connected, as G is connected. And an approximate solution is obtained whose throughput is with the approximation ratio $\frac{1}{(1+\delta_i)^h}$, where $\delta_i = \frac{p_{max}}{2^i \cdot p_{min}} - 1 \leq \delta = \frac{p_{max}}{p_{min}} - 1$.

It takes $O(|E| + |V|)$ time to check the connectivity of a graph, and the number of connectivity check is no greater than $\lceil \log \frac{p_{max}}{p_{min}} \rceil$. It takes $O(|V| \cdot |E|^2)$ to deliver a solution by calling algorithm `Appro`, referring to Theorem 4.9. Thus, algorithm `Impro_Appro` takes $O(|V| \cdot |E|^2 + |E| \cdot \lceil \log \frac{p_{max}}{p_{min}} \rceil)$ time.

It can be seen that although the running time of algorithm `Impro_Appro` is

longer than that of algorithm `Appro`, the throughput delivered by it is much higher in comparison with that by algorithm `Appro` in most cases. □

4.5 Performance Evaluation

In this section we evaluate the performance of proposed algorithms, including investigating the impact of constraint parameters on their performance, and comparing their performance with those of other algorithms.

4.5.1 Experiment Settings

We consider a sensor network consisting of 100 to 300 sensors randomly deployed in a $1000m \times 1000m$ square region. The transmission range of sensors is 120 meters, and the data generation rate is $r = 100Bytes/s$. The initial energy capacity of each sensor IE is $1,000$ *Joules*. The energy consumption parameters of IEEE 802.15.4 and 3G radios are referred to [2] and [5]. In our experiments, the following three different data plans provided by Vodafone [7] for one month monitoring period (i.e., $\tau = 30days \times 24hours \times 3,600seconds$) will be examined: (I) $Q = 2GB$ and $C_f = \$19$; (II) $Q = 4GB$ and $C_f = \$29$; and (III) $Q = 10GB$ and $C_f = \$39$. Each of these three data plans has the same penalty rate $c_p = \$0.02/MB$. and the link reliability is a random value within the interval $[0.1, 1.0]$. Each value in the figures is the mean of the results by applying the mentioned algorithm to 50 different network topologies of the same size.

4.5.2 Evaluation of Service Cost Minimization Algorithm

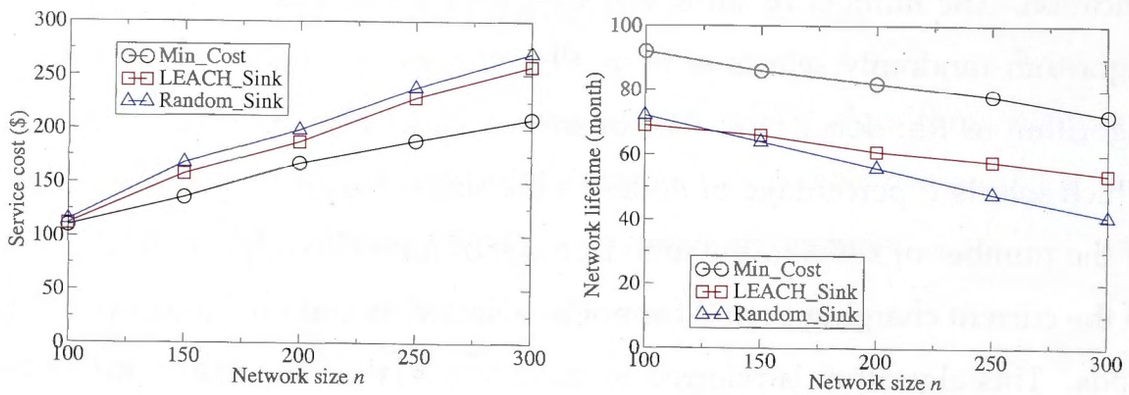
We first evaluate the performance of algorithm `Min_Cost` for service cost minimization with guaranteed network throughput. In the default setting, the network throughput threshold $\alpha = 0.7$, the search space percentage $\beta = 0.1$, the

weight adjustment parameter $\lambda = 2$, and data Plan (II) is adopted.

We study the performance of algorithm `Min_Cost` against that of the other two algorithms. The only difference between these algorithms is in sink identification. The number of sinks m is delivered by algorithm `Min_Cost`. One algorithm randomly selects m from all nodes as the sinks. We refer to this algorithm as `Random_Sink`. The other is a variant of algorithm `LEACH` [40] which selects P percentage of nodes as the sinks, where $P = m/n$ is the ratio of the number of sinks to the total number of nodes. Nodes serving as sinks in the current charging period cannot be selected as sinks for the next $1/P$ periods. This algorithm is referred to as `LEACH_Sink`. The rest of these three algorithms is identical, that is, the routing forest is built by adopting the forest establishment in algorithm `Iden_Sink`. We compare the performance of these three algorithms in terms of the service cost and network lifetime by varying n from 100 to 300 while fixing $\alpha = 0.7$ and r_g =100 $Bytes/s$.

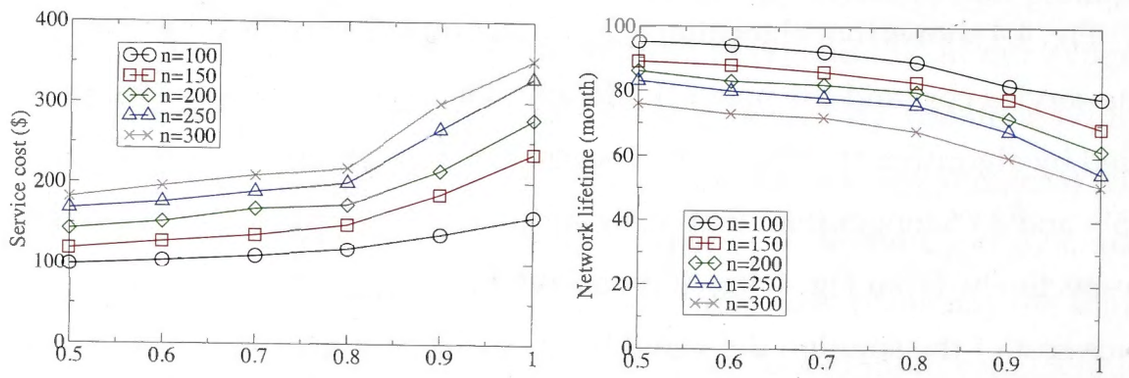
Fig. 4.4 shows that algorithm `Min_Cost` outperforms the other two in both the service cost and the network lifetime. On average, the service cost delivered by algorithm `Min_Cost` is 22% and 16% less, and the network lifetime is 45% and 33% longer than that of algorithms `Random_Sink` and `LEACH_Sink`, respectively. From Fig. 4.4(a), it is observed that with the increase in n, the service cost of the solution delivered by each algorithm goes up, because larger volume of data is required to be sent to the remote monitoring center and a higher cost is incurred. With the growth of n, the gap between the three service cost curves is further enlarged. Fig. 4.4(b) illustrates that the curves of network lifetime drop with the growth of n. The superiority of algorithm `Min_Cost` lies in a more efficient sink identification strategy to better balance the energy consumption among the sensor nodes. We also note that in terms of network lifetime, algorithm `LEACH_Sink` outperforms algorithm `Random_Sink` in most cases. The reason behind is that the nodes in algorithm `LEACH_Sink` cannot be repeatedly selected as the sinks in a number of consecutive rounds, while

algorithm `Random_Sink` does not pose such a restriction, which potentially increases the chances for more balanced energy distribution and a longer network lifetime.



