THE RESEARCH ON EXTENDING THE LIFETIME OF WIRELESS SENSOR NETWORKS

By

YOUNGSANG YUN

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2010
To my loving family
ACKNOWLEDGMENTS

I would like to begin by thanking Professor Ye Xia, my dissertation advisor and mentor during past several years. Dr. Ye Xia has been the greatest advisor, providing me with valuable ideas, support, and help during my graduate studies. His wide and deep knowledge and his enthusiasm always inspires me.

My special thanks also go to Prof. Jonathan Liu, Prof. Alin Dobra, Prof. Shigang Chen and Prof. Cole J. Smith for their comments and support during my studies.

Last but not least, I want to thank my family for their love while I am studying. Without their sacrifice, I would not be what I am. I would like to say that I love Chris, Han, and my wife, KyungHee.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>4</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>7</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>8</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>10</td>
</tr>
<tr>
<td>CHAPTER</td>
<td></td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>12</td>
</tr>
<tr>
<td>1.1 Contributions</td>
<td>15</td>
</tr>
<tr>
<td>1.2 Organization of the Study</td>
<td>16</td>
</tr>
<tr>
<td>2 EXPERIMENTS ON THE MINIMUM TRANSMISSION ENERGY ROUTING,</td>
<td>18</td>
</tr>
<tr>
<td>2.1 Overview</td>
<td>18</td>
</tr>
<tr>
<td>2.2 Related Works</td>
<td>19</td>
</tr>
<tr>
<td>2.3 System Model</td>
<td>22</td>
</tr>
<tr>
<td>2.3.1 Energy Model</td>
<td>23</td>
</tr>
<tr>
<td>2.3.2 Simulation Model Of Minimum Transmission Energy Routing</td>
<td>24</td>
</tr>
<tr>
<td>2.3.3 Linear Programming Model</td>
<td>25</td>
</tr>
<tr>
<td>2.4 Experiments</td>
<td>26</td>
</tr>
<tr>
<td>2.4.1 Graph Generation</td>
<td>26</td>
</tr>
<tr>
<td>2.4.2 Lifetime vs. Ratio</td>
<td>29</td>
</tr>
<tr>
<td>2.4.3 Effect Of Node Densities</td>
<td>29</td>
</tr>
<tr>
<td>2.4.4 Effect Of The Number Of Sinks</td>
<td>32</td>
</tr>
<tr>
<td>2.4.5 Effect Of Node Densities And The Number Of Sinks</td>
<td>34</td>
</tr>
<tr>
<td>2.4.6 Effect Of Node Deployment Strategies</td>
<td>35</td>
</tr>
<tr>
<td>2.4.7 Effect Of Communication Ranges</td>
<td>35</td>
</tr>
<tr>
<td>2.5 Experiments on the Minimum Transmission Energy Routing, Sequential</td>
<td>37</td>
</tr>
<tr>
<td>Minimum Transmission Energy Routing, and Optimal Lifetime Routing</td>
<td></td>
</tr>
<tr>
<td>2.6 Summary</td>
<td>38</td>
</tr>
<tr>
<td>3 A METHOD FOR DECIDING NODE DENSITY IN NON-UNIFORM DEPLOYMENT OF</td>
<td>40</td>
</tr>
<tr>
<td>WIRELESS SENSOR NETWORKS</td>
<td></td>
</tr>
<tr>
<td>3.1 Overview</td>
<td>40</td>
</tr>
<tr>
<td>3.2 Models with Discrete Ring Structure</td>
<td>44</td>
</tr>
<tr>
<td>3.2.1 Sensor Field and Energy Consumption Models</td>
<td>45</td>
</tr>
<tr>
<td>3.2.2 Deriving the Node Densities of the Rings</td>
<td>47</td>
</tr>
<tr>
<td>3.2.2.1 Case of density-independent routing</td>
<td>50</td>
</tr>
<tr>
<td>Table</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>2-1</td>
<td>System parameters used in the simulation</td>
</tr>
<tr>
<td>4-1</td>
<td>Experimental parameters and their values</td>
</tr>
<tr>
<td>5-1</td>
<td>Performance comparison between CPLEX and Algorithm 3</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>Graph example 1: node deployment by the uniform distribution</td>
<td>28</td>
</tr>
<tr>
<td>2-2</td>
<td>Graph example 2: node deployment by grid based strategy</td>
<td>30</td>
</tr>
<tr>
<td>2-3</td>
<td>Lifetime vs. ratio of remaining energy to the initial energy</td>
<td>31</td>
</tr>
<tr>
<td>2-4</td>
<td>Performance in various node densities</td>
<td>32</td>
</tr>
<tr>
<td>2-5</td>
<td>Performance in various number of sinks</td>
<td>33</td>
</tr>
<tr>
<td>2-6</td>
<td>Performance in different number of nodes and different number of sinks</td>
<td>34</td>
</tr>
<tr>
<td>2-7</td>
<td>Performance of different node deployment strategies</td>
<td>36</td>
</tr>
<tr>
<td>2-8</td>
<td>Performance of different communication ranges</td>
<td>37</td>
</tr>
<tr>
<td>2-9</td>
<td>Performance of different dimension of sensor fields</td>
<td>38</td>
</tr>
<tr>
<td>3-1</td>
<td>Sensor field model</td>
<td>46</td>
</tr>
<tr>
<td>3-2</td>
<td>Uniform node selection</td>
<td>54</td>
</tr>
<tr>
<td>3-3</td>
<td>Node densities and average per-node energy consumption rates for various maximum jump sizes, (l), under uniform ring selection. (\alpha = 2).</td>
<td>58</td>
</tr>
<tr>
<td>3-4</td>
<td>Node densities and average per-node energy consumption rates for various maximum jump sizes, (l), under uniform node selection. (\alpha = 0).</td>
<td>59</td>
</tr>
<tr>
<td>3-5</td>
<td>Node densities and average per-node energy consumption rates for various maximum jump sizes, (l), under uniform node selection. (\alpha = 1).</td>
<td>60</td>
</tr>
<tr>
<td>3-6</td>
<td>Node densities and average per-node energy consumption rates for various maximum jump sizes, (l), under uniform node selection. (\alpha = 2).</td>
<td>61</td>
</tr>
<tr>
<td>3-7</td>
<td>Node densities and average per-node energy consumption rates for various maximum jump sizes, (l), under uniform node selection. (\alpha = 3).</td>
<td>62</td>
</tr>
<tr>
<td>4-1</td>
<td>Examples of the static sink model (SSM), mobile sink model (MSM), and delay tolerant mobile sink model (DT-MSM)</td>
<td>77</td>
</tr>
<tr>
<td>4-2</td>
<td>Comparison of lifetimes of MSM and DT-MSM under the various radii of coverage</td>
<td>89</td>
</tr>
<tr>
<td>4-3</td>
<td>Lifetime against the number of sink locations; maximum coverage; (e = 2.0)</td>
<td>90</td>
</tr>
<tr>
<td>4-4</td>
<td>Lifetime against the number of sink locations; minimum coverage; (e = 3.0)</td>
<td>91</td>
</tr>
<tr>
<td>4-5</td>
<td>Lifetimes versus transmission range, (d): (</td>
<td>\mathcal{V}</td>
</tr>
<tr>
<td>5-1</td>
<td>Expanded graph of DT-MSM</td>
<td>97</td>
</tr>
</tbody>
</table>
5-2 Convergence to the optimal value, $z^*$ .................................................. 118
5-3 Lyapunov drift of the algorithm over time ....................................................... 119
5-4 Time average of total virtual queue size over time ........................................... 119
5-5 Long-time average of the difference between outflow and inflow ................. 120
Abstract of dissertation Presented to the Graduate School of the University of Florida in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

THE RESEARCH ON EXTENDING THE LIFETIME OF WIRELESS SENSOR NETWORKS

By
YoungSang Yun

August 2010

Chair: Ye Xia
Major: Computer Engineering

We study various problems efficient energy management for wireless sensor networks. First, we research the energy efficient deployment of wireless sensor nodes so that energy consumption rates of all nodes are equal during the lifetime of the wireless sensor network. If the sensors are deployed uniformly across the network, they experience different traffic intensities and energy depletion rates depending on their locations. Usually, the sensors near the sink tend to deplete their energy sooner; when enough of them exhaust their energy, they leave holes in the network, causing the remaining nodes to be disconnected from the sink. One of the solutions to this energy-hole problem is to deploy the sensors non-uniformly. Moreover, we describe a method for deciding the sensor deployment densities so as to equalize the energy consumption rates of all nodes. The method is general and can be applied to other objectives and constraints.

Second, we propose a framework to maximize the lifetime of the wireless sensor networks by using a mobile sink when the underlying applications tolerate delayed information delivery to the sink. Within a prescribed delay tolerance level, each node does not need to send the data immediately as it becomes available. Instead, the node can store the data temporarily and transmit it when the mobile sink is at the most favorable location for achieving the longest WSN lifetime. We call the proposed framework as Delay-Tolerant Mobile Sink Wireless Sensor Network. To find the best
solution within the proposed framework, we formulate optimization problems that maximize the lifetime of the WSN subject to the delay bound constraints, node energy constraints, and flow conservation constraints. We conduct extensive computational experiments on the optimization problems and find that the lifetime can be increased significantly as compared to not only the stationary sink model but also more traditional mobile sink models. We also show that the delay tolerance level does not affect the maximum lifetime of the WSN.

Third, we propose an adaptive and potentially decentralized algorithm for the DT-MSM. The distributed routing algorithms are very important in developing a practical routing protocol. Distributed algorithms are generally free from the network scalability issues in several reasons. They do not need to have knowledge about the whole network configurations and they also do not require the central node to compute the routes for all nodes in the network. Lagrange multiplier method solves dual of the primal problem. Dual problem sometimes has a nice structure with which we can decompose the dual problem into several sub-problems. We use a subgradient projection method to solve the dual problem and. A sensor node in our method keeps virtual queue which is a scalar product of the Lagrange multiplier and it is used in solving sub-problems. We propose (a possibly distributed implementable) decentralized algorithms for solving sub-problems. Moreover, we analytically show the our algorithm finds a solution arbitrarily close to the optimal solution of the primal problem. It is verified through the numerical experiments.
CHAPTER 1
INTRODUCTION

The technological breakthroughs in MEMS (Micro Electro Mechanical Systems), DSP (Digital Signal Processing), integrated circuit technologies, microprocessor hardware, and wireless communication techniques have been phenomenal since last few decades. In addition, the technological advance that researchers and engineers have achieved in ad-hoc networking routing and protocol, pervasive computing, embedded system technologies make it possible to mass-manufacture the low cost, small-sized form factored, and versatile sensor nodes. They integrate general purpose processors, wide varieties of sensing devices, and wireless communication devices. A Wireless Sensor Networks typically consists of these cheap sensor nodes deployed into the targeted area to be monitored. However, to make the cost of deploying wireless sensor network to be low, a sensor node has an on-board battery, as well as a low power processor and a limited memory space. Therefore it is necessary to make sensor nodes collaborate with other sensor nodes to overcome this limitation. Since a Wireless Sensor Network (WSN) is constructed with a huge number of sensor nodes, which are densely and sometimes randomly deployed into the region of our interest, the location of sensor nodes is not known at the time of deployment. Occasionally the sensor nodes need to be deployed randomly into the hostile or hazardous terrain so that it is not easy or even impossible to access that region. Thus, the protocols or algorithms used in the WSN should have the ability of organizing the network autonomously. Furthermore, since the number of deployed sensor nodes is often tremendous, an untethered operation for the individual sensor node is required.

The mission imposed on the sensor nodes is gathering information about the surrounding environment, processing sensed raw data such as compression or quantization, and transferring processed data to special locations called the sinks for further processing. The sinks are typically more powerful in processing power and
richer in energy reservoir. The sinks sometimes connected to the wired network like the Internet, thus played as a gateway to the outside world for the wireless sensor networks.

Due to the limitation of available energy or constrained capability of wireless transceiver, it is not feasible for sensor node to communicate with the sink directly. Instead, a sensor node uses other sensor nodes as next hops in order to get to the sink. For this reason, the communication behavior in the wireless sensor network shows multi-hop pattern. Therefore the individual wireless sensor node plays the roles of a source of the data and a relayer of the data for other nodes. This feature of WSN makes it very similar to the Mobile Ad-hoc Networks (MANET) in that communication is done in multi-hop fashion. However, there is a huge difference between these two types of network. While the connectivity is the ultimate goal of the MANET, the longevity of network is the primary objective of the WSN.

The functions of sensors are very diverse: seismic, magnetic, thermal, acoustic, visual, and radioactive. As the variety of the functions of sensor, the potential applications of the WSN is also boundless. According to [2], the applications of WSN can be categorized into the military applications, environmental applications, health care applications and home applications. But this classification can be broaden into more categories, such as applications in space exploration or disaster relief.

Since sensor nodes reply on the embedded battery power, sometimes replenishment of battery is very costly. When considering the number of sensor nodes deployed it does not seem possible. Therefore clever management of energy reservoir is required to extend the lifetime or improve the throughput of the WSN. There have been several challenging research issues on the efficient energy management.

Typically sensor node is able to alter its transmission power and as a consequence, it can change reachable transmission distance. By changing the transmission power, one can make a sensor node have the different set of neighbor nodes at times. In the perspective of network, different topologies can be obtained by the adjustment
of transmission power at each sensor node, so that resulting topology satisfies the application specific constraints such as the degree of the connectivity. This is the definition of the topology control. The proper adjustment of the transmission power also has a good side-effect. It reduces the possibility of interferences at the receiver and leads to the increase of the throughput. However, due to immense number of sensor nodes, topology control should be distributed. Moreover it also should be dynamic because of frequent node failures.

In WSN, a sensor node might be in one of three states: active, idle, and sleep. A sensor node consumes significant amount of energy even it is in an idle state. In order to save the energy, a sensor node sometimes goes into the sleep state. Thus, how to schedule sleep-wake sequence is also another research issues. Since it is not possible to synchronize the sleep-wake schedule for every sensor node, delay is essentially increased. Good schedule should minimize incurred delay as possible.

In the data centric WSN, without data aggregation, the sink might acquire much redundant data, because lots of sensor nodes would report the same event if they are located closely. There are lots of ways to aggregate data. One of these is clustering the sensor nodes. In each cluster, the cluster head aggregates raw data gathered by the sensor nodes in its own cluster. Data aggregation is very useful in reducing the energy consumption accompanied by communication as well.

Widely used routing protocol in the wired network may not be a good choice in the perspective of efficient energy management. To ensure the longevity of each sensor nodes, the routing path going through the node having insufficient energy should be avoided. Lots of routing protocols taking the current energy level of the sensor node into account have already been proposed. This kind of routing protocol is called Power-aware routing protocol. The minimum total transmission energy paths and maximum residual energy paths are such examples. With the help of a mature optimization theory,
optimal path to maximize the lifetime of the system can be also used in the routing algorithms.

If the mobility can be applied to the component of WSN, the better performance might be expected. The mobile sink seems a preferential choice rather than the mobile sensor nodes. When the sink has a capability of moving, overall energy consumption of particular sensor nodes can be mitigated as contrasted with the network where the sink does not move because a heavy relaying burden also tends to follow the sink’s movement.

1.1 Contributions

The contributions of this work are mainly discussed in chapter 3, chapter 4, and chapter 5. First, we study the density control that makes every sensor node experience the same rate of energy consumption. The same energy consumption for all sensor node is a key factor in maximizing the lifetime of the WSN. Our density control necessitates a non-uniform node deployment where the density at a point is determined according to the routing protocol used and its distance from the sink node. We propose a very general routing model that captures various existing routing protocol strategies. We also propose an iterative method that eventually finds densities of the ring where the sensor nodes are almost similarly distant from the sink.

Second, we propose new framework that exploits the mobile sinks. The application we are interested in can tolerate some extent of delay of delivery of information collection of sensed data. We formulate the Linear Programming problem that reflects features of our framework. Through the simulation, we also show its performance is better than other models: Static Sink Model and General Mobile Sink Model.

Third, we devise the adaptive and potentially distributed algorithm for the framework we propose. Our algorithm can work only with the information about the current queue size, receiving and transmitting traffic at the sensor node. We show that our algorithm find a solution which is arbitrarily close to the optimal solution.
1.2 Organization of the Study

Rest of this dissertation consists as follows: In chapter 2, as a preliminary work, We compare the lifetimes of several routing protocols which might be used in the WSN. The simplest one might be the Minimum Transmission Energy (MTE) routing in which every sensor nodes greedily choose the path along which total energy consumption in transmission is minimal. On the other hand, We can use the maximal lifetime routing in which routing paths are calculated by solving the optimization problem with the global view of the network topology. In the MTE, after the first node dies due to energy exhaustion, a large number of nodes still possess plenty of energy. Thus, rather than to pause the operation of the whole system, the network can continue working as long as the connectivity to the sink is preserved from all remaining sensor nodes. In the Sequential Minimum Transmission Energy protocol (SMTE), alive sensor nodes keep routing based on MTE after the failed sensor node is removed from the network topology until the sink is isolated from the network. These three routing protocols are compared through extensive experiments with various simulation settings.

In chapter 3, We discuss non-uniform node deployment strategies. When nodes are uniformly distributed in the region, the energy hole problem near the sink may not be an avoidable phenomenon. The rationale of non-uniform deployment is to put more nodes to the place where relaying burden is excessive. The area nearby the sink is such a place. In this chapter, We solve the problem of how many nodes should be deployed in a specific area of the sensor network. The sending (or outgoing ) traffic from a node is composed of relaying traffic and self-generated traffic. We establish the recurrence relations showing how the outgoing traffic of nodes is related according to the distance from the sink, when a geographical routing is used. In addition, We mention the way to obtain the solution that satisfying the system of recurrence relation.

In chapter 4, We study how the mobility would improve the lifetime of the WSN. Especially the mobility of the sink, or the mobile sink, is considered in this chapter. To
mitigate the energy hole problem, the sink needs to be moved around the area where the sensor nodes are deployed. By moving the sink, the traffic concentrated points may be moved along with the sink because those points are usually occurred near the sink. Assuming that the application over the WSN can tolerate some extent of delay, We propose the new framework that extends the lifetime of the WSN. We call it DT-MSM, Delay Tolerant Mobile Sink Model. We also present the experimental results showing that our framework outperforms other WSN models.

In chapter 5, We extend our work on the DT-MSM proposed in the chapter 4. Although the DT-MSM shows substantial improvement of the performance, it requires solution from the Linear Programming problem. However, there is no way to solve the Linear Programming problems in the distributed manner and even We need a very powerful computing node which runs LP solver. To be a practical routing method, it is necessary to be implementable in a decentralized manner. In this chapter, We propose a partially distributed algorithm implementing our framework. The proposed algorithm does not require an LP solver.