(a) Service cost (b) Network lifetime

Figure 4.4: Performance comparison of three algorithms when $\alpha = 0.7$, r_g =100 *Bytes/s*, $\lambda = 2$, and Plan (II) is adopted



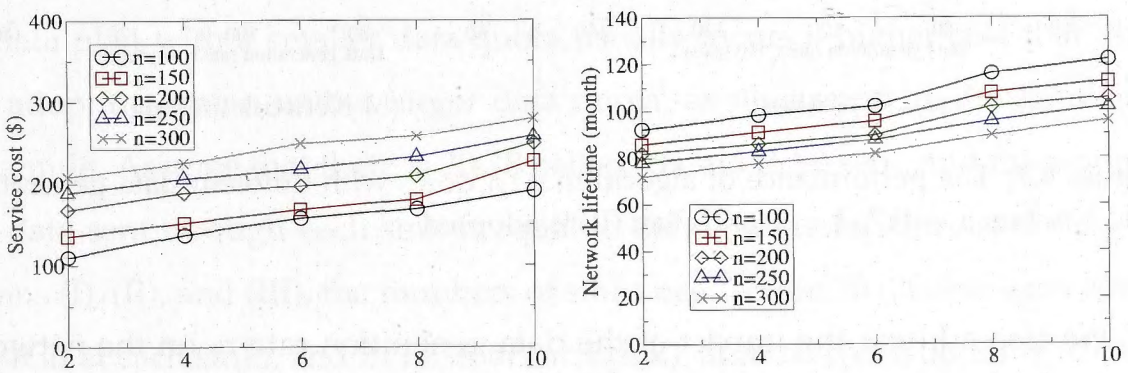
(a) Service cost (b) Network lifetime

Figure 4.5: The performance of algorithm `Min_Cost` with different throughput thresholds α when r_g =100 *Bytes/s*, $\lambda = 2$, and Plan (II) is adopted

We now investigate the impact of different constraint parameters on the performance of algorithm `Min_Cost` in terms of the service cost and network lifetime.

We start with the impact of the network throughput threshold α on the network performance by varying α from 0.5 to 1.0. As shown in Fig. 4.5, the

service cost increases while the network lifetime decreases as the value of α goes up. This is because the higher the throughput requirement, the larger the volume of sensed data collected from the sensor network, thereby resulting in a higher service cost and more energy consumption among sensors, thus a shorter network lifetime. However, note that when prior to $\alpha = 0.8$, the service cost and network lifetime vary slowly, while with α increasing from 0.8, both of them change dramatically. The rationale behind is that a larger α does not necessarily mean that the amount of data relayed by each sink increases accordingly. With the growth of α from 0.5 to 0.8, the forest of routing trees may not experience many changes, resulting in slight changes in the service cost and network lifetime. However, with further increase in α, a large number of sinks is expected to be used in order to meet the network throughput requirement, resulting in a greater service cost. The similar explanation applies to the trend of network lifetime.

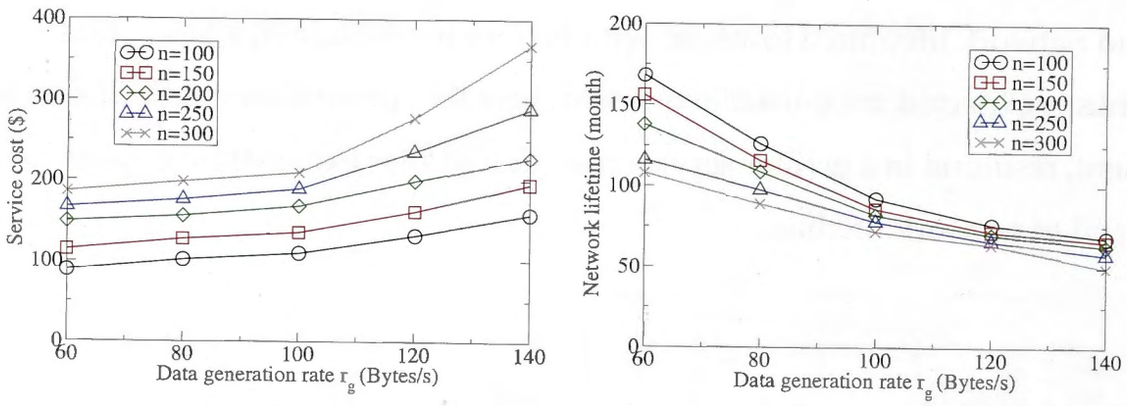


(a) Service cost (b) Network lifetime

Figure 4.6: The performance of algorithm `Min_Cost` with different weight adjustment parameter λ when r_g =100 *Bytes/s*, $\alpha = 0.7$, and Plan (II) is adopted

We then investigate the impact of weight adjustment parameter λ on the network performance by varying λ from 2 to 10. Fig. 4.6 indicates that the increase of λ results in a higher service cost yet a longer network lifetime. Recall that the weight of a directed edge $\langle v, u \rangle$ is $\omega(v, u) = IE \cdot \lambda^{1-er(v)/IE}/p(v, u)$.

The value of λ affects weights of edges thus the routing forest construction. The larger the value of λ, the greater the impact of the residual energy on the edge weight, and the more balanced energy consumption. However, a larger λ leads to a higher service cost, as shown in Fig. 4.6(a). In each iteration of searching the optimal number of sinks m, the routing forest algorithm with a larger λ delivers a forest with a lower throughput, compared with that delivered by the same algorithm with a smaller λ, which is very likely not to meet the specified throughput requirement. As a result, a larger number of sinks is required and a higher service cost is incurred.