In chapter 6, We conclude this dissertation.
CHAPTER 2
EXPERIMENTS ON THE MINIMUM TRANSMISSION ENERGY ROUTING, SEQUENTIAL MINIMUM TRANSMISSION ENERGY ROUTING, AND OPTIMAL LIFETIME ROUTING

2.1 Overview

Recent advances in micro-electro-mechanical technology, wireless networking, and integration technology provide new applications in various areas. Wireless sensor network is the one of these areas and may be perhaps the most active research issue. Sensor nodes can be implemented by integrating various sensing capabilities and wireless communication into low cost and small form-factor embedded system without paying much cost in these days. Wireless sensor networks which are the communication infrastructure formed from massively deployed such sensor nodes in ad-hoc manner have a variety range of application. For examples, environment monitoring (vehicular traffic monitoring, wild life habitat monitoring, etc.) and military surveillance.

Sensor nodes typically operate with a limited on-board battery and it’s very hard or even impossible to replace the battery or replenish the energy of the nodes. Thus, the longevity of wireless sensor networks has been considered as a primary goal in wireless sensor network researches and many works to prolong the lifetime of wireless sensor networks have been published [12] [11] [19] [35] [45]. On the other hand, some researchers focus on how to apply the proposed algorithms to the real world, for instance, distributed implementation of the linear programming approaches (refer to [26] [35]) or approximated algorithms (refer to [35] [15] [3] [45]).

In general, there are many factors that defines the lifetime of the sensor network, for example, the size of the sensor field, the number of sensor nodes, the number of the sink, etc. In practice, knowing how these parameters affect the lifetime of the networks. However, investigating this problem taking all these into account is so difficult task. As a preliminary step of this research, we do extensive simulations to get some intuition on the relationships that maybe exist between the lifetime and these factors.
2.2 Related Works

In [19], the authors propose the LEACH (Low-Energy Adaptive Clustering Hierarchy) to minimize global energy consumption by distributing the relaying burden to all the nodes. Sensor nodes are divided into several clusters and in each cluster there is a cluster head which is responsible for relaying the all the traffic generated in the corresponding cluster to the sink. In addition, the role of cluster head is not fixed to the specific node, that is, the cluster head is assigned to the highest energy node in an adaptive manner, which improves the lifetime of sensor network significantly.

In [12], the authors define the sensor network lifetime as the time until the node drains out its energy at the first time. They also show that minimum energy routes which minimizes energy consumptions as possible while relaying the traffic at each node are not good in the perspective of the lifetime of the sensor networks due to uneven energy depletion behavior. That is, the minimum energy routes causes the fast energy consumption rate of some nodes which are on the frequently used routes. The authors formulate the routing problem as an optimization problem where the objective is to find the flows that maximize the sensor network lifetime.

Formulation of maximizing the lifetime as a Linear Program can be found in many literatures. In [26], the authors propose two distributed algorithms for the same problem as [12]. The key idea of their algorithms is to solve dual problem instead of solving the primal problem directly. By changing the objective functions of dual problem slightly (In this case, the object function is no more linear function), We can make the separable nonlinear program problem, and this nice structure of the problem make easy to devise a distributed algorithm for solving maximum lifetime problem. One of their algorithms is a partially distributed one, the other can be implemented in fully distributed manner. However, both algorithms use a well-known sub-gradient algorithm. [35] also defines the maximum lifetime routing problem as a variant of maximum concurrent flow problem (MCFP), a sort of linear program problem. The original MCFP defined in [37] is a
multi-commodity flow problem in which all pairs of nodes can concurrently send or receive flow. The flow is sustained through links with a certain capacities. The MCFP tries to find the flow such that the ratio of the flow between each pair of nodes to the demand between that pair, so called throughput, is the same for all pairs of nodes. In addition, the solution flow should observe the flow conservation law and capacity constraint of the link. That is, the objective of the MCFP is to find the flow maximizing throughput in the multi-commodity network. However MCFP problem used in [35] is little bit different from the original one in that energy constraints, which is caused by the limited energy of the nodes, are used in the problem formulation instead of capacity constraints of the links. The objective is to find the flow maximizing the minimum lifetime of the system while satisfying above constraints. In [35], the authors propose an distributed and iterative approximation algorithms for the MCFP described above: Every node maintains the same number of queues as commodities for its links. The key of the algorithm is to keep each queue for a link equalized at any time.

Typically, the approximation algorithms would not give us optimal solutions, but with these algorithms we can obtain the solutions very fast and they are close enough to the optimal solution with an acceptable error. During the past several years, lots of approximation algorithms have been published [3, 4, 13, 15, 45]. The performance of the approximation algorithms, in terms of time complexity, are normally expressed as functions of several parameters including a tolerable error range.[15] solves dual problem with an iterative method. The problem is to find the largest $\lambda$ such that there is a multi-commodity flow to which $\lambda d(i)$ units of commodity $i$ is assigned. Each edge of the graph has positive capacity ($c : E \rightarrow R$) and there are $k$ commodities with a demand $d(i)$ ($i = 1, 2, \ldots, k$). Let's $x(P)$ be the amount of flow on some path $P$. then primal
problem can be expressed as follows:

$$\begin{align*}
\text{max} & \quad \lambda \\
\text{subject to} & \quad \sum_{P:e \in P} x(P) \leq c(e) \quad \text{for all } e \in E \quad (2-2) \\
& \quad \sum_{P \in P_j} \geq \lambda d(j) \quad \text{for all } j \in V \quad (2-3) \\
& \quad x(P) \geq 0 \quad \text{for all } P \quad (2-4)
\end{align*}$$

Now, the dual problem is like:

$$\begin{align*}
\text{min} & \quad \sum_e c(e)l(e) \\
\text{subject to} & \quad \sum_{e \in P} l(e) \geq z_j \quad \text{for all } j \in V, P \in P_j \quad (2-6) \\
& \quad \sum_{j=1}^k d(j)z_j \geq 1 \quad (2-7) \\
& \quad l(e) \geq 0 \quad \text{for all } e \in E \quad (2-8) \\
& \quad z_j \geq 0 \quad \text{for all } j \in V \quad (2-9)
\end{align*}$$

Note that $l(e), z_j$ are dual variable associated to the constraints for the edges and nodes in the primal problem, respectively. In the development of the approximation algorithm, the dual problem can be considered as an assignment of lengths ($l : E \rightarrow R$) to the edges such that $D(l)/\alpha(l)$ is minimized, where $\alpha(l) = \sum_{j=1}^k \text{mincost}_j(l)$ and $D(l) = \sum_e l(e)c(e)$. $\text{mincost}_j(l)$ is the minimum cost of $d(j)$ units of commodity to flow from source of $j$ to sink of $j$. Note that $l(e)$ means the cost of one unit of commodity to flow along edge $e$ and actually is the dual variable of the dual problem. The authors also show that if the shortest path algorithm can be used as a subroutine in each iteration instead of finding the minimum cost flow for a single commodity, convergence time of the algorithm can be improved. [3, 4] is very similar to [35] in that they are iterative algorithms trying to balancing the queue length for a single link (minimizing the
difference between head queue length and tail queue length for a link) at each iteration. However, while the latter has energy constraints on the nodes, the formers have capacity constraints on links. The algorithm proposed in [13] is almost identical to that of [15], but by changing the termination condition, running time of the [13] is now independent of the number of commodities. [45] also solves the MCFP based on the algorithm of [15], but the authors uses aggregation tree, which is an aggregated structure of unicast routes from all sources to the common sink. The running time to find aggregation tree for a common sink is not that much larger than that of finding the shortest path between an individual source to the common sink. Thus, instead of calculating the shortest path for every sources, [45] tries to find the aggregation tree for the common sink.

In other hands, [17] solves given Linear Program for a simple and regular topologies, for example linear array topology, with an analytical manner. However, since their analysis is restricted to the regular topology, it makes no sense to apply their analysis in real world.

Some researchers study how the multi-path routing can contribute to reduce the imbalance of energy burden of the sensor nodes. For example, [5] explains the optimizing trade-offs between the energy cost of spreading traffic and the improved spatial balance of energy burden. The authors propose the multi-path routing based on the node proximity. Their algorithm, in fact, is an heuristic approach to prevent energy hole around the sink node from occurring at a rather early time.

2.3 System Model

Consider a wireless sensor network with \( n \) sensors and base stations (or sinks). Assume that the location of the sensors and sinks are fixed and known in advance. We can model such a sensor network as a undirected graph \( G(N, L) \) where \( N \) is the set of sensor nodes and \( L \) is the set of edges between some two nodes. And also an individual node is represented as \( n_i \) and link between node \( n_i \) and \( n_j \) is denoted by \( e_{ij} \).
Let’s denote $N_i$ as the set of neighbor nodes which node $i$ can reach with a certain transmit power level. Each node $i$ is assumed to have a limited energy level to begin with or an initial energy level, denoted by $E_i$ which never be replenished during its lifetime and let $R(i)$ be the data generation rate which means the amount of data that needs to be forwarded to the sink during unit time interval. Note that for the single sink system, incoming data rate for the sink, denoted as $s$, should be equal to the aggregate of the data generation rate of all nodes except the sink. We can express a foregoing statement as $R(s) = -\sum_{i \in N, i \neq s} R(i)$.

2.3.1 Energy Model

Energy required for a node $i$ to transmit a unit of data to a node $j$, denoted as $e_{ij}^t$, is given by,

$$e_{ij}^t = \alpha + \beta d_{ij}^\delta$$  \hspace{1cm} (2–10)

while the energy consumed by a node $i$ in receiving a unit of data from a node $j$, denoted as $e_{ji}^r$ is given by,

$$e_{ji}^r = \alpha$$  \hspace{1cm} (2–11)

where $d_{ij}$ is the Euclidean distance between node $i$ and node $j$, and $\alpha$ is a consumed energy to run the transmitter or receiver circuitry in the sensor, and $\beta$ is the required energy to run the transmitter amplifier[19]. Although $\delta$ should be determined from various environmental factors, typically it is a constant between 2 and 4. For example, in free space propagation model, $\delta$ is considered as 2, so that energy consumption for the transmission of a single unit of data is proportional to the square of the distance [33].

As for the initial setting of networks, we assume that every source has a connectivity to the sink, that is, there are at least one path to the sink from every sources.
2.3.2 Simulation Model Of Minimum Transmission Energy Routing

Several routing algorithms for wireless sensor networks have been proposed. Among them, minimum transmission energy(MTE) routing has an objective to minimize the total transmission energy during lifetime, which is the time until the first node died. The typical approach to the minimum transmission energy routing is to apply the shortest path algorithm to the graph in which the edge cost is the energy that is required to transmit a unit of data between two nodes \[45\]. Refer to 2.3.1 for more details. In fact, MTE is another optimization problem and we can formulate it as a Linear Program:

\[
\begin{align*}
\text{min} & \quad \sum_{i \in N} \sum_{j \in N_i} e_{ij} x_{ij} \\
\text{s.t.} & \quad \sum_{j : j \in N} x_{ij} - \sum_{j : j \in N_i} x_{ji} = \begin{cases} R(i) & \forall i \in N, i \neq s \\ -\sum_{k \in N, k \neq s} R(i) & \text{otherwise} \end{cases} \\
& \quad T \sum_{j \in N_i} e_{ij} x_{ij} \leq E(i) \quad \forall i \in N \\
& \quad x_{ij} \geq 0 \quad \forall (i, j) \in L
\end{align*}
\]

If the \(\delta\) defined in the 2.3.1 equals to 1, then MTE is equivalent to Minimum Hop Routing. Due to the triangular inequality, MTE prefers the longer edges to get to the neighbor close to the sink. If the \(\delta\) is greater than 1, MTE prefers the paths which consist of many shorter edges to the paths of few longer edges.

However, the lifetime of the MTE strategy can perform arbitrarily badly, because this does not consider the residual energy of the intermediate nodes along the paths, thus can cause an unbalanced energy consumption distribution: the nodes on the minimum energy path to the sink are drained their energy very fast, so that entire network can be partitioned \(\cite{30,39}\). \(\cite{30}\) and \(\cite{39}\) argue that by the time the sensors close to the sink is drained its energy, sensors farther away from the sink still have plentiful energy. From
the experiments, we also observe that when the first node dies due to the depletion of
energy, almost 95 percents of energy of the remaining nodes is still unused.

Now let’s change the definition of the network lifetime from the time until the first
node died to the time until network is disconnected. This means that we can consider
that the network is still operational at the time the first node died, unless remaining
network is partitioned. Hence, MTE needs to be modified and extended as follows: For a
given network in which the edge cost is the transmission cost, every source nodes need
to find the shortest path to the sink and this path is used as a the minimum transmission
cost path for a node. When a certain node has drained its energy, by deleting this node
and all the edges incident to this node, we may get a reduced network. We can check
whether the reduced graph is strongly connected component or not. If the graph is still
a strongly connected component and that means the every source nodes must have
a connectivity to the sink. Thus, MTE can be applied on the reduced graph again. We
can apply this process over and over until the resulting network is no longer strongly
connected component. We call such a repeated MTE as a Sequential MTE (SMTE).

Note that above definition of the lifetime of the network in the SMTE is equivalent to the
time until network is partitioned. In fact, we do not consider the situation such that parts
of the sensing area become uncovered due to the outage of the sensor nodes.

2.3.3 Linear Programming Model

Let’s denote \( x_{ij} \), called a network flow, the rate of data from node \( i \) to node \( j \). To
transmit an unit bit of data from node \( i \) to node \( j \), \( e_{ij} \) amount of energy is used. The
lifespan of node \( i \) under flow \( x = \{x_{ij}\} \) is given by

\[
T_i(x) = \frac{E(i)}{\sum_{j \in N_i} e_{ij} t_{x_{ij}}} \tag{2–16}
\]

The network lifetime \( T \) under flow \( x \) is the time until the first node used up its
energy. That is,

\[
T(x) = \min_{i \in N} T_i(x) \tag{2–17}
\]
The flow that maximizes the network lifetime $T$ can be obtained by solving an following linear program.

$$(P_0) \quad \max \quad T$$

$$s.t. \quad \sum_{j \in N_i} x_{ij} - \sum_{j: i \in N_j} x_{ji} = \begin{cases} R(i) & \forall i \in N, i \neq s \\ -\sum_{k \in N, k \neq s} R(i) & \text{otherwise} \end{cases}$$  \hspace{1cm} (2–19)

$$T \sum_{j \in N_i} e_{ij} x_{ij} \leq E(i) \quad \forall i \in N$$  \hspace{1cm} (2–20)

$$x_{ij} \geq 0 \quad \forall (i, j) \in L \quad (2–21)$$

Note that this is the problem that maximizing the minimum lifetime of source nodes. The flow conservation in the network is represented as (2–19), and inequalities (2–20) mean the energy constraints on each node. By changing the variable $T$ to $1/z$, we can obtain a minimization problem but equivalent to the above maximization problem.

$$(P) \quad \min \quad z$$

$$s.t. \quad \sum_{j \in N_i} x_{ij} - \sum_{j: i \in N_j} x_{ji} = \begin{cases} R(i) & \forall i \in N, i \neq s \\ -\sum_{k \in N, k \neq s} R(i) & \text{otherwise} \end{cases}$$  \hspace{1cm} (2–23)

$$\sum_{j \in N_i} e_{ij} x_{ij} \leq zE(i) \quad \forall i \in N$$  \hspace{1cm} (2–24)

$$x_{ij} \geq 0 \quad \forall (i, j) \in L \quad (2–25)$$

Note that in this problem, we must determine $x = \{x_{ij}\}$ and $z$.

### 2.4 Experiments

#### 2.4.1 Graph Generation

We assume that the dimension of the sensor field is fixed as 50m x 50m and $n$ nodes are randomly created or deployed with a certain strategy in a sensor field. If there exists only one sink in the network, its position is also fixed to the exact center of the
area (For a 50x50 sensor field, sink's coordinate is (25.0, 25.0)). However in the case of multiple sinks, the locations of the sinks are randomly selected. For our simulation to be more realistic, we need to know a lot of variables and constant values, and we summarized them in the table 2-1. Note that we assume that energy required to run the transmitter/receiver circuitry in the sensor node is negligible, thus we just set $\alpha$ to 0.

<table>
<thead>
<tr>
<th>parameters</th>
<th>values</th>
</tr>
</thead>
<tbody>
<tr>
<td>communication range</td>
<td>${3.5m, 10.0m}$</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0</td>
</tr>
<tr>
<td>$\beta$</td>
<td>$1.3 \times 10^{-9} J/b/m^4$</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>${2, 3, 4}$</td>
</tr>
<tr>
<td>Initial Energy</td>
<td>500 J</td>
</tr>
<tr>
<td>Data Rate</td>
<td>500 bps</td>
</tr>
<tr>
<td>Dimension of Sensor Fields</td>
<td>${20 \times 20, 30 \times 30, 40 \times 40, 50 \times 50, 60 \times 60, 70 \times 70}$</td>
</tr>
<tr>
<td>The number of sinks</td>
<td>${1, 2, 3, \ldots, 60, 70, 80}$</td>
</tr>
<tr>
<td>The number of nodes</td>
<td>${200, 225, 250, 275, \ldots, 775, 800}$</td>
</tr>
<tr>
<td>Deployment methods</td>
<td>{uniform, grid-based}</td>
</tr>
</tbody>
</table>

Table 2-1. System parameters used in the simulation

We try to simulate the sensor network in various environments. For example, we want to figure out how does the lifetime change according to the different energy model, how can the number of the sink affect the lifetime of the sensor network and is there any noticeable relationship between the lifetime and the density of the sensor nodes, etc.

Figure 2-1 shows an example of graphs used in the simulation. In this graph, the location of node is determined in an uniformly distributed manner. If the distance between the node $i$ and the node $j$ is less than communication range, node $i$ and node $j$ have a bidirectional link. In our simulation, if the whole graph does not form a single strongly connected component, we just discard it and keep generating the new graph until we get a strongly connected one.

Figure 2-2 shows another graph which is generated by the grid-based node deployment. With this method, sensor field is divided into the fixed number of equal size grids and each grid contains at least one sensor in it. The maximum number of sensors in the grid is limited and determined by the number of nodes to be deployed.
Figure 2-1. Graph example 1: node deployment by the uniform distribution
and the number of grids so that too many nodes are not clumped in a single grid. The location of a node within the grid is determined randomly. With this method, we can get a graph in which nodes are well-scattered. In later section, we will explain the grid-based node deployment strategy more detail.