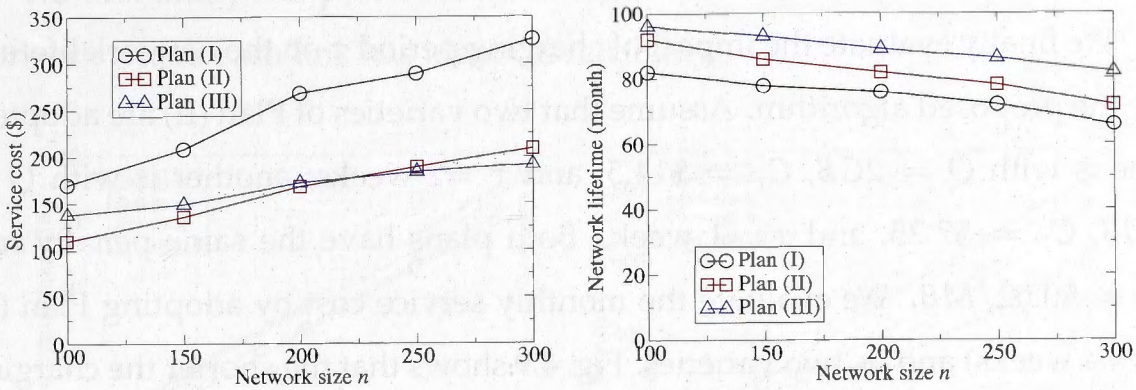
(a) Service cost

(b) Network lifetime

Figure 4.7: The performance of algorithm `Min_Cost` with different data generation rate r_g when $\alpha = 0.7$, $\lambda = 2$, and Plan (II) is adopted

We also address the impact of the data generation rate r_g on the network performance by varying r_g from 60 $Bytes/s$ to 140 $Bytes/s$ with the increment of 20 $Bytes/s$. Fig. 4.7 shows that with a fixed r_g, the larger the network size n, the higher the service cost, and the shorter the network lifetime, because more data is required to be transmitted and a larger number of sinks is required. This can also explain that with the increase in the data generation rate r_g, the service cost will go up while the network lifetime will drop.

What follows is to investigate the impact of different data plans on the network performance. Fig. 4.8(a) indicates that adopting Plan (I) incurs the



(a) Service cost (b) Network lifetime

Figure 4.8: The performance of algorithm `Min_Cost` with different data plans when $\alpha = 0.7$, $\lambda = 2$, and r_g =100 $Bytes/s$

highest service cost among the three plans. Adopting Plan (II) is the cheapest when $n \leq 200$, and when $n > 200$ Plan (III) leads to the smallest service cost. Recall that the number of sinks m is searched around $m_0 = \lfloor \frac{\alpha \cdot n \cdot \tau \cdot r_g}{Q} \rfloor$. A smaller data quota Q indicates a larger number of sinks needed and a higher service cost incurred, it is because transmitting the same amount of data by adopting a data plan with a smaller data quota usually incurs a higher cost than that of adopting a plan with a larger data quota, as illustrated by the following example. Assume that there is 20GB collected data to be sent, and the amount of data sent through each sink is equal to the data quota. Corresponding to plans (I), (II), and (III), the numbers of sinks needed are 10 (2GB at each sink), 5 (4GB at each sink), and 2 (10GB at each sink), and the corresponding service costs are \$190, \$145, \$79 respectively. Though the penalty is not considered, the fixed cost is dominant in the service cost. This explains the higher cost caused by Plan (I) in comparison with the other two plans. It is also interesting to see that when $n > 200$, adopting Plan (III) results in a lower cost compared with Plan (II). It is because when Plan (II) is adopted, a higher penalty is incurred, with the amount depending on the quota usage on individual sinks. In other words, adopting a plan with a larger quota (e.g., Plan (III)) means a smaller fixed cost yet might be accompanied with an expensive penalty, and results in

a high service cost in the end.

We finally evaluate the impact of charging period τ on the network lifetime by the proposed algorithm. Assume that two varieties of Plan (II) are adopted: one is with $Q = 2GB$, $C_f = \$14.5$, and τ =2 weeks; another is with $Q = 1GB$, $C_f = \$7.25$, and τ =1 week. Both plans have the same penalty rate $c_p = \$0.02/MB$. We evaluate the monthly service cost by adopting Plan (II) (τ =4 weeks) and its two varieties. Fig. 4.9 shows that the shorter the charging period, the longer the network lifetime delivered by the proposed algorithm, as this results in more frequent changes of sinks, thus more balanced energy consumption among the sensors.

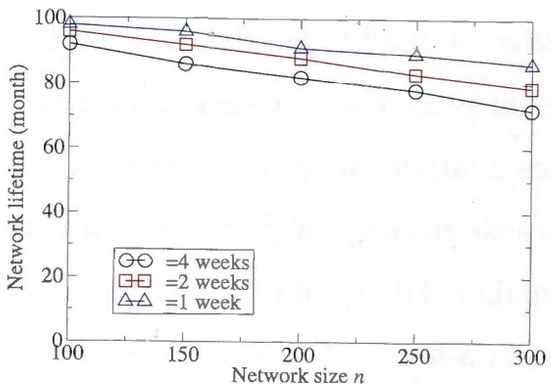
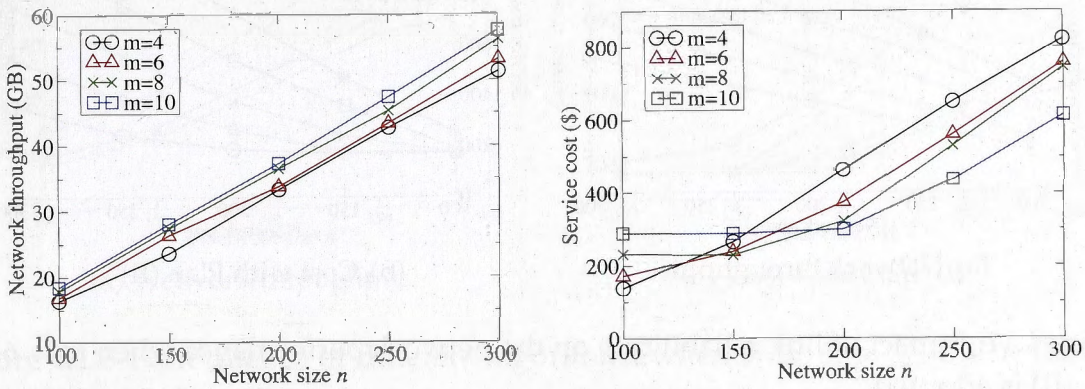


Figure 4.9: The performance of algorithm `Min_Cost` with different charging periods τ when r_g =100 $Bytes/s$, $\alpha = 0.7$, and $\lambda = 2$

4.5.3 Evaluation of Network Throughput Maximization Algorithm

We now evaluate the performance of the proposed algorithms for network throughput maximization with minimal service cost. In the default setting, the number of sinks m varies from 4 to 10, and they are deployed as follows. The monitoring region is divided into roughly equal-size m sub-regions, in each of which one sink is randomly deployed.

We first study the performance of algorithm `Uniform_Link` in a sensor network where each link has identical reliability.

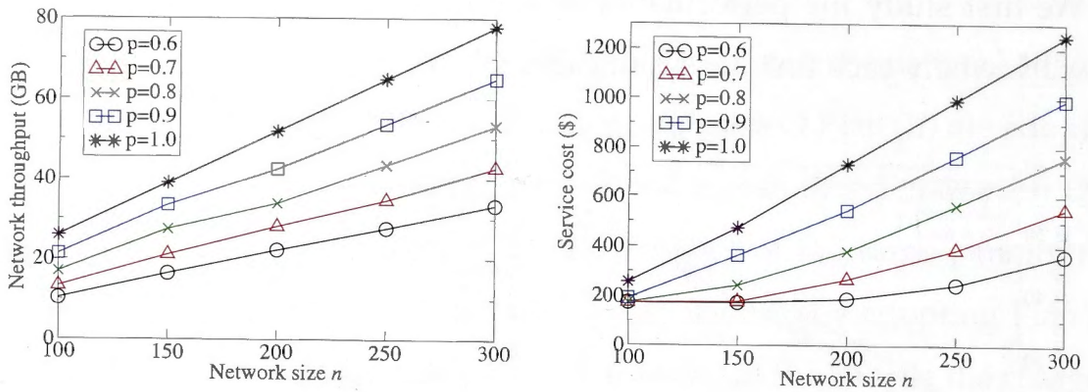


(a) Network throughput (b) Cost with Plan (II)

Figure 4.10: Impact of m on the network performance when $p = 0.8$ and Plan (II) is adopted.

We evaluate the impact of m on the network performance for different network sizes n, by varying m from 4 to 10 while keeping p fixed at 0.8. From Fig. 4.10(a) it can be seen that for a given network size, a larger m results in a higher network throughput. Fig. 4.10(b) shows that with the fixed m, a larger network throughput does not necessarily incur a higher service cost, depending on whether there is any sink exceeding the data quota. Also, we observe that with the fixed n, the service cost is insensitive to the value of m. For example, when $n = 100$, the service cost goes up when m increases as the network throughput is relatively low and penalties to individual sinks are unlikely to occur. The service cost essentially is the fixed cost of the m sinks. On the other hand, when n is large (e.g., $n > 200$), the network throughput becomes much higher, and the service cost is reduced with the growth of m as the penalties are reduced.

What followed is to evaluate the impact of link reliability on the network performance for different network sizes by varying p from 0.6 to 1.0 while keeping $m = 6$ and adopting Plan (II). Fig. 4.11 shows that the higher the link reliability, the larger the network throughput, and the higher the service cost.



(a) Network throughput (b) Cost with Plan (II)

Figure 4.11: Impact of link reliability p on the network performance when $m = 6$ and Plan (II) is adopted.

We also investigate the impact of different data plans on the network performance while fixing $p = 0.8$ and $m = 6$. Fig. 4.12 shows the corresponding service costs for transferring the collected data when different data plans are adopted. It is observed that when $n = 100$, Plan (III) has a higher service cost than the other two plans due to its higher fixed cost. However, with the growth of n, Plan (III) outperforms the other two plans since the data quota $Q = 10GB$ is sufficiently large and no sink exceeds the data quota, resulting in no penalties.

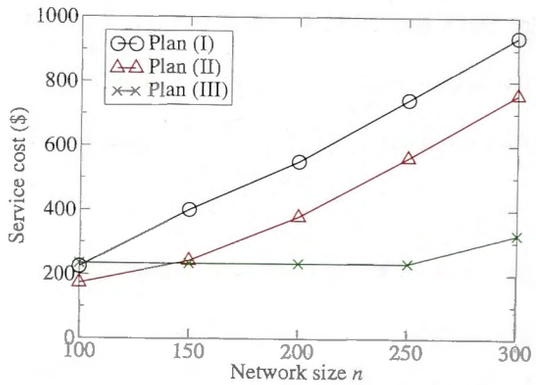
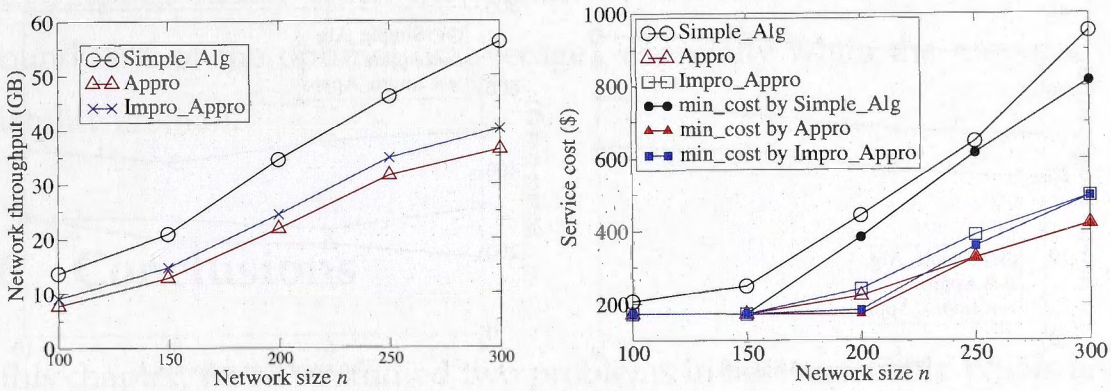


Figure 4.12: Impact of different data plans on the service cost when $p = 0.8$ and $m = 6$.

We now evaluate the performance of algorithms `Simple_Alg`, `Appro`, and

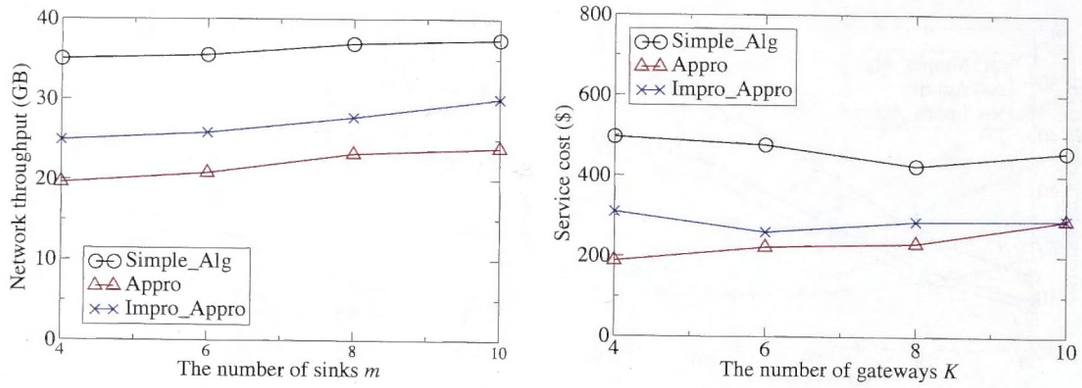


(a) Network throughput (b) Cost with Plan (II)

Figure 4.13: Performance of different algorithms when $m = 6$, $p_e \in [0.1, 1]$ for each link $e \in E$, and Plan (II) is adopted.