In this chapter, we mainly focus on two types of metrics as performance measures. One is about the network longevity, and the other is about the energy efficiency of the algorithm. Of course, the network longevity can be measured by the lifetime, and the energy efficiency can be represented by the ratio of the total remaining energy to the total initial energy after simulation is finished. When we say ratio throughout this chapter, it follows above definition of the ratio.

2.4.2 Lifetime vs. Ratio

Figure 2-3 shows how the ratio changes during the lifetime of sensor network. This is the result of the SMTE simulation and we assume that the communication range for a node is sufficiently large so that every node can reach the sink directly. In this figure, We can observe that the ratio is dropped sharply when the lifetime exceeds a certain point. We suspect lots of nodes in the sensor network died due to energy exhaustion after the lifetime exceeds the threshold point, thus the distance to traverse in a single hop might increase. The more nodes die, the faster energy is used. In fact, we might not get the same result for all the nodes as the figure 2-3, for example for the node near the sink, this sudden dropping of the ratio might come at earlier time. On the other hand, for the node farther from the sink, this dropping happens at the later time, but it is very steep.

2.4.3 Effect Of Node Densities

When the dimension of a sensor field is fixed, we can control the density of the nodes by changing the number of nodes to be deployed. Figure 2-4A shows the performance of the simulation under various number of nodes. In this graph, MTE, LP, and SMTE means the Minimum Transmission Energy simulation, Linear Program solution, and Sequential MTE, respectively. In our experiments, the number of nodes
Figure 2-2. Graph example 2: node deployment by grid based strategy
are increased by 25 from 200 to 800 and we run 100 times for each configuration. We assume that there is only one sink at the very center of the sensor field and we set the $\gamma$ of the energy model to 4. All three simulations show that lifetime increase with steady paces as the number of nodes increase. Note that the lifetimes of the SMTE and LP are almost the same. We observe the decrease in the ratio for the LP case, as the node density increases. However, in cases of MTE and SMTE, the ratio seems to be constant without regard to the increase of node density. Hence, we conclude that for the MTE, the increase of the node density may not contribute to the lifetime that much. As the node density increases, average distance between nodes get shorter, thus each node may get energy saving due to the shorter distance hop. However, increased number of nodes might also put the relaying stress on the nodes and this stress might be stronger to the nodes closer to the sink than the nodes farther from the sink. This relaying burden cuts down the benefit of the shortened average distance between nodes. For the LP cases, the decreased ratio may account for the increased lifetime of the network. SMTE is very
scheme that may make full advantage of the increase of node density. The lifetime for SMTE is much longer that MTE and its energy efficiency is better than LP.

2.4.4 Effect Of The Number Of Sinks

In this section, we focus on how the number of sinks would affect the performance of the sensor networks. When the number of the sink is 1, its position is fixed at the center of the sensor field. However, if there are several sinks in the network, their locations are randomly selected. Furthermore, in the multiple sink system, every sensor nodes should select only one of the sinks as a destination of the data depending on the routing algorithm. Thus, if the MTE used, node may select the sink which it can reach with the smallest energy. We observe the performance of the network with fixed number
of nodes (400 nodes) which are deployed by the uniform distribution and the number of sinks are increased from 1 to 60. In general, as the number of sinks is increased, average length of the paths and average number of hops becomes smaller. In addition, traffic relay load of the nodes is perhaps reduced. The result is shown in Figure 2-5A and figure 2-5B. As we see in figure 2-5A, the number of sinks becomes larger, SMTE

![A Lifetime (uniform distribution)](image1)

![B Ratio (uniform distribution)](image2)

![C Lifetime (grid based deployment)](image3)

![D Ratio (grid based deployment)](image4)

Figure 2-5. Performance in various number of sinks

outperforms the LP. Ratios of all three routing cases tends to be constant or slightly increasing after a certain number of sinks. Interestingly, we find that it is more effective to increase the number of the sinks rather than to increase the node density for the network lifetime. Note that, when the node density increases node has shorter links with
its neighbor. On the other hand, node may have shorter length paths to the sink when the number of sinks increases.

Figure 2-5C and figure 2-5D show the performance under grid-based node deployment scheme.

Although the performance in the grid-based deployment is slightly better than the performance in the uniform deployment in all the simulation, we can not find a significant difference in term of the performance between both deployment strategies as the number of the sink increases.

2.4.5 Effect Of Node Densities And The Number Of Sinks

Figure 2-6. Performance in different number of nodes and different number of sinks

Figure 2-6 shows the performance of the single sink network and multiple-sink networks (In the multiple sinks case, we set the number of the sink to 4) under various number of nodes. Obviously, the system of multiple sinks outperforms single sink system. In both cases, the lifetimes of MTE are very poor and the lifetime of LP is slightly superior to the that of SMTE, but its difference is negligible.

The rate of increase in lifetime for the multiple sink case is greater than the rate for the single sink system. This fact gives a hint that putting more sinks is beneficial to the lifetime when large number of sensors are deployed. Probably, the study of the impact of the multiple sinks system is a good research issue.
2.4.6 Effect Of Node Deployment Strategies

In this section, we consider the two types of node deployment strategies. One is a uniformly distributed node deployment (or uniform distribution) and the other is a grid-based node deployment (or grid based distribution). When a uniformly distributed node deployment is used, the coordinate of the node $i$, say $(x_i, y_i)$, is determined by uniform distribution. In other words, we suppose that $x_i$ and $y_i$, where $i = 1 \ldots n$, are identically independent random variables with uniform distribution in the interval $(0, 50)$. Note that our sensor field is fixed to 50x50 through this section. For the grid-based node deployment, we are given the number of grids as a parameter, denoted by $\Delta$. Then the average number of nodes in each grid must be $\rho = n/\Delta$. For the simplicity, we assume $\rho \geq 1$. To satisfy this condition, in each grid, we need to determine the total number of nodes and this number follows uniform distribution in $[1..2\rho - 1]$. Since, there exit average $\rho$ nodes in the grid and we have $\Delta = n/\rho$ grids in the system, on average we have $n$ nodes in the system. This is the key to the grid-based node deployment simulation.

Figure 2-7 shows the performances of the uniform node deployment method and grid-based node deployment method in the both single and multiple sink system. We observe, in the perspective of the performance, there is no significant difference between both deployment strategies, as the number of the nodes increases. In multiple sinks situation, we also observe the same phenomenon as the single sinks setting. However, the lifetime in the multiple sinks system is far better than that of the single sink system. Finally, we conclude that the lifetime has nothing to do with the node deployment strategy, especially for the dense sensor network.

2.4.7 Effect Of Communication Ranges

In this section, we focus on how different communication ranges affect the performance of the sensor networks. In general, communication range determines the neighbor set $N_i$ for the node $i$. If we draw a circle at node $i$ with the radius that is
equal to the communication range, the neighbor set $N_i$ of node $i$ is the set of all nodes reside within this circle. Thus, larger communication range means the bigger neighbor set. In addition, neighbor sets with different communication ranges have an inclusion property, that is, the neighbor set with a larger communication range must include the neighbor set with a smaller communication range.

Figure 2-8 shows the lifetimes and ratios of the simulations of various communication ranges. In these simulations, the number of nodes is fixed to 400 and the communication range varies from 3.5 meters to 10 meters. In each simulation, we use the same network topology for the fairness. Note that the lifetimes and ratios remain constant for the case of MTE without regard to the changes of the communication range. In the MTE routing,
each node prefers the shortest cost hop among its neighbor set. Due to the inclusion property, this shortest cost hop neighbor is included in the bigger neighbor set with a larger communication range. Thus, an enlarged neighbor set due to the increased communication range does not influence the selection of the next hop during routing. However, too short communication range perhaps destroys the network connectivity. In conclusion, as long as the connectivity of the network is guaranteed, the communication range does not have an effect on the performance of the network whose routing scheme is MTE.

### 2.5 Experiments on the Minimum Transmission Energy Routing, Sequential Minimum Transmission Energy Routing, and Optimal Lifetime Routing

For SMTE and LP cases, we observe that their lifetimes increase as the communication range gets longer, but they seem to converge to a particular value. However, the quantitative properties of this values are known yet. While the ratio of the LP converges to the 0, the ratio of the SMTE is still high. We are not sure whether SMTE outperforms LP or not, if we keep simulating the both two routing scheme beyond 10.

In this section, we study the effects of the different size of sensor fields on the performance of the sensor networks. In this simulation, we do not fix the number of nodes to a particular value. Instead, we set the node density to a constant value.
Since the node density is fixed and the dimension of the sensor fields varies, the number of nodes also varies according to the area of the sensor fields and the constant node density. We assume all sensor fields shape square, and we just change side of the sensor fields from 20 to 70. Therefore the areas of the sensor fields are 400, 900, 1600, and so on. The node density used in the simulation is set to 0.16. With these parameters, we can find the number of nodes in different sized sensor fields: 64, 144, 256, and so on. The other parameters are set to their default. Please refer to the table 2-1. With the same node density, larger sensor fields means the distance between nodes is the same without respect to the size of the sensor fields, but the paths have more hops in the larger sensor field.

Figure 2-9A shows the lifetime of the simulation. As the dimension of the sensor fields gets larger, the lifetime of the network gets smaller. This may be due to the longer paths (with more hops) as well as enlarged relaying load. On the other hand, the ratio of simulation keep increasing according to the size of sensor fields due to the decreasing lifetime.

2.6 Summary

In this chapter, we have reviewed the past research works on the sensor networks which basically aimed to prolong the lifetime, and we have presented the results of
extensive simulations performed on the sensor networks. As our intuition, we verified that lots of factors such as the number of sink nodes or nodes to be deployed, and the communication ranges have an influence on the lifetime of the sensor networks through simulations. However, on the contrary to the intuition, we found that there is no or at most little relationship between the lifetime and deployment strategies in the large sensor networks. In addition, we measured the lifetime of SMTE, an extended version of MTE, and discovered the lifetime of SMTE is almost equivalent to the lifetime of the optimal value that is calculated by Linear Program method.

In our observation, we found that the most effective way to increase the lifetime of the networks is to put more sink nodes into the sensor networks. The study on the performance of the sensor networks with multiple sinks might be one of our future research.
CHAPTER 3
A METHOD FOR DECIDING NODE DENSITY IN NON-UNIFORM DEPLOYMENT OF WIRELESS SENSOR NETWORKS

3.1 Overview

Wireless sensor networks have diverse applications such as environmental monitoring (e.g., vehicular traffic, wild life habitat, bridge or earthquake monitoring) and battlefield surveillance. A sensor network is in general a self-organized infrastructure that uses multi-hop routing to deliver the collected information to some collection center, or the sink. The sensor nodes in the network typically face severe energy, computation and communication constraints. They usually have limited on-board batteries and are often deployed in harsh environment where human operators cannot access them easily, making it difficult or impossible to replace the batteries. As a result, much of the research on sensor networks has focused on the longevity, or lifetime, of the network.

The lifetime of a sensor network has several definitions in the literature. One of the most popular definitions is the interval from the time when the system starts operation until the time when the first node exhausts its energy. This is equal to the shortest lifetime of the nodes. Much work has been published on how to prolong the lifetime of sensor networks [12] [11] [17] [19] [35] [45]. Some proposed energy efficient routing strategies, whereas others proposed efficient ways of clustering the nodes to save energy.

Regardless of the energy-saving strategies used, sensor networks often experience unbalanced traffic distribution because the multi-hop traffic pattern is typically many-to-one [24, 27, 30, 32, 34]. Sensor nodes in the network act as data originators and data relayers. The traffic transmitted by each node typically includes both self-generated and relayed traffic. Since the entire network traffic flows toward the sink, the nodes closer to the sink tend to experience more traffic. As a result, their energy consumption rates tend to be higher than those nodes that are far away from the sink, assuming the transmission distance is the same. This causes the nodes closer to the sink to die
earlier, leaving a hole near the sink and partitioning the whole network while many remaining nodes still have a plenty of energy. This phenomenon is called the energy hole problem \cite{23, 24, 43}. In \cite{30}, the authors claimed that when the nodes one hop away from the sink use up all their energy, the remaining nodes have used only 7\% of their energy on average. It has been shown analytically in \cite{23, 24} that the energy hole problem exists in various sensor networks.

On closer examination, the energy hole is most often observed in networks where the sensor nodes are homogeneous and uniformly deployed, and they report events generated at a constant rate to the sink. If we allow non-uniform deployment of the sensor nodes, by carefully increasing the number of nodes around the sink, we can prevent the sensor nodes near the sink from depleting their energy faster than others, and hence, resolve the energy hole problem. Adding more nodes in the sensor field also has other benefits, such as better connectivity and higher reliability. On the other hand, adding more nodes means a higher cost. Hence, this solution makes sense in situations where inexpensive sensors can be mass-produced or having a longer network lifetime outweighs the cost of the extra sensors. Recent advances in micro-electro-mechanical and integration technologies make the first situation more and more likely to occur.

This chapter contributes to the research area that seeks to extend network lifetime by deploying the sensors non-uniformly and by carefully controlling the node densities in different parts of the sensor field. The main question to be addressed is how many nodes per unit area (i.e., the node density) should be deployed at each location in order to achieve a prescribed lifetime-cost objective. The main result of the chapter is a mathematical method for computing the node densities.

This chapter illustrates the method using a particular example, in fact, a particular objective. We show how to derive the location-dependent node densities that equalize the energy dissipation rate of the sensor nodes throughout the network. The result is
that all nodes will exhaust their energy at the same time, and hence, energy holes will not emerge.

Although the illustration of the method uses an example, the method itself is intended to be general. It can be adapted for other lifetime-cost objectives and other constraints, including the routing strategy. For instance, if the cost of deploying extra sensors is not negligible, one can incorporate a cost function of the node densities as part of the objective and compute the required deployment densities.

The method requires a model for the underlying routing strategy. We consider several routing models, which are meant to capture the essence of the underlying routing protocols. For tractability, the routing models are necessarily simple and may not follow precisely the routing protocols. Some other issues are also overlooked in this chapter, such as the algorithms and protocols for determining which nodes become active or inactive in each region. These issues are either orthogonal to our work or left to future studies.

Several researchers have studied the energy hole problem and the uneven traffic distribution problem [19, 21, 23, 24, 30, 32, 43]. Among them, [43] is the most similar to our work in its goal of obtaining a balanced energy dissipation rate everywhere by non-uniform node deployment. The authors of [43] proved that an uneven energy consumption rate is unavoidable if all nodes are homogeneous and are deployed uniformly in the network. In their model, the sensor field is divided into several concentric coronas or rings around the sink. They gave a heuristic routing scheme that achieves an equal energy dissipation rate in all rings except the outmost one, provided the number of nodes increases geometrically from the outmost ring inward. However, their sensor field, energy consumption, and routing models are significantly different from ours.

1. The nodes in [43] can send data only to the nodes in the neighboring ring. However, even in our simpler model, nodes can send data to different inner rings with different probabilities. We then make further generalization so that the
sensor field can have an arbitrary two-dimensional shape rather than a disk and we allow data transmission from any node to any other node.

2. The nodes in [43] generate data at the same constant rate. However, the nodes in our models may have different event generation rates.

3. Our energy consumption model is more general: The required transmission power of a node is a function of the transmission distance to the receiver.

The resulting problem of determining the node densities is quite different and more complicated in our case. Furthermore, we can achieve an equal energy dissipation rate for all nodes in the entire network.

In [30], it is assumed that the sensor nodes are deployed uniformly. The sensor field is also decomposed into concentric rings around the sink. The assumption on routing is that each node can only transmit data to a node in the inner adjacent ring. The main question addressed by [30] is how to decide the widths of the rings so that all nodes in the network exhaust their energy at the same time.

In [32], the authors provided a formal description of the problem that traffic tends to be concentrated at the nodes close to the sink when the shortest path routing is used. They suggested a heuristic algorithm that finds some “curved” paths to the sink and showed that the traffic load is more balanced.

In [6, 42], density control is used as one of the means to guarantee the coverage requirement of the sensor network rather than to provide a balanced energy dissipation rate of the nodes. In [21], the authors studied the impact of carefully controlled deployment of the sensor nodes and the sink on the data capacity, which is defined as the total amount of data that can reach the sink. They proposed and analyzed several approaches to increase the data capacity, and showed that non-uniform deployment can outperform uniform deployment.

Many authors formulate the problem of maximizing the network lifetime as optimization problems [12], [26], [35], [45]. In [35] and [45], the authors proposed approximation algorithms to solve the multicommodity flow problem induced by the
problem of sensor network lifetime maximization. The authors of [17] calculated upper bounds on the lifetime of the networks that have regular topologies, such as a regular linear array and a regular two-dimensional circular network.

In [19], the authors proposed a dynamic node-clustering scheme known as LEACH, where each sensor node may operate as a cluster head depending on its remaining energy and the cluster heads change during the operation of the network. They compared the lifetime resulting from LEACH with the MTE (minimum transmission energy) routing in which each node uses the path that consumes the least amount of total energy among all possible paths.

Some researchers define the lifetime of sensor networks differently from the time until the first node dies [47], [22]. The authors of [47] introduced the $\alpha$-lifetime, which is the time until the remaining sensor nodes can still cover at least $\alpha$ portion of the entire sensor field. In [22], the authors were interested in prolonging the network operation time after the first node dies. They recursively maximize the $n$-th minimum lifetime of the nodes after $(n-1)$-th minimum lifetime of the nodes has been maximized, for all $n$.

### 3.2 Models with Discrete Ring Structure

In this section, we show how to compute the node densities required to equalize the energy consumption rates of all nodes in the network. We make some simplifications on the network model to illustrate the basic ideas of the method and to make numerical computation easier. The method can be applied to other objectives and network models (see Section 3.4).

For this section, we assume the shape of the sensor field is a disk and the sink is at the center of the disk (Figure 3-1). The disk is divided into concentric rings having the same width and the final node density in each ring will be constant. We consider the routing rule at the granularity of rings. That is, all nodes in a ring are subject to an identical routing rule, which specifies the next-hop ring rather than the next-hop node. The ring-based modeling approach here is typical (See [30], [43]). As a result, we
only need to compute a finite number of node densities, one for each ring. In spite of its simplicity, the model is still considerably more general than those in [30], [43] (See the discussion in Section 3.1.), and is already useful for a number of practical cases. Later in Section 3.4, we will make generalization that allows sensor fields of arbitrary two-dimensional shapes, routing between two arbitrary locations in the field, and node density as a function of the precise location.

3.2.1 Sensor Field and Energy Consumption Models

The sensor field in the shape of a disk is shown in Figure 3-1 with the sink at the center. The communication capability of the nodes is limited so that multi-hop routing is necessary to deliver the data to the sink. We introduce some definitions and notations.

- \( n \): the total number of rings.
- \( R_i \): ring \( i \), \( 0 \leq i \leq n \). We index the rings in the direction away from the center of the disk. For convenience, ring 0, \( R_0 \), refers to the center of the disk where the sink is. Ring 1 is also special. It is a small disk.
- \( w \): the width of each ring, \( R_1, \ldots, R_n \).
- \( A_i \): the area of the ring \( i \), \( 1 \leq i \leq n \). \( A_i = \pi (2i - 1)w^2 \).
- \( \rho_i \): the node density of ring \( i \), \( 1 \leq i \leq n \).
- \( N_i \): the number of the nodes in the ring \( i \), \( 1 \leq i \leq n \). \( N_i = \rho_i A_i = \pi (2i - 1)w^2 \rho_i \).

The energy consumption/dissipation model of the sensor nodes affects the final density of each ring. We adopt the energy dissipation model of [19]. The required transmission energy to send one unit of data to a node at a distance \( d \) away from the sender is given by

\[
E_t(d) = \gamma + \beta d^\alpha, 
\]

where \( \gamma \) is the required energy to operate the transceiver circuitry, \( \beta \) is a parameter characterizing the transceiver amplifiers energy consumption, and \( \alpha \) is the so-called path loss exponent. Normally, \( \alpha \) is between 2 to 6 depending on the operating
environment. Some amount of energy is also required to receive a unit of data. The energy consumed at a receiver is as follows.

\[ E_r(\cdot) = \gamma, \]  

(3–2) where \( \gamma \) is the same as in (3–1). Normally, the receiving energy requirement does not depend on the distance from the transmitter. For ease of presentation, in the analysis to be shown later, we sometimes ignore the receiving energy.

The maximum transmission range of a sensor node is also an important parameter. In this section, we assume this range is \( l \) rings, \( 1 \leq l \leq n \). That is, a sensor node can transmit data up to \( l \) rings away without relaying. We call this maximum range the *maximum jump*. If the maximum jump is equal to \( n \), then every sensor node is able to send data to the sink directly, which is the assumption of [30].

The routing strategy has major impact on the final densities. We will consider several routing strategies later.
3.2.2 Deriving the Node Densities of the Rings

Our goal is to make the energy consumption rate of every node equal by controlling the node densities in the sensor field. Since we assume that each ring has a constant node density, we need to ask what the node density should be in each ring so that a typical node in one ring consumes the same amount of energy per unit of time as a typical node in any other ring.

Suppose the nodes are uniformly distributed over ring \( j \) with a density \( \rho_j, 1 \leq j \leq n \). Consider an arbitrary node in ring \( j \). Since a typical node in the sensor network both generates traffic as a data source and relays traffic for other nodes, the node’s total data transmission rate is the sum of the rate of the locally-generated traffic and the rate of the traffic it relays. We assume flow conservation at every sensor node: A node cannot buffer an infinite amount of data, and, after the traffic is generated, there is no further in-network processing that may reduce or increase the traffic volume at the node. The data transmission rate at a node in ring \( j \) can be expressed as

\[
G_j = S_j + C_j, \tag{3–3}
\]

where \( S_j \) is the rate of the locally-generated data, and \( C_j \) is the rate of the traffic to be relayed by the node.

Let us assume that certain amount of data rate is needed to monitor a unit of area and this rate is a constant value of \( K \) throughout the sensor field. This is the inherent data rate needed for reporting events or conditions about a given area. There are at least two possibilities regarding how this inherent data rate affects the actual traffic rate generated by each nearby node. In the first, one can assume that the system has local coordination among the nearby sensors that reduces the amount of traffic generated. In the best case, every nearby node generates the minimum amount of traffic sufficient to cover the area. That is, for ring \( j \), the rate of the locally-generated traffic at a typical node
is

\[ S_j = \frac{K}{\rho_j}. \]  \hfill (3–4)

The second possibility is that there is no local coordination among the nodes and the nearby nodes all report the same events to the sink. In this case, the traffic generated by any node in the network is \( K \). Between the two extreme cases are a large number of other possibilities, depending on the degree of local coordination and other factors. Note that the local coordination can take the form of sleep-wake schedules, in which only a subset of the nodes are awake at any moment. Or, the rate of report generation at each node is made inversely proportional to the node density nearby. In either case, one can expect that the locally generated traffic rate at a node is proportional to \( \frac{K}{\rho_j} \), i.e., \( S_j = \eta \frac{K}{\rho_j} \) for some constant \( \eta > 0 \). In this section, we assume \( \eta = 1 \) for notational simplicity. In effect, we assume the first possibility (3–4) as the local traffic generation model. But, extension to the case with a general constant \( \eta \) is trivial. For other more different models, the entire methodology in the section still applies, although the results will differ more substantially.

The rate of the relay traffic at a node does not have a simple expression. It is easier to write out the recursive relationship it satisfies. For each pair of rings \( k \) and \( j \), \( 0 \leq j < k \), let

\[ F_k(j) = \{ \text{the probability that a node in ring } k \text{ selects} \]
\[ \quad \text{a node in ring } j \text{ as its next-hop neighbor} \}. \hfill (3–5) \]

Note that all nodes in the same ring have the same probability distribution. The behavior of different routing schemes can be captured by different choices of \( F_k(j) \). Hence, we call each particular matrix \( (F_k(j)) \) a \textit{routing model}.

\[ N_k G_k F_k(j) = \rho_k A_k G_k F_k(j) = \rho_k \pi (2k - 1) w^2 G_k F_k(j). \]
Therefore, the average rate of the traffic contributed by ring $k$ to a typical node in ring $j$ is
\[
\frac{\rho_k \pi (2k - 1) w^2 G_k F_k(j)}{\rho_j \pi (2j - 1) w^2} = \frac{\rho_k (2k - 1) G_k F_k(j)}{\rho_j (2j - 1)}.
\]

For each fixed $j$, $1 \leq j < n$, the rate of the relay traffic at a typical node in ring $j$, $C_j$, is the sum of the above quantity over all rings outside ring $j$ that can reach ring $j$. That is,
\[
C_j = \frac{\sum_{k=j+1}^{\min(n,j+l)} \rho_k (2k - 1) G_k F_k(j)}{\rho_j (2j - 1)}.
\]

(3–6)

Recall that, in $\min(n, j + l)$, ring $n$ is the outmost ring and $l$ is the maximum jump.

Now we can get the total data transmission rate for a node in ring $j$, $1 \leq j < n$, by summing (3–4) and (3–6).
\[
G_j = \frac{K (2j - 1) + \sum_{k=j+1}^{\min(n,j+l)} \rho_k (2k - 1) G_k F_k(j)}{\rho_j (2j - 1)}.
\]

(3–7)

For the outmost ring $n$, since $C_n = 0$ and $S_n = K/\rho_n$, we have
\[
G_n = K/\rho_n.
\]

(3–8)

From (3–7), we can write, for all $1 \leq j < n$,
\[
\rho_j = \frac{K}{G_j} + \sum_{k=j+1}^{\min(n,j+l)} \frac{(2k - 1) G_k F_k(j)}{(2j - 1) G_j} \rho_k.
\]

(3–9)

From (3–9), note that, if $G_k$ is known for all $k$, $\rho_n$ is known, and $F_k(j)$ is independent on $\rho$ for all $k$ and $j$, then, $\rho_j$ can be computed iteratively for $j$ from $n - 1$ down to 1.

We next show how $G_k$ can be determined. Consider the energy consumption rate for a typical node in ring $j$, denoted by $P_j$. $P_j$ depends on the energy consumption model discussed in Section 3.2.1. For notational simplicity, we set $\gamma = 0$, which means that we ignore the energy required for operating the transceiver circuitry and the energy required
for receiving data. The subsequent development still applies to the general case of \( \gamma \neq 0 \); but the expressions are more complicated.

In our energy model, the energy required to transmit a unit of data is a function of the transmission distance. To simplify the analysis, we make the approximation that the distance between a pair of nodes is determined by the number of rings separating them. The energy that a node in ring \( j, 1 \leq j \leq n \), consumes per unit of time to transfer the portion of its data directed to the nodes in ring \( i, 0 \leq i < j \), is

\[
P_j(i) = (j - i)^\alpha w^\alpha G_j F_j(i).
\]

Thus, the energy consumption rate for a node in ring \( j, 1 \leq j \leq n \), is

\[
P_j = \sum_{i=(j-1)_+}^{j-1} P_j(i) = w^\alpha G_j \sum_{i=(j-1)_+}^{j-1} (j - i)^\alpha F_j(i).
\]

\[\text{(3–10)}\]

### 3.2.2.1 Case of density-independent routing

Our goal is to equalize the energy consumption rate of all nodes. In other words, for all \( 1 \leq j < n \), we want to make \( P_j = P_n \). From (3–10), we get the following relationship for all \( j, 1 \leq j < n \).

---

1. Here, the probability \( F_j(i) \) is interpreted as the portion of data at a fixed node in ring \( j \) that is transmitted to some node in ring \( i \). Throughout, we will use the two interpretations (probability or proportion) interchangeably depending on convenience.

2. We use the notation \((a)_+ = \max(a, 0)\).
\[ G_j = \frac{\sum_{i=(n-l)_+}^{n-1} (n-i)^\alpha F_n(i)}{\sum_{i=(j-l)_+}^{j-1} (j-i)^\alpha F_j(i)} G_n. \]  

(3–11)

Note that \( G_n \) is given by (3–8), which depends only on \( \rho_n \). Throughout, \( \rho_n \) can be considered as a given parameter. Hence, if \( F_k(j) \) has no dependency on any \( \rho_k \), one can compute \( G_j \) using (3–11) for all \( j \). The resulting \( G_j \) is parameterized by \( \rho_n \). After that, one can compute the densities \( \rho_j \) iteratively using (3–9) for all \( j \) from \( n-1 \) down to 1.

3.2.2.2 Case of density-dependent routing

The situation becomes more complicated if \( F_k(j) \) depends on some \( \rho_k \). In that case, \( G_j \) in (3–11) depends on the unknown \( \rho_k \). One can still start with (3–9) and eliminate all \( G_j \)'s from (3–9) by using (3–8) and (3–11). First, note that

\[ \frac{G_k}{G_j} = \frac{\sum_{i=(j-l)_+}^{j-1} (j-i)^\alpha F_j(i)}{\sum_{i=(k-l)_+}^{k-1} (k-i)^\alpha F_k(i)}. \]  

(3–12)

Then, (3–9) can be re-written as follows.

\[
\rho_j = \rho_n \frac{\sum_{i=(j-l)_+}^{j-1} (j-i)^\alpha F_j(i)}{\sum_{i=(n-l)_+}^{n-1} (n-i)^\alpha F_n(i)} + \sum_{k=j+1}^{\min(n,j+l)} \frac{\sum_{i=(j-l)_+}^{j-1} (j-i)^\alpha F_j(i)}{\sum_{i=(k-l)_+}^{k-1} (k-i)^\alpha F_k(i)} \frac{(2k-1)}{(2j-1)} \times F_k(j) \rho_k.
\]  

(3–13)

The above expression does not imply that, if \( \rho_n \) is given, then one can compute the densities \( \rho_j \) for all other \( j \). This is because \( F_k(j) \) may depend on the densities in complicated ways. The set of equations in (3–13), for \( 1 \leq j < n \), is a fairly complex
system of nonlinear equations with $\rho_j$ as the variables. However, (3–13) does suggest a different kind of iterative method to compute each $\rho_j$, which will be called *successive substitution*. Suppose, we initialize the iteration at some constant $\rho_j^{(0)}$ for all $1 \leq j < n$. This gives $F_k^{(0)}(j)$, for different $k$ and $j$. Then, one can substitute $\rho_j^{(0)}$ and $F_k^{(0)}(j)$ into the right hand side of (3–13) and derive $\rho_j^{(1)}$ for all $1 \leq j < n$. In a general iteration step $t$, the following iteration occurs, for $1 \leq j < n$.

$$
\rho_j^{(t+1)} = \rho_n \sum_{i=(j-l)}^{j-1} \frac{(j-i)^\alpha F_j^{(t)}(i)}{\sum_{i=(n-l)}^{n-1} (n-i)^\alpha F_n^{(t)}(i)} + \\
\sum_{k=j+1}^{\min(n,j+l)} \frac{(2k-1) (2j-1)}{(k-l)^\alpha F_k^{(t)}(j) (k-i)^\alpha F_k^{(t)}(i)}
$$

\[ (3–14) \]

where, for a fixed $t$, $(F_k^{(t)}(j))$ is computed using $(\rho_j^{(t)})$. The process can be continued until $\rho_j^{(t)}$ converges, for $1 \leq j < n$. We can take values in the limit as the solutions of the equations in (3–13). We will show by experiments in Section 3.3 that this procedure indeed works.

Note that the constant $K$ does not show up in (3–13). Hence, for the purpose of computing the node densities, we can set $K = 1$ without affecting the solution, provided $\rho_n$ is given. Suppose $(F_k(j))$ is independent of the scaling of the node density functions. That is, $F_k(j)$ remains unchanged for all $k$ and $j$ when $\rho_i$ is scaled by a constant factor $\kappa > 0$ for all $i, 1 \leq i \leq n$. It can be observed that if $(\rho_j)_{j=1}^{n-1}$ is a solution to (3–13) given $\rho_n$, then $(\kappa \rho_j)_{j=1}^{n-1}$ is a solution to (3–13) given $\kappa \rho_n$. In this case, one can decide the node densities as follows: Choose an arbitrary positive value for $\rho_n$; compute all $\rho_j$ for $1 \leq j < n$; and find a suitable constant $\kappa$ and use $(\kappa \rho_j)_{j=1}^{n-1}$ as the node densities for deployment so that every ring has sufficient nodes to satisfy the monitoring need.

**3.2.3 Models of Routing/Node Selection**

We now have general equations that the node densities must satisfy to achieve an equal energy dissipation rate for all nodes under a generic routing model $(F_k(j))$. For
any routing strategy used in practice, if one can cast it into a specification in terms of 
$F_k(j)$, then one can use (3–8), (3–11) and (3–9) (or, equivalently, (3–13)) to compute 
the required node densities. Next, we will consider some simple routing schemes as 
examples and later show numerical results about them. Many more routing schemes 
can be modeled similarly.

3.2.3.1 Uniform ring selection

With this scheme, a node finds its next-hop node in the direction to the sink via two 
steps. First, the node selects a reachable ring with a uniform probability distribution. 
Second, the sending node randomly chooses the next-hop node among the nodes in the 
intersection of the selected ring and the sender’s communication range. Therefore, the 
probability that ring $i$, $(j - l)_+ \leq i < j$, is selected as the next-hop ring by a node in ring $j$ 
is

$$F_j(i) = 1 / \min(l, j). \quad (3–15)$$

Note that, in this case, $F_j(i)$ is independent on the node densities. From (3–11), $G_j$ 
becomes

$$G_j = \frac{\sum_{i=(j-l)_+}^{n-1} \frac{(n-i)^{\alpha}}{\min(n,l)}}{\sum_{i=(j-l)_+}^{n-1} \frac{(j-i)^{\alpha}}{\min(j,l)}} G_n. \quad (3–16)$$

$G_n$ is the same as before, equal to $K/\rho_n$. From (3–13), the node densities can be 
computed iteratively from $n – 1$ to 1 by the following expression.

$$\rho_j = \rho_n \frac{\min(n, l)}{\min(j, l)} \left( \frac{\sum_{i=(j-l)_+}^{j-1} (j-i)^{\alpha}}{\sum_{i=(n-l)_+}^{n-1} (n-i)^{\alpha}} + \right.$$ 
$$\left. \sum_{\min(n,j+l)}^{\min(n,j+l)} \frac{\sum_{i=(j-l)_+}^{j-1} (j-i)^{\alpha}}{\sum_{i=(k-l)_+}^{k-1} (k-i)^{\alpha}} \frac{1}{\min(j,l)} \rho_k. \right) \quad (3–17)$$
3.2.3.2 Uniform node selection

In this routing model, a node \( X \) can select any node with the same probability as long as the target node resides in \( X \)'s communication range and is closer to the sink than \( X \) is. This scheme is motivated by geographical routing.

![Uniform node selection diagram](image)

Figure 3-2. Uniform node selection

Figure 3-2 illustrates the underlying geometry for Uniform Node Selection. Node \( X \) can choose any node in the shaded part in the figure as the next-hop node. Let the region within the communication range of node \( X \) be denoted by \( Q \). As shown in Figure 3-2, node \( X \) can choose a next-hop node in the intersection of \( Q \) and the inner rings that \( X \) can reach, i.e., \( \cup_{k=(j-l)+}^{j-1} (R_k \cap Q) \). When the nodes in ring \( k \) are uniformly distributed, the number of nodes in \( R_k \cap Q \) is \( \rho_k \lambda(R_k \cap Q) \), where the notation \( \lambda(S) \) represents the area of a region \( S \). The number of possible next-hop nodes for node \( X \) is \( \sum_{k=(j-l)+}^{j-1} \rho_k \lambda(R_k \cap Q) \). Hence, the probability \( F_j(i) \) is, for \( (j-l)_+ \leq i < j \),

\[
F_j(i) = \frac{\rho_i \lambda(R_i \cap Q)}{\sum_{k=(j-l)_+}^{j-1} \rho_k \lambda(R_k \cap Q)}. \quad (3-18)
\]
From basic knowledge of geometry, we can find the area of the intersection of two disks. If the distance between the centers of two disks of radii \( r \) and \( R \), respectively, is \( d \), the area of the intersection is given by

\[
\Lambda(r,R,d) = r^2 \cos^{-1}\left(\frac{d^2 + r^2 - R^2}{2dr}\right) + R^2 \cos^{-1}\left(\frac{d^2 + R^2 - r^2}{2dR}\right) - \frac{1}{2} \sqrt{(-d + r + R)(d + r - R)(d - r + R)(d + r + R)}.
\]

Now, we can find the area of \( R_i \cap Q \). Let \( d \) be the distance of node \( X \) from the sink and note that \( lw \) is the communication range of node \( X \). The area of \( R_i \cap Q \), for \((j-l)_+ < i < j\), is obtained by,

\[
\lambda(Q \cap R_i) = \Lambda(lw, iw, d) - \Lambda(lw, (i-1)w, d).
\] (3–19)

The area of \( R_{(j-l)_+} \cap Q \) is given by

\[
\lambda(R_{(j-l)_+} \cap Q) = \Lambda(lw, w(j-l)_+, d).
\] (3–20)

The probability \( F_j(i) \) can be obtained by plugging equations (3–19) and (3–20) into (3–18). By applying the expressions for \( F_j(i) \) to (3–13), we derive a set of equations in \( \rho_j \) only, for different \( j \). From (3–18), note that \( F_j(i) \) depends on various \( \rho_i \). Hence, the resulting equations cannot be solved by deriving each \( \rho_i \) for \( i = n - 1 \) down to 1 iteratively using (3–13). But, they may be solved by successive substitution, which is to iterate \( (\rho_j^{(t)}) \) over \( t \) as in (3–14).