`Impro_Appro` when $m = 6$, $p_e \in [0.1, 1]$ for each link $e \in E$, and adopting Plan (II). Shown in Fig. 4.13(a), the network throughput by algorithm `Appro` and `Impro_Appro` are no less than 78% and 87% of the maximum one delivered by algorithm `Simple_Alg`. Algorithm `Impro_Appro` improves the throughput delivered by algorithm `Appro` by 11% with only 3% higher service cost incurred. Given an amount of network throughput $D^{(\tau)}$, the lower bound of the minimal cost is calculated as $m \cdot C_f + (D^{(\tau)} - m \cdot Q) \cdot c_p$, that is, data is allocated to the m sinks such that the number of sinks whose load is above the data quotas is minimized while the other sinks are assigned the load equal to the data quota. Fig. 4.13(b) shows that the service cost delivered by algorithms `Appro` and `Impro_Appro` are no more than 106% and 103% of the minimal cost. It also verifies that algorithm `Impro_Appro` best approximates the minimum service cost.

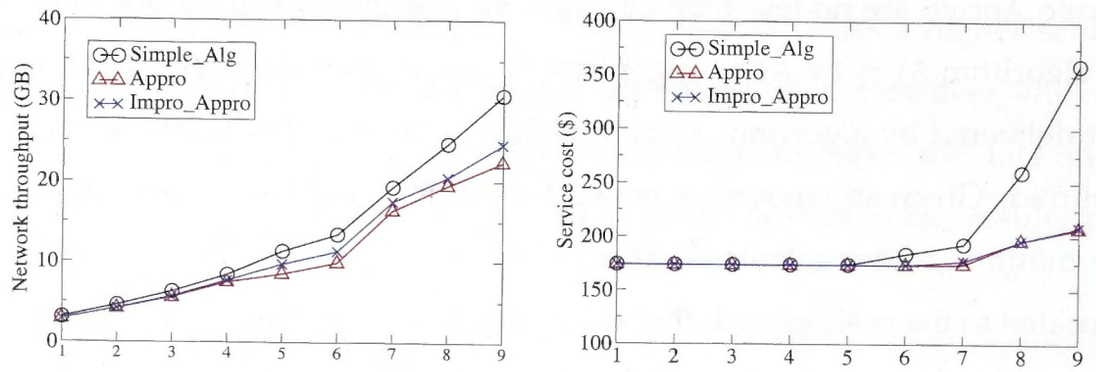
We next investigate the impact of the number of sinks m on the network performance when $n = 200$, $p_e \in [0.1, 1]$ for each link $e \in E$, and Plan (II) is adopted. Fig. 4.14 indicates that a larger m will result in a higher network throughput but does not necessarily incur a higher service cost. As a larger number of sinks can reduce the average length of the routing path from a node



(a) Network throughput (b) Cost with Plan (II)

Figure 4.14: Impact of m on the network performance when $n = 200$, $p_e \in [0.1, 1]$ for each link $e \in E$, and Plan (II) is adopted.

to a sink, this implies more reliable end-to-end data routing and a higher network throughput then follows.



(a) Network throughput (b) Cost with Plan (II)

Figure 4.15: Impact of the range of link reliability on the network performance when $m = 6$, $n = 200$, and Plan (II) is adopted.

We finally study the impact of the range of link reliability on the network performance. We examine the ranges of link reliability: $[0.1, 0.2]$, $[0.1, 0.4]$, $[0.1, 0.6]$, $[0.1, 0.8]$, and $[0.1, 1.0]$ by fixing $m = 6$, $n = 200$, and adopting Plan (II). Accordingly, the values of δ are 1, 3, 5, 7, 9. Fig. 4.15 shows that with the growth of the link reliability range, the network throughput increases, so does the service cost. The network throughput delivered by `Appro` and

`Impro_Appro` nearly reach the maximum one delivered by `Simple_Alg` (around 96% of the optimal on average), especially when the range of link reliability is small.

4.6 Conclusions

In this chapter, we have studied two problems in link-unreliable WSNs under the remote monitoring scenario, both of which are proven to be NP-complete. We first devised algorithm `Min_Cost` for service cost minimization while maintaining the required network throughput. The probabilities of throughput failure and cost exceeding are shown to be bounded. We then designed approximation algorithms for network throughput maximization with minimal service cost. Algorithm `Simple_Alg` was developed for the maximum throughput without optimizing the cost. Approximation algorithms `Appro` and `Impro_Appro` were proposed, with delivered solutions fractional of the optimum. The experimental results showed the superiority of the proposed algorithms over others in terms of network throughput, service cost, and network lifetime.

Chapter 5

Conclusions and Future Work

This chapter summarizes the contributions we made in this thesis, followed by discussing potential research topics derived from this work.

5.1 Summary of Contributions

The use of wireless sensor networks for remote monitoring has been studied in this thesis. New concepts, models and optimization techniques were proposed for achieving long-lasting, large-scale monitoring, and reliable, economical remote data transfer. Approaches for multiple sink placement and assignment were developed to prolong network lifetime significantly. A strategy for scheduling a motion-constrained sink was devised to improve monitoring quality considerably. Approximation algorithms for balancing workload among sinks were designed to achieve a fine trade-off between optimization of network throughput and service cost. The main contributions of this thesis are summarized as follows.

- We proposed remote monitoring by deploying wireless sensor networks in remote regions to sense phenomena of interest and employing a third party network to transfer data from the WSNs to a monitoring center that is located distant from the monitored regions. We explored challenges of using WSNs for remote monitoring and addressed research aims of this thesis, including network lifetime prolongation, monitoring quality

maximization, and network throughput optimization with minimal service cost.

- We investigated the longevity of the deployed WSN by exploring multiple sinks. We formulated energy cost models of sensor nodes working in different modes and studied their energy consumption components. Based on the models, we devised algorithm `Dynamic_Alg` which dynamically assigns nodes as sinks and rotates the assignment within network lifetime to balance energy consumption in the WSN, while maintaining a required network throughput. We also developed algorithm `Find_Optimal_Sink` to find a non-trivial trade-off between the number of sinks and network lifetime. Algorithm `Find_Optimal_Sink` identifies the appropriate number of sinks and their preferable locations in the network, and routes data energy-efficiently to maximize network lifetime.

- We dealt with the data loss issue that occurs in a WSN with a mobile sink. We explored sensed data correlation and identified a subset of nodes whose sensed data can be successfully sent to the sink, and thereby monitoring quality is improved. We proposed algorithm `MQM` to find a trajectory in the given road-map subject to constraints of travelling length and sojourn locations of the sink, so that the sink moving along the trajectory is able to energy-efficiently collect the most representative data for the maximum monitoring quality.

- We addressed the optimization of network throughput the service cost in two application scenarios. We first studied the throughput guaranteed cost minimization problem in a budget-oriented scenario and proposed algorithm `Min_Cost` for the problem, which determines the proper number of sinks, identifies the set of sinks, and routes data strategically to the sinks. We also investigated the throughput maximization problem in a

throughput-targeting scenario and developed approximation algorithms with delivered solutions proven to be fractional of the optimum.

- We conducted extensive experiments by simulation to evaluate all proposed algorithms including investigating the impact of constraint parameters on their performance, and comparing their performance with that of comparable algorithms. Experimental results showed that the proposed algorithms outperform the existing ones significantly in aspects of network lifetime, monitoring quality, network throughput, and service cost.