Note also that, previously, we have assumed that the distance between node \( X \) and its next-hop node is determined by the rings in which the two nodes lie. This assumption is not accurate in the current model. The distance to the next-hop node depends on where the next-hop node lies in its ring. We will later describe an extended formulation that incorporates the accurate distances between nodes, but will do so in a much more general setting with respect to other aspects as well (Section 3.4). The price to pay is
higher theoretical and computational complexity. For now, we ignore this inaccuracy for the benefit of simpler numerical computation.

### 3.2.3.3 Simplified uniform node selection

In this routing model, the probability that a node in ring \( j \) takes a node in ring \( i \) as the next-hop node, \( F_j(i) \), is proportional to the number of nodes in ring \( i \), where \( (j - l)_+ < i < j \). Thus, the probability can be written as follows.

\[
F_j(i) = \frac{\rho_i(2i - 1)}{\sum_{k=\max(j-l,1)}^{j-1} \rho_k(2k - 1)}, \quad (j - l)_+ < i < j. \tag{3-21}
\]

Since \( F_j(i) \) depends on \( (\rho_j) \), successive substitution of the form in (3–14) are needed to find the solution.

The following reasoning shows why this model can be viewed as a simplification of the Uniform Node Selection scheme. Suppose the range of each node is \( l_r \times w \). However, the node only selects a next-hop node in its nearby \( l \) rings, where \( l \ll l_r \). In this case, the ratio of the area of \( R_i \cap Q \) to that of \( R_k \cap Q \) can be well approximated by \((2i - 1)/(2k - 1)\), for \((j - l)_+ < i, k < j \). (See the first two rings next to node X in Figure 3-2.)

### 3.3 Experimental Results

In this section, we use experimental results to show how well our method for computing the node densities works. The radius of the sensor field is 50 and the total number of rings is 20 \((= n)\). The procedure of the experiments is as follows. First, for the given routing strategy and experimental parameters, we calculate the node densities of the rings using the equations introduced in Section 3.2. Then, in our simulation setup, we randomly deploy the sensor nodes into the sensor field according to the calculated densities and have each node select its next-hop neighbor according to the given routing scheme. The density of the outmost ring \( \rho_n \) can be tuned to control the total number of nodes in the sensor field. In the simulation run, we measure the energy consumption
rate of each node. Finally, we compute the average per-node energy consumption rate for each ring. The goal is to verify whether the calculated densities result in an even energy consumption rate in all rings.

### 3.3.1 Uniform Ring Selection

The results for various maximum jump sizes are shown in Figure 3-3, where the path loss exponent, $\alpha$, is 2. We have conducted extensive experiments for other values of $\alpha$; but the results are omitted for brevity. In Figure 3-3, we show both the average per-node energy consumption rate and the calculated node density in each of the rings. Several observations can be made. First, the average per-node energy consumption rates of the rings are nearly identical. This demonstrates that our modeling approach and analytical method are highly accurate, and that correct node densities can be derived from the resulting mathematical expressions. Second, the shape of the density function, as a function of the ring index, is somewhat surprising in some cases. The functions are not even monotonic in the case of $l = 10$ or $l = 20$.

In the cases of $l = 1$ or $l = 2$, the density function is monotonic and increases very fast as the ring gets closer to the sink. It is easy to explain the case of $l = 1$. Since the maximum jump size is 1, all the traffic of a node must flow through the adjacent ring on the inside. Therefore, the traffic load becomes heavier as the ring gets closer to the sink. It is necessary to deploy more nodes in the rings closer to the sink so as to balance the energy dissipation rates across the rings. As it approaches the sink, the area of the ring decreases while the number of nodes in the ring increases. Hence, the density increases fast.

For larger values of the maximum jump, e.g., $l = 10$, it is not necessarily true that higher node densities are required for rings closer to the sink. This is more due to the “boundary effect”. In this case, each node can directly transmit its traffic to multiple inside rings. However, longer transmission distance requires more energy. A node in one of the $l$ inner-most rings ($R_i$, $1 \leq i \leq l$) has fewer than $l$ rings left on the inside.
Figure 3-3. Node densities and average per-node energy consumption rates for various maximum jump sizes, \( l \), under uniform ring selection. \( \alpha = 2 \).

Hence, its maximum transmission distance is less than \( l \) rings away. As a result, it tends to consume less energy on average than a node in a ring further outside, say \( R_j \) for \( j > l \). The precise situation is complicated, depending on the parameters of the energy consumption model and the routing probabilities.

### 3.3.2 Uniform Node Selection

Unlike the case of Uniform Ring Selection, here, the node densities are computed by successive substitution as in (3–14). The results for the case of \( \alpha = 0 \) are given in Figure 3-4. The average per-node energy consumption rates in all rings are nearly identical in each of the four plots, which correspond to \( l = 1, 2, 10 \) and 20, respectively. When the path loss exponent, \( \alpha \), is 0, it takes a constant amount of energy for a node to
transmit one unit of traffic to any receiver in its range, regardless of the distance. Many wireless devices do not have the capability to adjust the transmission power level, and hence, fit into the case of $\alpha = 0$.

Figure 3-4. Node densities and average per-node energy consumption rates for various maximum jump sizes, $l$, under uniform node selection. $\alpha = 0$.

The results for $\alpha = 1$, 2 and 3 are shown in Figure 3-5, 3-6 and 3-7, respectively. In all plots, the curve for the average per-node energy consumption rate is flat. This means that, if we deploy the nodes according to the computed densities, we can achieve an even energy dissipation rate in all rings. These results indicate that our modeling approach, analytical method and numerical solutions are all accurate or sound. Observe the curves for the node densities, which can be quite oscillatory or irregular. We see that it is hard to predict the deployment densities without precise computation.
3.4 General Sensor Field and Routing Models

In this section, we consider general models where the shape of the sensor field is arbitrary and the routing depends on the precise node location. The resulting node density may vary continuously over the sensor field. For brevity, we omit some details in the derivation, which is similar to the case with rings.

3.4.1 General Two-Dimensional Model

3.4.1.1 Node-density independent routing

Let $\mathcal{A} \subset \mathbb{R}^2$ denote the whole region of the sensor field and let the sink be at the origin. The region near the sink is an anomaly for the model. Let $B(\delta) = \{ y \in \mathbb{R}^2 | \|y\| < \delta \}$ for some small $\delta > 0$. We assume that there are no other nodes in $B(\delta)$ except the
sink. Once a transmission reaches inside $B(\delta)$, it is received by the sink. We let $A_\delta$ be the sensor field with $B(\delta)$ removed, i.e., $A_\delta = A \setminus B(\delta)$. We wish to find the node density in $A_\delta$.

Consider a point $y \in A_\delta$. Let $g(y)$ be the total traffic rate of a node at $y$. Let $c(y)$ be the rate of the traffic to be relayed by a node at $y$. We again assume that the rate of the locally generated traffic at each point is inversely proportional to the node density at that point. Then, this rate at a node is $K / \rho(y)$ for some constant $K > 0$. We have,

$$g(y) = c(y) + K / \rho(y).$$

(3–22)
Figure 3-7. Node densities and average per-node energy consumption rates for various maximum jump sizes, \( l \), under uniform node selection. \( \alpha = 3 \).

Let \( f(y, x) \) be the probability density function for a node at \( y \) to choose a node at location \( x \) as the next-hop node, \( x \in \mathcal{A} \). It satisfies \( \int_{\mathcal{A}} f(y, x) dx = 1 \) for every \( y \).\(^3\) Note that a node at \( y \) in general may not be able to reach directly everywhere in the whole sensor field. If the region that it can reach directly is denoted by \( A(y) \), where \( A(y) \subseteq \mathcal{A} \), we can assume that \( f(y, x) \) is non-zero only on \( A(y) \), and \( \int_{A(y)} f(y, x) dx = 1 \).

\(^3\) We allow the next-hop node to be the sink. Hence, we assume \( f(y, x) = 0 \) for \( x \in B(\delta) \) and \( x \neq 0 \). That is, if the next-hop node is inside the small neighborhood of the sink, \( B(\delta) \), it must be the sink.
We then have,
\[ c(y) = \int_{A_\delta} \frac{\rho(z)}{\rho(y)} g(z) f(z, y) dz, \quad y \in A_\delta. \] (3–23)

Note that \( c(y) \) depends on the node density function \( \rho \). But, we suppress it in the notation for now. Then, we have
\[ g(y) = \int_{A_\delta} \frac{\rho(z)}{\rho(y)} g(z) f(z, y) dz + \frac{K}{\rho(y)}, \quad y \in A_\delta. \] (3–24)

After rearrangement,
\[ \rho(y) = \int_{A_\delta} \frac{g(z)}{g(y)} f(z, y) \rho(z) dz + \frac{K}{g(y)}, \quad y \in A_\delta. \] (3–25)

Let \( P(y) \) be the expected energy consumption for a node at \( y \in A_\delta \).
\[ P(y) = \int_A (\gamma + \beta \| y - x \|^\alpha) g(y) f(y, x) dx, \quad y \in A_\delta. \] (3–26)

Let \( y^* \) be an arbitrary point on the boundary of \( A \), whose node density is assumed to be a known parameter. Our objective is to have \( P(y) = P(y^*) \) for all \( y \in A_\delta \), where
\[ P(y^*) = \frac{K}{\rho(y^*)} \int_A (\gamma + \beta \| y^* - x \|^\alpha) f(y^*, x) dx. \] (3–27)

This gives the following.
\[ g(y) = \frac{P(y^*)}{\int_A (\gamma + \beta \| y - x \|^\alpha) f(y, x) dx}, \quad y \in A_\delta. \] (3–28)

In the above, we assume \( f \) is independent on the node density function. \( \rho(y^*) \) is a constant (parameter). Then, \( g(y) \) can be determined for all \( y \in A_\delta \). Then, (3–25) is a linear integral equation with the unknown function \( \rho \). It is known as a Fredholm equation of the second kind \[28\], which has the following general form.
\[ \phi(x) = f(x) + \lambda \int_V k(x, s) \phi(s) ds. \] (3–29)

In (3–29), \( \lambda \) is a known constant, \( \phi(x) \) is an unknown function and \( f(x) \) is a known function, \( f(x) \neq 0 \), where \( \phi, f : U \to \mathbb{R} \) for some \( U \subseteq \mathbb{R}^m \). The function \( k(x, s) \) is called a
kernel, where \( k : U \times U \to \mathbb{R} \). In the integral of (3–29), \( V \) is a subset of \( U \). For our case of (3–25), \( \lambda = 1 \), the unknown function is \( \rho \), the known function is \( K/g \), and the kernel is

\[
k(z, y) = \frac{g(z)}{g(y)} f(z, y)
= \frac{\int_A (\gamma + \beta \|y - x\|^\alpha) f(y, x) dx}{\int_A (\gamma + \beta \|z - x\|^\alpha) f(z, x) dx} f(z, y).
\tag{3–30}
\]

It has been shown by the theorems known as Fredholm Alternatives that the solution to (3–29) nearly always exists and is nearly always unique.

### 3.4.1.2 Node-density dependent routing

In general, we can assume \( f \) depends on the density function, and write \( f(y, x, \rho) \), where \( \rho \) is a function on \( A_\delta \). As an example, \( f(y, x, \rho) \) may be proportional to the node density at \( x \) and some other properties at \( y \) and \( x \), denoted by \( h(y, x) \), if \( x \) is in the region that a node at \( y \) can reach. That is,

\[
f(y, x, \rho) = \begin{cases} 
\frac{h(y, x, \rho)\rho(x)}{\int_{A_\delta} h(y, x, \rho) dx} & x \in A(y) \\
0 & \text{otherwise.} 
\end{cases}
\tag{3–31}
\]

Then, we can write,

\[
g(y, \rho) = c(y, \rho) + \frac{K}{\rho(y)}, \quad y \in A_\delta.
\tag{3–32}
\]

\[
c(y, \rho) = \int_{A_\delta} \frac{\rho(z)}{\rho(y)} g(z, \rho) f(z, y, \rho) dz, \quad y \in A_\delta.
\tag{3–33}
\]

If it is required that the per-node energy consumption rate is equal everywhere, the outgoing traffic from a node at \( y \) satisfies the following, which also depends on \( \rho \).

\[
g(y, \rho) = \frac{\int_A (\gamma + \beta \|y^* - x\|^\alpha) f(y^*, x, \rho) dx}{\int_A (\gamma + \beta \|y - x\|^\alpha) f(y, x, \rho) dx} K \frac{1}{\rho(y^*)}, \quad y \in A_\delta.
\tag{3–34}
\]

The function \( \rho \) satisfies the following functional equation.

\[
\rho(y) = \int_{A_\delta} \frac{g(z, \rho)}{g(y, \rho)} f(z, y, \rho) \rho(z) dz + \frac{K}{g(y, \rho)}, \quad y \in A_\delta.
\tag{3–35}
\]
Unlike the case with node density independent routing, (3–35) is not a typical linear integral equation and the mathematical theory on the existence and uniqueness of the solution is unknown at this point. However, the earlier model with the ring structure and energy-dependent routing in Section 3.2.2.2 is a special discrete analog of this and, there, our computation experience has shown that successive substitution always finds a solution. There are good reasons to believe that a solution to (3–35) often exists and can be found by successive substitution.
CHAPTER 4
MAXIMIZING THE LIFETIME OF WIRELESS SENSOR NETWORKS WITH MOBILE SINK IN DELAY-TOLERANT APPLICATIONS

4.1 Overview

A wireless sensor network (WSN) consists of sensor nodes capable of collecting information from the environment and communicating with each other via wireless transceivers. The collected data will be delivered to one or more sinks, generally via multi-hop communication. The sensor nodes are typically expected to operate with batteries and are often deployed in not-easily-accessible or hostile environments, sometimes in large quantities. It can be difficult or impossible to replace the batteries of the sensor nodes. On the other hand, the sink is typically rich in energy. Since the sensor energy is the most precious resource in a WSN, efficient utilization of the energy to prolong the network lifetime has been the focus of much of the research on WSNs.

Although the lifetime of a WSN can be defined in many ways, we adopt the widely used definition, which is the time until the first node exhausts its energy. Much work has been done during recent years to increase the lifetime of a WSN. Among them, in spite of the difficulties in realization, taking advantage of mobility in the WSN has attracted much interest from researchers [7, 14, 16, 25, 31, 36, 38, 40, 41]. We can take the mobile sink as an example of mobility in a WSN. Communication in a WSN often has the many-to-one property in that data from a large number of sensor nodes needs to be concentrated to one or a few sinks. Since multi-hop routing is generally needed for distant sensor nodes to send data to the sink\(^1\), the nodes near the sink can be burdened with relaying a large amount of traffic from other nodes. This phenomenon is sometimes called the “crowded center effect” [32] or the “energy hole problem” [23, 24, 44]. It results in early energy depletion at the nodes near the sink, potentially

---

\(^1\) For ease of discussion, we assume there is only one sink.
disconnecting the sink from the remaining sensors that still have plenty of energy. However, by moving the sink in the sensor field, one can avoid or mitigate the energy hole problem and expect an increased network lifetime.

This chapter proposes a framework to maximize the lifetime of a WSN by taking advantage of sink mobility. Compared with other mobile-sink proposals, the main novelty is that we consider the case where the underlying applications tolerate delayed information delivery to the sink. One of the application examples is battlefield surveillance, where sensor nodes are deployed to monitor the movement of enemy vehicles or troops. A mobile sink attached to an unmanned aerial vehicle flies over the monitored region regularly to harvest the collected intelligence. To avoid being intercepted or detected by enemy forces, the mobile sink needs to operate in only a few safe locations within a limited operation time. Another example is habitat monitoring where a mobile robot is used to collect information from the sensor nodes in the field. If much of the habitat area is not be accessible by the robot or if it is desirable to minimize disturbance to the targeted animal species, the mobile robot will trace predetermined paths and stop by a set of pre-arranged locations regularly for data collection.

In our proposal, within a prescribed delay tolerance level, each node does not need to send the data immediately as it becomes available. Instead, the node can store the data temporarily and transmit it when the mobile sink is at the location most favorable for achieving the longest network lifetime. To find the best solution within the proposed framework, we formulate optimization problems that maximize the lifetime of the WSN subject to the delay bound constraints, node energy constraints and flow conservation constraints. Another one of our contributions is that we compare our proposal with several other lifetime-maximization proposals and quantify the performance differences among them. Our computational experiments have shown that our proposal increases the lifetime significantly when compared to not only the stationary sink model but also more traditional mobile sink models without delay tolerance.
Our proposal is more sophisticated than most previous lifetime-improvement proposals that we know of. It integrates the following energy-saving techniques, multipath routing, a mobile sink, delayed data delivery and active region control, into a single optimization problem. Such sophistication comes at a cost. Whether the proposal should be adopted in practice will depend on the tradeoff between the lifetime gain and the actual system cost. The latter includes all costs/complexity in implementing the proposal and in actual operations. These may include extra communication protocols for coordination and control, e.g., new routing and rate control protocols, extra memory for keeping delayed data and memory-management costs, and application-level costs incurred by delayed information delivery. Even if the decision is not to adopt it due to a high cost or high complexity, the framework in the chapter is still useful because it can supply the practitioners with a performance benchmark, e.g., how much lifetime improvement opportunity there is. By also formulating the optimization problems related to other proposals and providing cross comparison, the chapter provides extra convenience for comparing and understanding different proposals.

Being one of the early papers on extending the network lifetime with mobility and delay tolerance, the chapter focuses on formulating several simple and typical lifetime-maximization problems and evaluating the lifetime improvement. There can be many variants of the problem formulation, some of which can be very difficult, often involving NP-hard combinatorial sub-problems. The degree of lifetime improvement demonstrated by this chapter can justify further work on more difficult problems.

We now briefly review the most relevant work on how to exploit mobility to increase the network lifetime. In [36], the authors introduced mobile agents, which move around and collect data from nearby sensor nodes on behalf of the immobile sink. When the mobile agents move to the vicinity of the sink, they forward the collected data to the sink. In that framework, communication occurs only from the sensor nodes to the mobile agents or from the mobile agents to the sink via a single hop; the sensor nodes do not
relay traffic. Hence, it is different from our multi-hop communication framework. It was assumed that the mobile agents have plenty of energy. The movement of each mobile agent is modeled as a random walk. It was shown that the queues in the mobile agents and the sensor nodes are finite and the delay of the collected data is bounded. However, the authors did not show the quantitative improvement of the network lifetime by using mobile agents.