5.2 Future Work

There are several potential research topics that can be explored based on the work in this thesis.

Firstly, the use of multiple mobile sinks can be investigated to further balance energy consumption among nodes for network lifetime prolongation. It is challenging to arrange the motion of multiple sinks jointly and more complicated problems follow. For example, determining the minimum number of motion-constrained sinks and finding their trajectories to eliminate data loss. Or, with a given number of sinks, finding multiple closed paths in a pre-defined road-map to maximize the monitoring quality subject to the motion constraints. These problems need to be addressed for a better monitoring quality in a mobile-sink WSN.

Secondly, the latency of delivering data from sinks to the monitoring center through the third party network can be considered as another metric to evaluate the performance of remote monitoring. Such latency is an essential performance metric for remote event detection applications, such as fire detecting or seismic sensing systems, in which real-time data delivery is required. This ad-

ditional metric makes the performance optimization more difficult. Moreover, the third party communication service will be charged in relation to the data transmission rate provided, and the service cost, with both network throughput and data delivery latency taken into account, should be re-formulated accordingly.

Thirdly, light-weighted distributed implementation of the proposed techniques is to be investigated to enable their applications in real networks. The proposed algorithms in this thesis are centralized and executed by a coordinator, such as the energy-unconstrained sink or the monitoring center, and then corresponding operations are designated to nodes in the network. Devising simple distributed strategies that can be conducted by individual nodes will reduce the control message exchanges thereby conserving energy and decreasing channel interference. The solutions delivered by centralized algorithm can be used as benchmarks to evaluate the distributed algorithms.

Finally, to further improve the remote monitoring performance, the cross layer design should be researched in the future. Network lifetime prolongation, monitoring quality maximization, network throughput and service cost optimization considered in this thesis are in the routing layer. Taking into account the techniques in other layers, e.g., MAC layer, will generate algorithms more effective and efficient in improving the application performance. Moreover, combining approaches of multiple layers to form a joint optimization framework will eliminate any improper assumption raised by designing algorithms from the aspect of a single layer. And this will enable the developed mechanisms much more practical in real world.

Bibliography

[1] 128m x 8 bit / 64m x 16 bit NAND flash memory. http://www.datasheetcatalog.org/datasheets/1150/264846_DS.pdf.

[2] CC2420 2.4 GHz IEEE 802.15.4/ZigBee-ready RF transceiver. www.ti.com/lit/ds/symlink/cc2420.pdf.

[3] National renewable energy laboratory. http://www.nrel.gov/.

[4] The network simulator - ns-2. http://www.isi.edu/nsnam/ns/.

[5] PCI express minicard and LGA modules high-speed multi-mode 3G. www.embeddedworks.net/ewdatasheets/option/EW-Gobi3000.pdf.

[6] Sensor wikipedia. http://en.wikipedia.org/wiki/Sensor.

[7] Vodafone. http://www.vodafone.com.au.

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

[9] A. Arun, R. Aditya, and B. Mani. Mobile element scheduling with dynamic deadlines. *IEEE Transactions on Mobile Computing*, 50:395–410, 2007.

[10] A. Ayadi. Energy-efficient and reliable transport protocols for wireless sensor networks: State-of-art. *Wireless Sensor Network*, 3:106–113, 2011.

[11] A. Bagadi, S. Sarode, and J. Bakal. A survey of reliable transport layer protocols for wireless sensor network. *International Journal of Computer Applications*, 33:44–50, 2011.

[12] A. Bakre and B. Badrinath. I-TCP: Indirect TCP for mobile hosts. In *Proceedings of ICDCS*. IEEE, 1995.

[13] C. Bas and S. Ergen. Spatio-temporal characteristics of link quality in wireless sensor networks. In *Proceedings of WCNC*. IEEE, 2012.

[14] S. Basagn, A. Carosi, E. Melachrinoudis, C. Petrioli, and Z. M. Wang. A new milp formulation and distributed protocols for wireless sensor networks lifetime maximization. In *Proceedings of ICC*. IEEE, 2006.

[15] S. Basagni. Controlled sink mobility for prolonging wireless sensor networks lifetime. *Journal of Wireless Networks*, 14:831–858, 2008.

[16] A. Bogdanov, E. Maneva, and S. Riesenfeld. Power-aware base station positioning for sensor networks. In *Proceedings of INFOCOM*. IEEE, 2004.

[17] K. Brown and S. Singh. M-TCP: TCP for mobile cellular networks. In *Proceedings of SIGCOMM*. ACM, 1997.

[18] A. Cerpa, J. Wong, M. Potkonjak, and D. Estrin. Temporal properties of low power wireless links: modeling and implications on multi-hop routing. In *Proceedings of MobiHoc*. ACM, 2005.

[19] J. Chang and L. Tassiulas. Energy conserving routing in wireless ad hoc networks. In *Proceedings of INFOCOM*. IEEE, 2000.

[20] S. Chen. Routing support for providing guaranteed end-to-end quality-of-service. *PhD Thesis*, 1999.

[21] S. Chen, M. Song, and S. Sahni. Two techniques for fast computation of constrained shortest paths. *IEEE/ACM Transactions on Networking*, 167:105–115, 2008.

[22] T. Cormen, C. Leiserson, R. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2009.

[23] D. Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of Mobicom*. ACM, 2003.

[24] H. Dai and R. Han. A node-centric load balancing algorithm for wireless sensor networks. In *Proceedings of Globecom*. IEEE, 2003.

[25] K. Dasgupta, K. Kalpakis, and P. Namjoshi. An efficient clustering-based heuristic for data gathering and aggregation in sensor networks. In *Proceedings of WCNC*. IEEE, 2003.

[26] A. Deligiannakis, Y. Kotidis, and N. Roussopoulos. Hierarchical in-network data aggregation with quality guarantees. In *Proceedings of EDBT*, 2004.

[27] Q. Dong, S. Banerjee, M. Adler, and A. Misra. Minimum energy reliable paths using unreliable wireless links. In *Proceedings of MobiHoc*. ACM, 2005.

[28] G. Durgin, T. Rappaport, and D. Wolf. New analytical models and probability density functions for fading in wireless communications. *IEEE Transactions on Communications*, 50(6):1005–1015, 2002.

[29] F. Fabbri, C. Buratti, and R. Verdone. A multi-sink multi-hop wireless sensor network over a square region: Connectivity and energy consumption issues. In *Proceedings of Globecom Workshop*. IEEE, 2008.

[30] K. Farkas, T. Hossmann, L. Ruf, and B. Plattner. Pattern matching based link quality prediction in wireless mobile ad hoc networks. In *Proceedings of MSWiM*. ACM, 2006.

[31] R. Fonseca, O. Gnawali, K. Jamieson, and P. Levis. Four-bit wireless link estimation. In *Proceedings of HotNets*. ACM, 2007.

[32] S. Gandham, M. Dawande, R. Prakask, and S. Venkatesan. Energy efficient schemes for wireless sensor networks with multiple mobile base stations. In *Proceedings of Globecom*. IEEE, 2003.

[33] M. Garey, D. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1:237–267, 1976.

[34] N. Garg and J. Konemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. In *Proceedings of Foundations of Computer Science*. IEEE, 1998.

[35] V. Gungor and P. Hancke. Industrial wireless sensor networks: Challenges, design principles, and technical approaches. *IEEE Transactions on Industrial Electronics*, 56:4258–4265, 2009.

[36] G. Gupta and M. Younis. Load-balanced clustering of wireless sensor networks. In *Proceedings of ICC*. IEEE, 2003.

[37] M. Halldorsson and J. Radhakrishnan. Greed is good: Approximating independent sets in sparse and bounded-degree graphs. *Algorithmica*, 18:145–163, 1997.

[38] M. Handy, M. Haase, and D. Timmermann. Low energy adaptive clustering hierarchy with deterministic cluster-head selection. In *Proceedings of International Workshop on Mobile and Wireless Communications Network*. IEEE, 2002.

[39] R. Hassin. Approximation schemes for the restricted shortest path problem. *Mathematics of Operations Research*, 17:36–42, 1992.

[40] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proceedings of HICSS*. IEEE, 2000.

[41] W. Heinzelman, J. Kulik, and H. Balakrishna. Negotiation-based protocols for disseminating information in wireless sensor networks. In *Proceedings of Mobicom*. ACM, 1999.

[42] P. Hsiao, A. Hwang, H. Kung, and D. Vlah. Load-balancing routing for wireless access networks. In *Proceedings of INFOCOM*. IEEE, 2001.

[43] S. Hussain and O. Islam. An energy efficient spanning tree based multi-hop routing in wireless sensor networks. In *Proceedings of WCNC*. IEEE, 2007.

[44] D. Johnson. Approximation algorithms for combinatorial problems. In *Proceedings of the ACM Symposium on Theory of Computing*. ACM, 1973.

[45] R. Jothi and B. Raghavachari. Approximation algorithms for the capacitated minimum spanning tree problem and its variants in network design. *ACM Transactions on Algorithms*, 1:265–282, 2005.

[46] D. Karger and C. Stein. A new approach to the minimum cut problem. *Journal of the ACM*, 43:601–640, 1996.

[47] H. Karl and A. Willig. *Protocols and Architectures for Wireless Sensor Networks*. WILEY, 2005.

[48] S. Keshav and S. Morgan. Smart retransmission: Performance with overload and random losses. In *Proceedings of INFOCOM*. IEEE, 1997.

[49] H. Kim, T. Abdelzaher, and W. Kwon. Minimum energy asynchronous dissemination to mobile sinks in wireless sensor networks. In *Proceedings of SenSys*. ACM, 2003.

[50] H. Kim, T. Kwon, and P. Mah. Multiple sink positioning and routing to maximize the lifetime of sensor networks. *Institute of Electronics, Information and Communication Engineers Transactions on Communications*, E91-B:3499–3506, 2008.

[51] H. Kim, Y. Seok, N. Choi, Y. Choi, and T. Kwon. Optimal multi-sink positioning and energy-efficient routing in wireless sensor networks. In *Proceedings of International Conference on Information Networking: Convergence in Broadband and Mobile Networking*. LNCS, 2005.

[52] Y. Kotidis. Snapshot queries: towards data-centric sensor networks. In *Proceedings of ICDE*. IEEE, 2005.

[53] L. Kou, G. Markowsky, and L. Berman. A fast algorithm for steiner trees. *Acta Informatica*, 15:141–145, 1981.

[54] H. Kuhn and B. Yaw. The hungarian method for the assignment problem. *Naval Research Logistics Quart*, 2:83–97, 1955.

[55] Q. Li and D. Rus. Sending messages to mobile users in disconnected ad-hoc wireless networks. In *Proceedings of Mobicom*. ACM, 2000.

[56] J. Lian, K. Naik, and G. Agnew. Data capacity improvement of wireless sensor networks using non-uniform sensor distribution. *International Journal of Distributed Sensor Networks*, 2:121–145, 2006.

[57] W. Liang and Y. Liu. On-line data gathering for maximinizing network lifetime in sensor networks. *IEEE Transactions on Mobile Computing*, 6:2–11, 2007.

[58] W. Liang, J. Luo, and X. Xu. Prolonging network lifetime via a controlled mobile sink in wireless sensor networks. In *Proceedings of Globecom*. IEEE, 2010.

[59] W. Liang, J. Luo, and X. Xu. Network lifetime maximization for time-sensitive data gathering in wireless sensor networks with a mobile sink. *Journal of Wireless Communications and Mobile Computing*, 2011.

[60] S. Lindsey and C. Raghavendra. PEGASIS: Power-efficient gathering in sensor information systems. In *Proceedings of Aerospace Conference*. IEEE, 2002.

[61] Y. Liu, H. Ngan, and L. Ni. Power-aware node deployment in wireless sensor networks. In *Proceedings of the International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing*. IEEE, 2006.

[62] P. Loh. A scalable, efficient and reliable routing protocol for wireless sensor networks. In *Proceedings of International Conference on Ubiquitous Intelligence and Computing*. Springer, 2006.

[63] J. Luo and J. Hubaux. Joint mobility and routing for lifetime elongation in wireless sensor networks. In *Proceedings of INFOCOM*. IEEE, 2005.

[64] J. Luo, J. Panchard, M. Piorkowski, M. Grossblauser, and J. Hubaux. Mobiroute: routing towards a mobile sink for improving lifetime in sensor

networks. In *Proceedings of DCOSS*. IEEE, 2006.

[65] A. Manjeshwar and D. Agrawal. TEEN: a routing protocol for enhanced efficiency in wireless sensor networks. In *Proceedings of IPDPS*. IEEE, 2000.

[66] A. Meier, T. Rein, J. Beutel, and L. Thiele. Coping with unreliable channels: Efficient link estimation for low-power wireless sensor networks. In *Proceedings of International Conference Networked Sensing Systems*. IEEE, 2008.

[67] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[68] C. Olston, J. Jiang, and J. Widom. Adaptive filters for continuous queries over distributed data streams. In *Proceedings of SIGMOD*. ACM, 2003.

[69] E. Oyman and C. Ersoy. Multiple sink network design problem in large scale wireless sensor networks. In *Proceedings of ICC*. IEEE, 2004.

[70] J. Pan, L. Cai, Y. Hou, Y. Shi, and S. Shen. Optimal base-station locations in two-tiered wireless sensor networks. *IEEE Transactions on Mobile Computing*, 4:458–473, 2005.

[71] C. Papadimitriou. The complexity of the capacitated tree problem. *Networks*, 8:217–230, 1978.

[72] W. Poe and J. Schmitt. Placing multiple sinks in time-sensitive wireless sensor networks using a genetic algorithm. In *Measuring, Modelling and Evaluation of Computer and Communication Systems*. IEEE, 2008.