In [41], the authors formulated a linear programming problem of determining how to move the mobile sink and how long to park the mobile sink at each stop along the path of the sink so as to maximize the lifetime of the WSN. However, in their model, data flows are not decision variables of the lifetime optimization problem. On the contrary, in our formulations, not only the sink sojourn times at different sink stops but also the routing scheme are decision variables. The analysis and experiments in [41] were conducted under a simple structured network topology where the sensor nodes are deployed in a grid-like pattern. In [7] and [8], the authors further extended the research of [41]. The model proposed in [7] [8] includes the cost of moving the mobile sink (such as nodal energy consumption for route establishment/release when the sink moves to a new stop) and the sink mobility rate determined by the minimum sink sojourn time at the sink stops. Furthermore, the model incorporates a hop-length limit when the sink moves to next stop. This restricts the packet latency, which is related to the traveling time of the sink between stops. The authors proposed an MILP (Mixed Integer Linear Programming) problem formulation to obtain the optimal travel route of the sink and the sojourn times at the sink stops for maximizing the lifetime of the system. They also suggested a distributed heuristic algorithm to circumvent the complexity of the proposed mathematical formulation.

The authors of [25] showed that the network lifetime can be extended significantly if the mobile sink moves around the periphery of the WSN. They assumed that, if the mobile sink can balance the traffic load of the nodes, the lifetime of the network can
increase. Therefore, they proposed an optimization problem for choosing a mobility strategy that minimizes the maximum traffic load of the nodes. However, they assumed the shortest path routing, which, in general, does not produce the best lifetime.

The problem of finding the trajectory of the mobile sink so as to optimize the lifetime of the WSN is hard to solve due to its infinite search space when the locations for the sink stops are not constrained. In [38], the authors studied how to find the optimal sink stops and the schedule of visit to each of the stops. If the candidate locations for the stops are unconstrained, this problem is also NP-hard. However, if the stops are constrained to be selected from a finite set of known locations, the problem can be easily formulated into linear programming. They proposed an approximation algorithm to the unconstrained problem by properly dividing the whole sensor field into a finite number of disjoint small areas, and then, converted the unconstrained problem into a constrained problem. However, to obtain a good approximation ratio, the number of small areas can potentially be very large, making the linear programming computation time-consuming. Therefore, in this chapter, we restrict the set of potential sink stops to be from a small number of given locations rather than from arbitrary locations.

The WSN model proposed in [31] is close to ours. The authors studied the maximum lifetime problem of the WSN where the mobile sink can visit only small number of locations. They showed that the lifetime can be further increased by optimizing not only the schedule of sink visits but also routing of the traffic. However, they did not consider applications where delayed information delivery is allowed.

4.2 Related Lifetime Maximization Problems

In this section, we discuss related lifetime maximization problems that have been published in the literature. We will later compare their performance with our new proposal.

First, we will describe the general assumptions about the WSN models. Let the set of sensor nodes be denoted by \( \mathcal{N} \). For experimental convenience, we suppose they are
uniformly randomly deployed into a circular area with radius $R$. Let the center of the disk be the origin. Each node $i$ is assumed to generate data at a constant rate of $d_i$ during its life span and the initial energy of $i$ is denoted by $E_i$. Furthermore, the nodes have the ability of adjusting their transmission power level to match the transmission distance. Similar to [19], the energy required per unit of time to transmit data at the rate of $x_{ij}$ from node $i$ to $j$ can be determined as follows.

$$E_{ij}^t = C_{ij}^t \cdot x_{ij}, \quad (4-1)$$

where $C_{ij}^t$ is the required energy for transmitting one unit of data from node $i$ to $j$ and it can be modeled as follows [33].

$$C_{ij}^t = \alpha + \beta \cdot d(i,j)^e, \quad (4-2)$$

where $d(i,j)$ is the Euclidean distance between node $i$ and $j$, $\alpha$ and $\beta$ are nonnegative constants, and $e$ is the path loss exponent. Typically, $e$ is in the range of 2 to 6, depending on the environment. Here, the energy cost per unit of data does not depend on the link rate, and this is valid for the low rate regime. Hence, we need to assume that the traffic rate $x_{ij}$ is sufficiently small compared to the capacity of the wireless link.

The energy consumed at node $i$ per unit of time for receiving data from node $k$ is given by [19]

$$E_{ki}^r = \gamma \cdot x_{ki}, \quad (4-3)$$

where $\gamma$ is a given constant. Hence the total energy consumption per unit time at node $i$ is

$$\sum_{j \in \mathcal{N}} E_{ij}^t + \sum_{k \in \mathcal{N}} E_{ki}^r = \sum_{j \in \mathcal{N}} C_{ij}^t \cdot x_{ij} + \sum_{k \in \mathcal{N}} \gamma \cdot x_{ki}. \quad (4-4)$$

We assume that each sensor node has the same transmission range. Let $l$ denote the sink. In this paper, we take the convention that the sink is a special node different from the sensor nodes and $l \notin \mathcal{N}$. The required energy for transmitting one unit of
data from a sensor node \(i\) to the sink \(l\) is denoted by \(C_{il}^j\), and it is given by \(4.2\) with \(j\) replaced by \(l\). We define the (downstream) neighbors of node \(i\) as \(N(i) = \{j \in \mathcal{N} \cup \{l\} | d(i, j) \leq \bar{d}\}\), when the transmission range is \(\bar{d}\). Note that the neighbors may include the sink.

The paper does not consider MAC-layer contention. It is assumed that contention is resolved by some MAC-layer protocol. The operation of the MAC-layer protocol determines the link rates, which are assumed to be large enough so that they do not impose a constraint on the data rates. Future work may try to relax these assumptions. Conversely, if the data rates are small, then even simple MAC-layer protocols will be able to deliver the required link rates. In other words, it can be easy to design one of such protocols.

### 4.2.1 Static Sink Model

In the static sink model (SSM), the sink is located at the origin and remains stationary during the operation of the WSN. Data originated from the sensor nodes flows into the sink in a multi-hop fashion. As soon as the data becomes available at a node, it gets transmitted toward the sink. Typically, the rate at which each sensor node \(i\) harvests data from the outside world is a constant. We denote it by \(d_i\). The data generated by a source is sometimes called a commodity or a sub-flow [1, 15]. Let \(x_{ij}^c\) be the rate assignment from node \(i\) to the node \(j\) for the traffic generated by node \(c\) (commodity \(c\)). The problem of maximizing the lifetime in this model is formulated as follows [11, 12].
The constraint (4–7) is the “flow conservation constraint”, which states that, at a node \(i\), the sum of all outgoing flows for a commodity \(c\) is equal to the sum of all incoming flows for the commodity \(c\). If \(i = c\), the incoming flows should include the flows generated at node \(i\) itself, or \(d_i\). The inequality (4–8) is the energy constraint and it means that the total energy consumed by a node during the lifetime \((Z)\) cannot exceed the initial energy of the node. With this formulation, the routing is dynamic and allows multipath communications. There is no assumption on fixed-path routing, such as the shortest path routing. The above optimization problem can be easily converted into a linear programming (LP) problem.

The particular formulation above is equivalent to the following formulation, where the flows of the commodities are aggregated into a single arc flow. The new formulation has much reduced complexity and is useful for finding numerical solutions quickly.
However, it is less generalizable.

\[ \text{Aggregate-Traffic SSM} \] (4–11)

\[
\begin{align*}
\text{max} & \quad Z \\
\text{s. t.} & \quad \sum_{j \in N(i)} x_{ij} - \sum_{k : i \in N(k)} x_{ki} = d_i, \quad \forall i \in N \\
& \quad \left( \sum_{j \in N(i)} C_{ij}^t \cdot x_{ij} + \sum_{k : i \in N(k)} \gamma \cdot x_{ki} \right) \cdot Z \leq E_i, \quad \forall i \in N \\
& \quad x_{ij} \geq 0, \quad \forall i \in N; \forall j \in N(i) \\
& \quad Z \geq 0.
\end{align*}
\] (4–13) (4–14) (4–15) (4–16)

Here, \( x_{ij} \) is the aggregate flow rate of all commodities from node \( i \) to node \( j \), i.e., \( \sum_c x_{ij}^c = x_{ij} \). The equivalence of the problems, (4–5) and (4–11), can be argued as follows.

Clearly, we can always construct a feasible solution to problem (4–11) from any feasible solution to problem (4–5) by letting \( x_{ij} = \sum_c x_{ij}^c \). Conversely, given a feasible solution \( \{x_{ij}\} \) to problem (4–11), one can apply the flow decomposition algorithm [1] to the arc flows \( \{x_{ij}\} \) and obtain path flows for the commodities\(^2\). The path flows in turn give the per-commodity arc flows \( \{x_{ij}^c\} \) feasible to problem (4–5).

**Remark:** The equivalence is only true for the particular constraints considered here.

The two formulations are not usually equivalent in more general settings, for instance, if the costs of the commodities (energy per unit of data transmitted or received) are different, or if some individual commodity rate at some link is upper bounded or lower bounded by a non-zero value, which in turn might be the result of assigning different importance levels to different commodities. The per-commodity formulation (4–5) is

\(^2\) In general, the flow decomposition algorithm produces both path flows and cycle flows. However, in any optimal solution of (4–11), the flows on every cycle must be zero. Hence, we can restrict ourself to the set of feasible flows that can be decomposed into path flows only.
more generalizable. The aggregate-traffic formulation (4–11) is not necessarily useful if one wishes to incorporate more constraints. But it is useful in this paper because it is easier to compute.

4.2.2 Mobile Sink Model

In the mobile sink model (MSM), we assume that the sink can move around within the sensor field and stop at certain locations to gather the data from the sensor nodes. Let \( \mathcal{L} \) be the set of possible locations where the sink can stop (also known as sink stops). The sink does not necessarily stop at (i.e., stays for a positive duration) all locations in \( \mathcal{L} \) in the interest of maximizing the network lifetime [38, 41].

As previous authors [38], throughout the paper, we make the assumption that the traveling time of the sink between locations is negligible. This way, the resulting problem formulations are simple enough for us to obtain precise numerical solutions for evaluation purpose. The assumption is appropriate when the traveling time is much smaller than the time spent by the sink to collect data in each location.

In this model, the order of visit to the stops has no effect on the network lifetime and can be arbitrary. The sink sojourn time at a location \( l \in \mathcal{L} \) is denoted by \( z_l \); it is the time that the sink spends at \( l \) to collect data from the sensor nodes. The overall network lifetime is \( Z = \sum_{l \in \mathcal{L}} z_l \). When the sink is at stop \( l \), we denote the (downstream) neighbors of node \( i \) as

\[
N(i, l) = \{ j \in \mathcal{N} \cup \{ l \} | d(i, j) \leq \bar{d} \}. \quad (4–17)
\]

To find the optimal network lifetime, we need to consider the routing of the traffic as well as the duration of the sink’s sojourn time at each stop (also see [16, 31, 38, 41]).

Similar to the case of the static sink model in Section 4.2.1, there is a per-commodity-based formulation of the lifetime-maximization problem, and there is an equivalent, simpler, aggregate-traffic-based formulation. For brevity, we will only present the latter. However, we re-iterate that, if additional constraints are present, the per-commodity-based
formulation may be necessary. Let $x_{ij}^{(l)}$ be the aggregate flow on link $(i,j)$ while the sink is at stop $l$. The lifetime maximization problem can be formulated as follows.

**Aggregate-Traffic Mobile Sink Model (MSM)**  

\[
\text{max } Z = z_1 + z_2 + \cdots + z_{|\mathcal{L}|} 
\]  
\[
\text{s. t. } \sum_{j \in N(i,l)} x_{ij}^{(l)} - \sum_{k : i \in N(k,l)} x_{ki}^{(l)} = d_i, \quad \forall i \in \mathcal{N}, \forall l \in \mathcal{L} 
\]  
\[
\sum_{l=1}^{|\mathcal{L}|} z_l (\sum_{j \in N(i,l)} C_{ij}^{(l)} x_{ij}^{(l)} + \sum_{k : i \in N(k,l)} \gamma x_{ki}^{(l)}) \leq E_i, \quad \forall i \in \mathcal{N} 
\]  
\[
x_{ij}^{(l)} \geq 0, \quad \forall i \in \mathcal{N}; \forall l \in \mathcal{L}; \forall j \in N(i,l) 
\]  
\[
z_l \geq 0, \quad \forall l \in \mathcal{L}. 
\]

Constraint (4–20) denotes the flow conservation for all nodes when the sink is at $l$. Constraint (4–21) says that the total energy consumed at the node $i$ can not exceed the initial energy $E_i$. By multiplying (4–20) with $z_l$ and substituting $x_{ij}^{(l)} \cdot z_l$ with a new variable $y_{ij}^{(l)}$, we can replace (4–20) with the following new constraint.

\[
\sum_{j \in N(i,l)} y_{ij}^{(l)} - \sum_{k : i \in N(k,l)} y_{ki}^{(l)} = z_l \cdot d_i, \quad \forall i \in \mathcal{N}; \forall l \in \mathcal{L}. 
\]

Similarly, constraint (4–21) can be changed into

\[
\sum_{l=1}^{|\mathcal{L}|} \left( \sum_{j \in N(i,l)} C_{ij}^{(l)} \cdot y_{ij}^{(l)} + \sum_{k : i \in N(k,l)} \gamma \cdot y_{ki}^{(l)} \right) \leq E_i, \quad \forall i \in \mathcal{N}. 
\]

With the constraints (4–24), (4–25), (4–23), and the non-negativity constraints for $y_{ij}^{(l)}$, the above optimization problem is converted into an LP problem. Here, $y_{ij}^{(l)}$ is interpreted as the total traffic volume that node $i$ sends to node $j$ while the sink stays at $l$. 

76
4.3 Lifetime Maximization in Delay Tolerant Mobile Sink Model

In this section, we consider how to maximize the lifetime of WSN with the mobile sink in applications that can tolerate a certain amount of delay. We call the resulting WSN model *delay tolerant mobile sink model* (DT-MSM). In this setting, each node can postpone the transmission of data until the sink is at the stop most favorable for extending the network lifetime. This way, the nodes can collectively achieve a longer network lifetime. In contrast, the SSM and MSM do not exploit this possibility.

Let $D$ be the maximum tolerable delay, or the delay tolerance level. We assume that the sink finishes one round of visit to all the stops (where the sink stays for a positive duration to collect data) in $D$ time units, and then, repeats with another round again and again. Note that two consecutive visits to the same stop takes a time $D$.

Let’s take an example to show how our framework can outperform other ones. Consider the two-node example shown in Figure 4-1. $N_1$ and $N_2$ are two sensor nodes and $L_1$ and $L_2$ are the candidate stops of the mobile sink. Suppose we ignore the receiving energy requirement and suppose the transmission energy per unit of data is equal to the square of the distance between the sender and the receiver. Both nodes $N_1$ and $N_2$ generate data at 1 bps and have 100 units of energy initially. If the sink is located at $O$ in the SSM, both nodes spend 4 units of energy for sending a bit of data. It is obvious that the optimal lifetime is 25 seconds. In the MSM with sink locations.
\{L_1, L_2\}$, due to the symmetry of the structure, the sink stays at both $L_1$ and $L_2$ for the same amount of time to achieve the maximum lifetime. Each node spends 1 or 9 units of energy for sending 1 bit of data depending whether the sink is at $L_1$ or $L_2$. The average energy consumption per bit is 5 units. Thus, the lifetime is 20 seconds. In the DT-MSM, we assume that the sink alternates between the two stops and stays for 1 second at each stop in each cycle. Hence, this is the case that $D = 2$ seconds. When the sink stays at $L_1$, only $N_1$ sends 2 bits of data to the sink; when the sink moves to $L_2$, only $N_2$ transmits 2 bits of data ($N_2$ keeps its data while the sink is at $L_1$). Both nodes spend 2 units of energy every 2 seconds or 1 unit of energy per second on average. Thus, the lifetime is 100 seconds, a significant increase compared to the SSM and MSM. This is because, in the DT-MSM, the nodes do not always participate in communication for all the sink stops; they each wait until the sink’s location is most favorable for energy saving, and then send data at the higher rate. Recall that we have assumed that the traffic rate is sufficiently small compared to the capacity of the wireless link, and hence, sending data at a higher rate does not alter the per-bit energy consumption.

Unlike the MSM or SSM, the sink in the DT-MSM can collect data from only a subset of the set of all sensor nodes, $\mathcal{N}$, at each stop. Let $R_l$ be the subset of $\mathcal{N}$ such that only nodes in $R_l$ can participate in the communication toward the sink when the sink is at $l \in \mathcal{L}$. We call $R_l$ the \textit{coverage} of the sink location $l$. Note that the union of $R_l$ over $l \in \mathcal{L}$ must be the set of all sensor nodes, $\mathcal{N}$. In other words, any sensor node should be covered by at least one sink location. When the node $i$ is in $R_l$, node $i$ is said to be \textit{active} at $l \in \mathcal{L}$. Although we can construct $R_l$ in many ways depending on the application of interest, in this paper, a very simple method of constructing $R_l$ is considered. Fix a positive number $r$. We call $r$ the \textit{radius of coverage} of the sink. For each $l \in \mathcal{L}$, if $d(i, l) \leq r$, where $i \in \mathcal{N}$, then $i \in R_l$. Here, the radius of coverage of the sink ($r$) should be large enough so that every sensor node belongs to at least one $R_l$. Note that the minimum $r$ depends on the locations of the sink stops.
In both SSM and MSM, the sink collects data from each node $i$ at the same rate at which node $i$ generates the data. However, in the DT-MSM, the data transmission rate at node $i$ during the collection time is no longer the same as the constant data generation rate $d_i$. When node $i$ is not active (i.e., not covered by the current sink location), it continues to gather data and should store the newly generated data. Hence, data buffering is required by our framework. Within a cycle of $D$ time units, the total stored data at each node $i$ is at most $D \cdot d_i$.

For ease of presentation, we assume the sink visits all locations in $\mathcal{L}$ in the order of $1 \rightarrow 2 \rightarrow \cdots |\mathcal{L}| \rightarrow 1 \cdots$. The sink may stay at some location for zero time. With slight abuse of terminology, we define the network lifetime $T$ to be the number of cycles made by the sink until the first node dies due to energy exhaustion. The actual lifetime is $T \cdot D$.

Once traffic is allowed to be buffered, there are different strategies on whose traffic is buffered. Which strategy gets adopted in practice probably depends on the application, other practical concerns, and the designer’s preference. Since we do not know these factors in advance, we next describe two strategies, or two variants of the model: the sub-flow-based model and the queue-based model. The main purpose is to illustrate that choices exist and they lead to different performance-complexity tradeoffs.

4.3.1 Sub-Flow-Based Model

In the sub-flow-based model, the nodes in the current coverage $R_i$ are not allowed to buffer the relayed traffic from other nodes; as soon as a node in $R_i$ receives the data from other nodes, it immediately forwards the data to its neighboring nodes. To model this constraint at each node $i$, we need to differentiate the data generated by node $i$ itself and the data originally generated by other nodes but forwarded to node $i$.

Again, let $x_{ij}^{(c:\ell)}$ be the rate assignment from node $i$ to the node $j$, while the sink is at $\ell$, for the traffic generated by node $c$ (commodity $c$). Let $x_{ij}^{(\ell)}$ be the aggregated rate of
traffic that needs to be forwarded to node $j$ from node $i$ when the sink is at $l$. That is,

$$x_{ij}^{(l)} = \sum_{c \in R_i} x_{ij}^{(c,l)}, \quad \forall l \in \mathcal{L}; \forall i, \forall j \in R_l.$$  \hfill (4–26)

Since at node $i \in \mathcal{N}$, the commodity or sub-flow of other nodes $c \in R_l, c \neq i$ must be forwarded as soon as it has been received, we must have

$$\sum_{k: i \in N_l(k)} x_{ki}^{(c,l)} = \sum_{j \in N_l(i)} x_{ij}^{(c,l)}. \quad \hfill (4–27)$$

Here, we define $N_l(i) = R_l \cap N(i, l)$, where $N(i, l)$ is as given in (4–17). The flow conservation at node $i$ can be expressed as follows, which is the same as in the MSM except that the amounts of traffic originated from node $i$ itself, $(w_{i}^{(l)}; l \in \mathcal{L}, i \in R_l)$, are now decision variables.

$$z_{l} \left( \sum_{j \in N_l(i)} x_{ij}^{(l)} - \sum_{k: i \in N_l(k)} x_{ki}^{(l)} \right) = w_{i}^{(l)}. \quad \hfill (4–28)$$

The data buffered during the previous sink-movement cycle must be cleared in the current cycle. This requirement can be written as

$$\sum_{l: i \in R_l} w_{i}^{(l)} = D \cdot d_i. \quad \hfill (4–29)$$
The following is the formulation for the lifetime maximization problem in sub-flow-based DT-MSM.

\[
\text{Per-Commodity Sub-Flow-Based DT-MSM} \tag{4\textendash}30
\]

\[
\begin{align*}
\text{max} & \quad T \tag{4\textendash}31 \\
\text{s. t.} & \quad \sum_{c \in R_l} x_{ij}^{(c,l)} = x_{ij}^{(l)}, \quad \forall l \in \mathcal{L}, \forall i, \forall j \in R_l \\
& \quad \sum_{k : i \in N_l(k)} x_{kj}^{(c,l)} = \sum_{j : i \in N_l(i)} x_{kj}^{(c,l)}, \quad \forall l \in \mathcal{L}, \forall i, \forall c(\neq i) \in R_l \\
& \quad z_l \left( \sum_{j \in N_l(i)} x_{ij}^{(l)} - \sum_{k : i \in N_l(k)} x_{ki}^{(l)} \right) = w_i^{(l)}, \quad \forall l \in \mathcal{L}, \forall i \in R_l \\
& \quad \left\{ \sum_{l : i \in R_l} w_i^{(l)} = D \cdot d_i, \quad \forall i \in \mathcal{N} \right. \\
& \quad \left. \sum_{c \in R_l} x_{ij}^{(l)} \geq 0, \quad \forall l \in \mathcal{L}, \forall i, \forall c \in R_l, \forall j \in N_l(i) \right. \\
& \quad \left. w_i^{(l)} \geq 0, \quad \forall l \in \mathcal{L}, \forall i \in R_l \right. \\
& \quad z_l \geq 0, \quad \forall l \in \mathcal{L} \\
& \quad T \geq 0.
\end{align*}
\]

The above is a per-commodity-based formulation of the problem. Similar to the case of the SSM problem (4\textendash}5) in Section 4.2.1, there is a simpler, equivalent, aggregate-traffic formulation, using only the aggregate arc flow variables \( x_{ij}^{(l)} = \sum_{c \in R_l} x_{ij}^{(c,l)} \). We can do away with constraint (4\textendash}33) in the resulting formulation.
\[ \text{Aggregate-Traffic Sub-Flow-Based DT-MSM} \]

\[
\begin{align*}
\text{max } & \quad T \\
\text{s. t. } & \quad z_i \left( \sum_{j \in N_l(i)} x_{ij}^{(l)} - \sum_{k: i \in N_l(k)} x_{ki}^{(l)} \right) = w_i^{(l)}, \quad \forall l \in L; \forall i \in R_l \\
& \left\{ \sum_{l=1}^{|L|} z_l \left( \sum_{j \in N_l(i)} C_{ij}^l \cdot x_{ij}^{(l)} + \sum_{k: i \in N_l(k)} \gamma \cdot x_{ki}^{(l)} \right) \right\} \cdot T \leq E_i, \quad \forall i \in N \\
& \sum_{l: i \in R_l} w_i^{(l)} = D \cdot d_i, \quad \forall i \in N \\
& x_{ij}^{(l)} \geq 0, \quad \forall l \in L; \forall i \in R_l; \forall j \in N_l(i) \\
& w_i^{(l)} \geq 0, \quad \forall l \in L, \forall i \in R_l \\
& z_l \geq 0, \quad \forall l \in L \\
& T \geq 0.
\end{align*}
\]

The equivalence of the two formulations can be argued as follows. First, it is clear that the feasibility set of the per-commodity-based formulation is inside the feasibility set of the aggregate-traffic-based formulation. Conversely, given a feasible solution \( \{ x_{ij}^{(l)}, w_i^{(l)}, z_l \} \) to the latter formulation, we can treat \( w_i^{(l)}/z_l \) as the supply at each node \( i \) when the sink is at stop \( l \). For each \( l \), we can decompose the arc flows \( \{ x_{ij}^{(l)} \} \) into path flows for all the commodities, \( \{ x_{ij}^{(c,l)} \} \) (again, without loss of generality, we assume the decomposition does not lead to positive cycle flows), satisfying (4–32), (4–33) and (4–34).

Hence, \( \{ x_{ij}^{(c,l)}, w_i^{(l)}, z_l \} \) is a feasible solution to the per-commodity-based formulation.

\subsection*{4.3.2 Queue-Based Model}

In the \textit{queue-based model}, each sensor node can buffer data originated from any node. Let \( q_i^{(l)} \) be the queue length at node \( i \) just before the sink moves from location \( l \) to \( l + 1 \). Assume that each node \( i \) has \( D \cdot d_i \) amount of data at the beginning of a cycle,
which is denoted by \( q_i^{(0)} \). When the sink finishes a cycle of visit, the queue at node \( i \) must be cleared. Thus we have \( q_i^{(|L|)} = 0 \). In this model, the flow conservation constraint is replaced by the queue length dynamics, which is expressed as follows.

\[
z_l(\sum_{k: i \in N_l(k)} x_{ki}^{(l)}) + q_i^{(l-1)} - z_l(\sum_{j \in N_l(i)} x_{ij}^{(l)}) = q_i^{(l)}, \quad \forall l \in L; \forall i \in N. \tag{4–50}
\]

The energy constraints can be expressed in the same way as in the sub-flow based MSM. From the above discussion, we have the following optimization problem for maximizing the lifetime.

**Queue-Based DT-MSM**

\[
\begin{align*}
\text{max} & \quad T \tag{4–51} \\
\text{s. t.} & \quad z_l(\sum_{k: i \in N_l(k)} x_{ki}^{(l)}) + q_i^{(l-1)} - z_l(\sum_{j \in N_l(i)} x_{ij}^{(l)}) = q_i^{(l)}, \quad \forall l \in L; \forall i \in N \tag{4–53} \\
& \quad \left\{ \sum_{l=1}^{|L|} z_l \left( \sum_{j \in N_l(i)} C_{ij}^{l} \cdot x_{ij}^{(l)} + \sum_{k: i \in N_l(k)} \gamma \cdot x_{ki}^{(l)} \right) \right\} \cdot T \leq E_i, \quad \forall i \in N \tag{4–54} \\
& \quad q_i^{(0)} = D \cdot d_i, \quad \forall i \in N \tag{4–55} \\
& \quad q_i^{(|L|)} = 0, \quad \forall i \in N \tag{4–56} \\
& \quad x_{ij}^{(l)} \geq 0, \quad \forall l \in L; \forall i \in R_l; \forall j \in N_l(i) \tag{4–57} \\
& \quad q_i^{(l)} \geq 0, \quad \forall i \in N; \forall l \in L \tag{4–58} \\
& \quad z_l \geq 0, \quad \forall l \in L \tag{4–59} \\
& \quad T \geq 0. \tag{4–60}
\end{align*}
\]

The problem shown above can be converted into an LP problem by substituting \( y_{ij}^{(l)} \) for \( z_l \cdot x_{ij}^{(l)} \) and introducing the new variable \( u = 1 / T \). This linearization method can also be applied to the sub-flow based MSM.

**Discussion on the two delay-tolerant models:**

• The two delay-tolerance formulations represent two strategies on what data to buffer. The sub-flow-based formulation allows buffering of only self-generated traffic; the queue-based formulation allows buffering of any traffic, which naturally leads to the best lifetime performance among different strategies. The two models can be considered as two “extreme cases”, and various intermediate strategies can be similarly formulated.

• In the sub-flow-based formulation, the maximum required buffer size at node $i$ is $Dd_i$. In the queue-based case, the maximum buffer size at a node may depend on the total number of other nodes in the same coverage area, which can be much larger.

• The sub-flow-based formulation looks more similar to the standard multi-commodity flow problem. It can be easier to find fast, specialized algorithms to solve this problem.

4.3.3 Properties of Delay-Tolerant Mobile Sink Model

Both delay-tolerant models include the coverage of each sink location in the formulation. This is motivated by practical concerns, in particular, how easy it is to design practical protocols for coordinating the communication. When at a sink location, it is far easier for the sink to coordinate with the nearby sensors and set up the data collection process. Hence, a small radius of coverage is preferable from the protocol complexity point of view. However, the radius of coverage can affect the network lifetime, which we will explore next.

For illustration, consider the optimization problem for the sub-flow-based model. Depending on the radius of coverage, we may obtain different instances of the optimization problem. Thus, we can parameterize these instances according to the radius of coverage. Let $P(\mathcal{N}, \mathcal{L}, r)$ be the optimization problem when the radius of coverage of the sink is $r$, the set of sensor nodes is $\mathcal{N}$, and the set of sink locations is $\mathcal{L}$. The value $r$ must be large enough so that all sensor nodes can be covered by at least one sink location and we denote this minimum radius of coverage for connectivity by $r_0$. Under the same configuration with $\mathcal{N}$ and $\mathcal{L}$, different $r$ values only affect $R_i$ and $N_i(i)$. We will use the notations $R_i(r)$ and $N_i(i, r)$ if it is necessary to specify the radius.
of coverage. In the next theorem, we prove that the bigger the radius of coverage, the longer the optimal lifetime is.

**Theorem 1.** If \( r_0 \leq r_1 < r_2 \), then the optimal objective value for the problem \( P(\mathcal{N}, \mathcal{L}, r_2) \) is greater than or equal to that for the problem \( P(\mathcal{N}, \mathcal{L}, r_1) \).

**Proof.** Consider the two optimization problems \( P(\mathcal{N}, \mathcal{L}, r_1) \) and \( P(\mathcal{N}, \mathcal{L}, r_2) \) with \( r_1 < r_2 \).

It is obvious that \( N_l(i, r_1) \subseteq N_l(i, r_2) \) for all \( i \in \mathcal{N}, l \in \mathcal{L} \). Therefore, we can split the larger set \( N_l(i, r_2) \) into two sets \( A \) and \( \bar{A} \), where \( A = N_l(i, r_1) \), and \( \bar{A} = N_l(i, r_2) \setminus A \). Similarly, we can also split the upstream neighbor set for node \( i \) into \( B = \{ k \in \mathcal{N} | i \in N_l(k, r_1) \} \) and \( \bar{B} = \{ k \in \mathcal{N} | i \in N_l(k, r_2) \} \setminus B \). In other words, \( A \) and \( \bar{B} \) are the sets of additional downstream and upstream neighbors for node \( i \), respectively, as the radius of coverage increases from \( r_1 \) to \( r_2 \).

Suppose that \((\hat{x}, \hat{w}, \hat{z}, \hat{T})\) is a feasible solution to the problem \( P(\mathcal{N}, \mathcal{L}, r_1) \). Now, consider equation (4–63) for the optimization problem \( P(\mathcal{N}, \mathcal{L}, r_2) \). For \( \forall l \in \mathcal{L}, \forall i \in R_l(r_2) \),

\[
\begin{align*}
  z_l \left( \sum_{j \in N_l(i, r_2)} x_{ij}^{(l)} - \sum_{k: i \in N_l(k, r_2)} x_{ki}^{(l)} \right) = w_i^{(l)}. \quad (4–61)
\end{align*}
\]

We have the following by separating the neighbor sets into \( A, \bar{A}, B, \) and \( \bar{B} \).

\[
\begin{align*}
  z_l \left( \sum_{j \in A} x_{ij}^{(l)} + \sum_{j \in \bar{A}} x_{ij}^{(l)} - \sum_{k \in B} x_{ki}^{(l)} - \sum_{k \in \bar{B}} x_{ki}^{(l)} \right) = w_i^{(l)}. \quad (4–62)
\end{align*}
\]

Fix \( l \in \mathcal{L} \). Suppose \( i \in R_l(r_1) \). We extend the vector \( \hat{x} \) so that \( \hat{x}_{ij}^{(l)} = 0 \) when \( j \in \bar{A} \) and \( \hat{x}_{ki}^{(l)} = 0 \) when \( k \in \bar{B} \). Then, for such \( l \) and \( i \), the extended vector \((\hat{x}, \hat{w}, \hat{z}, \hat{T})\) satisfies (4–62) since the original vector satisfies (4–63) for \( i \in R_l(r_1) \).

Next suppose \( i \in R_l(r_2) \setminus R_l(r_1) \). Then, we can extend the vector \( \hat{x} \) further by setting \( \hat{x}_{ij}^{(l)} = 0 \) for all \( j \in N_l(i, r_2) \) and \( \hat{x}_{ki}^{(l)} = 0 \) for all \( k \) such that \( i \in N_l(k, r_2) \). Furthermore, we extend \( \hat{w} \) by setting \( \hat{w}_{i}^{(l)} = 0 \). After such extension, \((\hat{x}, \hat{w}, \hat{z}, \hat{T})\) satisfies (4–62) for \( i \in R_l(r_2) \setminus R_l(r_1) \) (since all terms are zero).
For the energy constraint (4–44), we can apply a similar procedure. Hence, we can conclude that any feasible solution to the problem $P(N, L, r_1)$, after suitable extension, is also a feasible solution to the problem $P(N, L, r_2)$.

Next, the queue-based model is less constraining than the sub-flow-based model; this results in lifetime gains in the former model. The following theorem formalizes the fact that the queue-based model always outperforms the sub-flow-based model.

**Theorem 2.** Let $\hat{T}$ be the optimal objective value to problem (4–41), and $T^*$ be the optimal objective value to problem (4–51) with the same configuration $(N, L)$ and the same radius of coverage $r$. Then $\hat{T} \leq T^*$.

**Proof.** Let $((\hat{x}^{(l)}_j), (\hat{w}^{(l)}_i), (\hat{z}_k), \hat{T})$ be the optimal solution to problem (4–41). We will prove this theorem by constructing a feasible solution to problem (4–51) with $((\hat{x}^{(l)}_j), (\hat{w}^{(l)}_i), (\hat{Z}_k), \hat{T})$ and showing that under this feasible solution, the objective value of problem (4–51) is $\hat{T}$.

We now define a vector $w$ as follows. For each $l \in L$, we let $w^{(l)}_i = \hat{w}^{(l)}_i$ if $i \in R_l$, and $w^{(l)}_i = 0$ otherwise. Then, we let $q^{(0)}_i = D \cdot d_i$, and $q^{(l)}_i = q^{(l-1)}_i - w^{(l)}_i$ for all $i \in N$. We have the following sequence of assignments for the $q^{(l)}_i$.

$$
q^{(1)}_i = q^{(0)}_i - w^{(1)}_i = D \cdot d_i - w^{(1)}_i \\
q^{(2)}_i = q^{(1)}_i - w^{(2)}_i \\
\vdots \\
q^{(|L|)}_i = q^{(|L|-1)}_i - w^{(|L|)}_i.
$$

By summing up above assignments for all $l \in L$, we have $q^{(|L|)}_i = D \cdot d_i - \sum_l w^{(l)}_i = D \cdot d_i - D \cdot d_i = 0$ by (4–45) and the construction of $w$. Hence, (4–56) is satisfied. Since the configuration and radius of coverage $r$ for problem (4–41) are the same as those for problem (4–51), $N_l(i), i \in N, l \in L$ are the same for both problems. Because of this and by (4–43) and $w^{(l)}_i = q^{(l-1)}_i - q^{(l)}_i$, (4–53) is satisfied. The energy constraints (4–44) and
(4–54) are identical. Hence, given the optimal solution \(((\hat{x}_{ij}^{(i)}), (\hat{w}_i^{(i)}), (\hat{2}_{k}), \hat{T})\) to problem (4–41), we just constructed a feasible solution \(((\hat{x}_{ij}^{(i)}), (q_i^{(i)}), (\hat{2}_{k}), \hat{T})\) to problem (4–51) with the same objective value \(\hat{T}\). Hence, \(T^* \geq \hat{T}\).

In the following theorem, we show that the maximum lifetime of the system is the same for all values of \(D\). Here, the maximum lifetime of the system is equal to the product of \(D\) and the corresponding optimal objective value \(T^*(D)\).

**Theorem 3.** Define \(P(D)\) as the lifetime optimization problem parameterized by the value \(D\), for some fixed network configuration. Let \(T^*(D)\) and \(T^*(D')\) be the optimal objective values for the problem \(P(D)\) and \(P(D')\), respectively. Then, \(T^*(D) \cdot D = T^*(D') \cdot D'\).

**Proof.** Consider the queue-based model.\(^3\) Let \((x^*(D), q^*(D), z^*(D), T^*(D))\) be the optimal solution to the problem \(P(D)\), and let \((x^*(D'), q^*(D'), z^*(D'), T^*(D'))\) be the optimal solution to the problem \(P(D')\).