[73] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *Proceedings of Sensys*. ACM, 2004.

[74] G. Pottie. Wireless sensor networks. In *Proceedings of Information Theory Workshop*. IEEE, 1998.

[75] L. Qiu, R. Chandra, K. Jian, and M. Mahdian. Optimizing the placement of integration points in multi-hop wireless sensor networks. In *Proceedings of ICNP*. IEEE, 2004.

[76] F. Ren, J. Zhang, C. L. T. He, and S. Ren. EBRP: Energy-balanced routing protocol for data gathering in wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 22:2108–2125, 2011.

[77] R. Shah, S. Jain, and W. Brunette. Data mules: Modeling a three-tier architecture for sparse sensor networks. In *Proceedings of International Workshop on Sensor Network Protocols and Applications*. IEEE, 2003.

[78] F. Shan, W. Liang, J. Luo, and X. Shen. Network lifetime maximization for time-sensitive data gathering in wireless sensor networks. *Computer Networks*, 57:1063–1077, 2013.

[79] K. Sharma, R. Patel, and H. Singh. A reliable and energy efficient transport protocol for wireless sensor networks. *International Journal of Computer Networks & Communications*, 2:92–103, 2010.

[80] Y. Shi and Y. Hou. Theoretical results on base station movement problem for sensor network. In *Proceedings of INFOCOM*. IEEE, 2008.

[81] A. Somasundara, A. Ramamoorthy, and M. Srivastava. Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines. In *Proceedings of RTSS*. IEEE, 2004.

[82] A. Srinivas and E. Modiano. Minimum energy disjoint path routing in wireless ad-hoc networks. In *Proceedings of Mobicom*. ACM, 2003.

[83] F. Stann and J. Heidemann. RMST: reliable data transport in sensor networks. In *Proceedings of International Workshop on Sensor Network Protocols and Applications*. IEEE, 2003.

[84] R. Sugihara and R. Gupta. Optimizing energy-latency trade-off in sensor networks with controlled mobility. In *Proceedings of INFOCOM*. IEEE, 2009.

[85] R. Tan, G. Xing, J. Chen, W. Song, and R. Huang. Quality-driven volcanic earthquake detection using wireless sensor networks. In *Proceedings of*

RTSS. IEEE, 2007.

[86] M. Vuran, O. B. Akan, and I. Akyildiz. Spatio-temporal correlation: theory and applications for wireless sensor networks. *Computer Networks*, 45:245–259, 2002.

[87] C. Wan, A. Campbell, and L. Krisnamurthy. Pump-slowly, fetch-quickly (PSFQ): A reliable transport protocol for wireless sensor networks. In *Proceedings of WSNA*. IEEE, 2002.

[88] Q. Wang, G. Takahara, H. Hassanein, and K. Xu. On relay node placement and locally optimal traffic allocation in heterogeneous wireless sensor networks. In *Proceedings of LCN*. IEEE, 2005.

[89] Q. Wang, K. Xu, G. Takahara, and H. Hassanein. Locally optimal relay node placement in heterogeneous wireless sensor networks. In *Proceedings of Globecom*. IEEE, 2005.

[90] Z. Wang, S. Basagni, E. Melachrinoudis, and C. Petrioli. Exploiting sink mobility for maximizing sensor networks lifetime. In *Proceedings of HICSS*. IEEE, 2005.

[91] T. Wark, C. Crossman, W. Hu, Y. Guo, P. Valencia, P. Sikka, P. Corke, C. Lee, J. Henshall, K. Prayaga, J. O'Grady, M. Reed, and A. Fisher. The design and evaluation of a mobile sensor/actuator network for autonomous animal control. In *Proceedings of IPSN*. IEEE, 2007.

[92] 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*, 19:710–720, 2008.

[93] G. Xing, T. Wang, W. Jia, and M. Li. Rendezvous design algorithms for wireless sensor networks with a mobile base station. In *Proceedings of MobiHoc*. ACM, 2008.

[94] K. Xu, H. Hassanein, G. Takahara, and Q. Wang. Relay node deployment

strategies in heterogeneous wireless sensor networks. *IEEE Transactions on Mobile Computing*, 9:145–159, 2010.

[95] K. Xu, Q. Wang, H. Hassanein, and G. Takahara. Optimal wireless sensor networks (WSNs) deployment: minimum cost with lifetime constraint. In *IEEE International Conference on Wireless And Mobile Computing, Networking And Communications*. IEEE, 2005.

[96] N. Xu, S. Ranngwala, K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin. A wireless sensor networks for structural monitoring. In *Proceedings of SenSys*. ACM, 2004.

[97] X. Xu, Y. Hu, J. Bi, and W. Liu. Adaptive nodes scheduling approach for clustered sensor networks. In *IEEE Symposium on Computers and Communications*. IEEE, 2009.

[98] X. Xu and W. Liang. Monitoring quality optimization in wireless sensor networks with a mobile sink. In *Proceedings of MSWiM*. ACM, 2011.

[99] X. Xu and W. Liang. Placing optimal number of sinks in sensor networks for network lifetime maximization. In *Proceedings of ICC*. IEEE, 2011.

[100] X. Xu, W. Liang, X. Jia, and W. Xu. Maximizing network throughput with minimal remote data transfer cost in unreliable wireless sensor networks. In *Proceedings of MobiHoc*. ACM, 2013.

[101] X. Xu, W. Liang, and T. Wark. Data quality maximization in sensor networks with a mobile sink. In *Proceedings of DCOSS*. IEEE, 2011.

[102] X. Xu, W. Liang, and Z. Xu. Minimizing remote monitoring cost of wireless sensor networks. In *Proceedings of WCNC*. IEEE, 2013.

[103] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad hoc routing. In *Proceedings of Mobicom*. ACM, 2001.

[104] T. Yamada, H. Takahashi, and S. Kataoka. A heuristic algorithm for the minimax spanning forest problem. *European Journal of Operational Research,*

91:565–572, 1996.

[105] F. Ye, G. Zhong, S. Lu, and L. Zhang. A robust data delivery protocol for large scale sensor networks. In *Proceedings of IPSN*. ACM, 2003.

[106] M. Younis and K. Akkaya. Strategies and techniques for node placement in wireless sensor networks: A survey. *Ad Hoc Networks*, 6:621–655, 2008.

[107] M. Younis, M. Youssef, and K. Arisha. Energy-aware management for cluster-based sensor networks. *Computer Networks*, 43:649–668, 2003.

[108] W. Youssef and M. Younis. Intelligent gateways placement for reduced data latency in wireless sensor networks. In *Proceedings of ICC*. IEEE, 2007.

[109] Y. Zhang and M. Fromherz. Constrained flooding: a robust and efficient routing framework for wireless sensor networks. In *Proceedings of International Conference on Advanced Information Networking and Applications*. IEEE, 2006.

[110] 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*, 60:400–415, 2010.