Let \(x = (\frac{D}{D'})x^*(D'), q = (\frac{D}{D'})q^*(D'), z = z^*(D'), T = (\frac{D}{D'}) T^*(D')\). We want to show that \((x, q, z, T)\) satisfies the constraints (4–53)-(4–60). Since it is obvious that the solution \((x, q, z, T)\) satisfies the constraints (4–57), (4–58), (4–56), (4–59), and (4–60), we focus here on constraints (4–53), (4–54), and (4–55) only. Since the optimal solution \((x^*(D'), q^*(D'), z^*(D'), T^*(D'))\) is feasible to the problem \(P(D')\), it must satisfy constraint (4–53). Next, let us plug \((\frac{D}{D'})x, z,\) and \((\frac{D}{D'})q\) into constraint (4–53) in the places for \(x^*(D'), z^*(D'),\) and \(q^*(D')\), respectively. Then, we have

\[
z_i \sum_{k:i \in N_i(k)} x_{ki}^{(i)} + q_i^{(i-1)} - z_i \sum_{j \in N_i(i)} x_{ij}^{(i)} = q_i^{(i)}.
\] (4–63)

\(^3\) Note that the proof can be adapted to the sub-flow-based model.
If we put \((\frac{D'}{D})x, z,\) and \((\frac{D'}{D})T\) in the places for \(x^*(D'), z^*(D')\) and \(T^*(D')\) on the left hand side of constraint \((4–54)\), we have
\[
\begin{equation}
\left\{ \sum_{l=1}^{|\mathcal{E}|} z_l \left( \sum_{j \in \mathcal{N}(i)} C_{ij} \cdot x_{ij}^{(l)} \left( \frac{D'}{D} \right) + \sum_{k \in \mathcal{N}(i)} \gamma \cdot x_{ki}^{(l)} \left( \frac{D'}{D} \right) \right) \right\} \cdot T \left( \frac{D}{D'} \right).
\end{equation}

(4–64)
\]
After canceling \(D\) and \(D'\), it is easy to see that the new solution \((x, q, z, T)\) satisfies the energy constraint of the problem \(P(D)\).

From the constraint \((4–55)\) for the problem \(P(D')\), we have \(q_i^{*(0)}(D') = D'd_i\). Since \(q = (\frac{D}{D'})q^*(D')\), \(q_i^{*(0)}(D') = q_i^{(0)}(\frac{D'}{D}) = D'd_i (\forall i \in \mathcal{N})\). Therefore we have
\[
q_i^{(0)} = Dd_i.
\]
(4–65)
From above argument, we have shown that new solution \((x, q, z, T)\) is feasible to the problem \(P(D)\). Hence, we have
\[
T^*(D) \geq T = \left( \frac{D'}{D} \right) T^*(D').
\]
Thus, it must be that \(T^*(D)D \geq T^*(D')D'\).

Using a similar argument, we can also conclude that \(T^*(D)D \leq T^*(D')D'\). Hence, \(T^*(D)D\) must equal to \(T^*(D')D'\)

\[4.4\] Experimental Results

In this section, we will present the results from numerical experiments. In particular we have compared the network lifetimes of the following models.

- Static Sink Model (SSM): The stationary sink is located at the origin. We take the performance of this model as the reference for comparison.
- Mobile Sink Model (MSM): The sink can move to several locations to collect data. When the sink is at each location, all sensors participate in the communication, sending and relaying traffic to the sink.
Table 4-1. Experimental parameters and their values

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td># of sensor nodes</td>
<td>{100, 200}</td>
</tr>
<tr>
<td># of possible sink locations</td>
<td>{5, 6, 7, 8, 9, 10, 15, 20, 30, 40}</td>
</tr>
<tr>
<td>path loss exponent (c)</td>
<td>{2.0, 3.0}</td>
</tr>
<tr>
<td>transmission range</td>
<td>{5, 6, 7, 8, 9, 10, 15, 20, 30, 40, 50}</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>10 pJ/bit/m$^2$</td>
</tr>
<tr>
<td>$\beta$</td>
<td>0.0013 pJ/bit/m$^4$</td>
</tr>
<tr>
<td>Initial Energy ($E_i$)</td>
<td>500 J</td>
</tr>
<tr>
<td>Data generation rate ($d_i$)</td>
<td>500 bps</td>
</tr>
</tbody>
</table>

![Figure 4-2. Comparison of lifetimes of MSM and DT-MSM under the various radii of coverage](image)

- Delay-Tolerant Mobile Sink Model (DT-MSM): When the mobile sink is at a stop, a subset of the sensor nodes can participate in the communication. We use the queue-based variant of this model to evaluate the performance.

We have experimented with different parameters extensively, such as the number of nodes, the number of possible sink locations and the parameters for the energy consumption model. Only a small subset of the results are reported here for brevity.

In Table 4-1, we provide the system parameters and their values for the reported experiments in this paper. We adopt the data for the last four parameters from [18]. In all experiments, we use GLPK for solving the linear programming problem.

First, we would like to mention the impact of the radius of coverage of the sink on the performance of the DT-MSM. For this experiment, the positions for 100 nodes and 20 mobile-sink locations are randomly generated ($|V| = 100, |L| = 20$) in a
circular area with radius 25. We use a simple algorithm to find the minimum radius of coverage (denoted by $r_0$): At each sink location, we increase the radius of coverage from 0 simultaneously until the union of all coverages contains all sensor nodes. At that point, we have reached the minimum radius of coverage required to cover all nodes. After that, we increase the radius of coverage in 0.1 increments. The result of the experiment is plotted in Figure 4-2. Note that, in the figure, the lifetime is normalized to the optimal lifetime of the MSM. As shown in the figure, the lifetime of the DT-MSM increases as the radius of coverage increases, which is consistent with Theorem 1. The increase is the sharpest when the radius just exceeds the minimum radius required to cover all nodes. After that, further increase of the radius has a negligible effect. Recall that, when the mobile sink reaches one of the stops, say $l$, only those sensor nodes in the coverage of $l$ (i.e., $R_l$) can communicate. It is generally desirable for $R_l$ to have as few nodes as possible, since this reduces the communication and coordination complexity. The aforementioned behavior of lifetime increase is desirable.

Next, we compare the lifetimes of models under various numbers of the sink locations. The number of nodes is set to 100 or 200, and the path loss exponent $e$ is 2.0. The coverage is set large enough to always cover the entire sensor field. We ran the experiment 100 times for each configuration. The lifetimes of the MSM and DT-MSM
Figure 4-4. Lifetime against the number of sink locations; minimum coverage; \( e = 3.0 \)

are again normalized to the optimal lifetime of the SSM. As shown in Figure 4-3, the lifetime of the MSM is about 100% \( \sim \) 200% greater than that of the SSM. However, the DT-MSM is 200% \( \sim \) 1000% better than the SSM. Moreover, the curves all look linear; the performance gap can grow even larger with more sink locations.

Interestingly, the lifetime of the MSM increases very slowly with the number of sink locations. As explained in [38], in the optimal solution, only a few locations from the set of sink locations are chosen as the true stops for the sink. However, in DT-MSM, the rate of lifetime increase is substantial as \( |L| \) increases. This is because each node can have better and better sink location as \( |L| \) increases, and it is not forced to participate in the communication when the current coverage is not the most favorable for energy saving, even if the node may belong to that coverage. This is not possible in the MSM because no matter where the sink stops, every node must participate in the communication.

We wish to make the following remarks. First, our formulations and reported experiments all use the optimal routing with respect to maximizing the system lifetime. The routing strategy is important for increasing the system lifetime. For instance, based on our experiences, when the shortest path routing is used in the static sink model (results not shown), the lifetime performance is quite inferior to the case of optimal routing. Second, in our model, the locations of the sink stop candidates are randomly
chosen. We expect more performance gain if these candidate locations are carefully selected.

We conduct similar experiments with the same configuration but minimum coverage. The result is shown in Figure 4-4. Although the slope of lifetime increase of the DT-MSM is lowered when compared to the maximum coverage case, the increase pattern is similar. Although a larger set of sink locations increases the network lifetime, it can be undesirable if the sink-traveling time cannot be ignored. The longer traveling time may exceed the delay tolerance level $D$. Therefore, there is a tradeoff between the gain from more sink locations and the delay or other system costs.

In Figure 4-5, we show the lifetimes of the three models under various values for the transmission range. The transmission range determines whether a link exists between a pair of nodes. Whether an existing link is useful or not depends on the radius of coverage: A node cannot use a link to another node if the two nodes are not in the common coverage area.

Both the MSM and the DT-MSM exhibit a sharp lifetime increase when the transmission range is small but increasing. However, as the transmission range becomes large, the lifetime increase comes to a stop for all three models. This is because the energy cost increases with the transmission distance, and hence, in an
optimal solution, a node does not pick far-away nodes as the next-hop neighbors even if the transmission range allows it. The observed fluctuation in the curves is due to statistical fluctuation in the samples of the random network topologies.
CHAPTER 5
A DECOMPOSITION TECHNIQUE FOR DT-MSM

5.1 Overview

We propose an adaptive and potentially decentralized algorithm for the DT-MSM. The distributed routing algorithms are very important in developing a practical routing protocol. Distributed algorithms are generally free from the network scalability issues in several reasons. They do not need to have knowledge about the whole network configurations and they also do not require the central node to compute the routes for all nodes in the network. Lagrange multiplier method solves dual of the primal problem. Dual problem sometimes has a nice structure with which we can decompose the dual problem into several sub-problems. We use a subgradient projection method to solve the dual problem and. A sensor node in our method keeps virtual queue which is a scalar product of the Lagrange multiplier and it is used in solving sub-problems. We propose (a possibly distributed implementable) decentralized algorithms for solving sub-problems. Moreover, we analytically show the our algorithm finds a solution arbitrarily close to the optimal solution of the primal problem. It is verified through the numerical experiments.

5.2 System Model and Problem Formulation

The wireless sensor network is modeled as a directed graph, denoted by $G^0 = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N} = \{1, \ldots, N\}$ is the set of vertices representing the sensor nodes and $\mathcal{A}$ is the set of edges representing the wireless links. Each sensor node $i$ generates data at a constant rate $d_i$ and has an initially energy endowment $E_i$. Let $d(i, j)$ be the Euclidean distance between nodes $i$ and $j$.

Let $N_i$ denote the set of (downstream) neighbor nodes of node $i$, i.e., $N(i) = \{j | (i, j) \in \mathcal{A}\}$. Let $c : \mathcal{A} \rightarrow \mathbb{R}^+$ be a given cost function on the edge set. The cost $c(i, j)$ is the required energy to send a unit of data from node $i$ to $j$ and it is usually a function of the distance between $i$ and $j$. Let $\mathcal{L}$ be the set of the sink locations indexed from 1 through $L$. 

94
As stated earlier, we assume that the traveling time between the sink locations is negligible. With this assumption, the orders of visits to the sink locations does not matter for the optimization problem. Thus, for simplicity of explanation, we assume that the order of visits is given as $1 \rightarrow 2 \rightarrow \cdots \rightarrow L$.

DT-MSM was introduced in [46]. It is suitable for an application that can tolerate a certain amount of delay. In DT-MSM, each node can postpone the transmission of data until the sink is at the location most favorable for extending the system lifetime. However, there is usually a maximum delay that the application can tolerate. This maximum delay tolerance is denoted by $D$. The sink must complete one of its tours from node 1 to $L$ and back to 1 within $D$ time units and then repeats the same tour in the next round.

Since each tour takes $D$ time units, the problem of maximizing the system lifetime is to maximize the number of tours, which is denoted as $T$. The actual lifetime is $T \cdot D$. The decision variables are how much time the sink stays at each location $l \in L$ within each tour, denoted by $t_i$, and what the rate of data transmission from node $i$ to $j$ will be, denoted by $a_{ij}^{(l)}$. Note that in an optimal solution, the mobile sink does not necessarily visit all the sink locations. In that case, we still let the sink visit such a node; but the time of stay is 0.

It turns out, in the problem formulation, $t_i$ and $a_{ij}^{(l)}$ always show up together in the form $t_i a_{ij}^{(l)}$. We can define $x_{ij}^{(l)} = t_i a_{ij}^{(l)}$ to replace $t_i a_{ij}^{(l)}$. Clearly, $x_{ij}^{(l)}$ can be interpreted as the traffic volume on the link $(i, j)$ when the sink is at $l$. We will take the view of traffic volume in the following discussion.

Unlike the non-delay-tolerant mobile sink model, where, regardless where the sink is, every sensor node $i$ must transmit all newly generated data at the data-generation rate $d_i$, a sensor node in DT-MSM can temporarily delay data transmission by storing the data in the local buffer. With delay tolerance and data buffering, there is another flexibility of similar nature that one can take advantage of. When a sink is at location $l$, it may be required to collect data only from nearby sensors. This set of sensors is denoted
by $R_l$, where $R_l \subseteq \mathcal{N}$, and we call it the coverage of sink location $l$. The nodes from outside $R_l$ do not even attempt to communicate. The motivation is that, since the nodes in $R_l$ are close to location $l$, any coordination can be accomplished faster and with less overhead. For instance, the announcement that the sink is at location $l$ only needs to be made to the nodes in $R_l$. We assume $R_l$ is given for each $l$. In a degenerate case, each $R_l$ may be the same as $\mathcal{N}$.

Thus, for each location $l$, there is a graph $G' = (\mathcal{N} \cup \{l\}, \mathcal{A}')$, where $\mathcal{A}' = \{(i, j) \in \mathcal{A} | \, i \in R_l, j \in R_l \cup \{l\}\}$. When the sink is at location $l$, the (downstream) neighbor set of node $i$ is denoted by $N_l(i) = \{j | (i, j) \in \mathcal{A}'\}$.

We create an expanded graph from the graphs $G', l \in \mathcal{L}$. As we make clear shortly, the lifetime maximization problem will be a network flow problem on the expanded graph. In Figure 5-1, we show an example of the expanded graph. Some details about its construction are as follows.

1. Start each column with $G'$, for all $l \in \mathcal{L}$.
2. Relabel node $i$ in $G'$ as $i^{(l)}$.
3. Add a vertex $s$, which represents the sink.
4. For each $l$, replace the edge $(i^{(l)}, l)$ with $(i^{(l)}, s)$ and remove node $l$ from $G'$.
5. For each $i^{(l)}, l = 1, \ldots, L - 1$, add an edge $(i^{(l)}, i^{(l+1)})$.
6. Set the supply at node $i^{(1)}$ to be $Dd_i$ and the demand at node $s$ to be $D \sum_{i \in \mathcal{N}} d_i$.

The cost of each vertical edge (of the form $(i^{(l)}, j^{(l)})$) is assigned as follows:

$$e^{(l)}_{ij} = \begin{cases} 0, & \text{if } \, i \in R_l, j \in R_l \\ c(i, j), & \text{if } \, i, j \in R_l, j \neq l \\ c(i, l), & \text{if } \, i \in R_l, j = l \\ \infty, & \text{otherwise.} \end{cases}$$

The cost of each horizontal edge (of the form of $(i^{(l)}, i^{(l+1)})$) is set to be $0$, because the head and tail of this type of edge are the same physical node and real communication
Figure 5-1. Expanded graph of DT-MSM

does not occur. Let \( x_{ij}^{(l)} \) and \( y_{i}^{(l)} \) be the traffic volume on edge \((i^{(l)}, j^{(l)})\) and \((i^{(l)}, i^{(l+1)})\), respectively.

The flow conservation law at the sensor nodes is as follows.

\[
\begin{align*}
\sum_{j: j \in N_{l}(i)} x_{ji}^{(1)} - \sum_{j: j \in N_{l}(i)} x_{ij}^{(1)} - y_{i}^{(1)} &= -D \cdot d_{i}, \quad \text{if} \ l = 1, \forall i \in \mathcal{N} \\
\sum_{j: j \in N_{l}(i)} x_{ji}^{(l)} - \sum_{j: j \in N_{l}(i)} x_{ij}^{(l)} + y_{i}^{(l-1)} - y_{i}^{(l)} &= 0, \quad \text{if} \ l \in \{2, \ldots, L-1\}, \forall i \in \mathcal{N} \\
\sum_{j: j \in N_{L}(i)} x_{ji}^{(L)} - \sum_{j: j \in N_{L}(i)} x_{ij}^{(L)} + y_{i}^{(L-1)} &= 0, \quad \text{if} \ l = L, \forall i \in \mathcal{N}
\end{align*}
\]

The interpretation is the following. At the beginning of a cycle of length \( D \) time units, node \( i \) has accumulated \( Dd_{i} \) amount of data, which was generated in the previous cycle. This amount of data must be delivered to the sink by the end of the current cycle. \( x_{ij}^{(l)} \) is the amount of data sent on edge \((i, j)\) when the sink is at location \( l \); \( y_{i}^{(l)} \) is the amount
of buffered data (i.e., queue size) at node $i$ just after the sink leaves location $l$. Thus, $y_i^{(l-1)} - y_i^{(l)}$ is the change in the buffered data at node $i$ while the sink is at location $l$.

In addition, at the sink (node $s$), all arrival traffic must be drained. Thus, we have

$$\sum_{l=1}^{L} \sum_{j:s \in N_l(j)} x_{ij}^{(l)} - \sum_{i=1}^{N} Dd_i = 0. \quad (5-3)$$

The problem we address in this paper is to maximize the number of rounds (or cycles), $T$, made by the mobile sink while maintaining the flow conservation $(5-2)$ and $(5-3)$, subject to the energy constraints at the sensor nodes. More precisely, the problem can be written as follows.

$$\max \ T \quad (5-4)$$

s. t. $(5-2), (5-3)$

$$\left(\sum_{i=1}^{L} \sum_{j \in N_l(i)} e_{ij}^{(l)} x_{ij}^{(l)} \right) T \leq E_i, \ \forall i \in \mathcal{N} \quad (5-6)$$

$$x_{ij}^{(l)} \geq 0, \ \forall i \in \mathcal{L}, \forall j \in R_l, \forall j \in N_l(i) \quad (5-7)$$

$$y_i^{(l)} \geq 0, \ \forall i \in \mathcal{N}, \forall l \in \mathcal{L} \quad (5-8)$$

$$T \geq 0. \quad (5-9)$$

Constraint $(5-6)$ means that the total energy expenditure at a node during $T$ rounds should be less than or equal to the node’s initial energy endowment. The above problem can be easily transformed into a linear programming problem, which will be shown next.

5.3 Decomposition by the Lagrange Method

In this section, we illustrate a distributed algorithm to solve the problem defined in Section 5.2. The following is the equivalent linear problem, which is obtained from the maximization problem of $(5-4) - (5-9)$ by replacing $1/T$ with $z$. For convenience, we also define $M = \sum_{i=1}^{N} Dd_i$, $y_i^{(0)} = Dd_i$ and $y_i^{(L)} = 0$ for all $i \in \mathcal{N}$.
\[
\begin{align*}
\min \, z \\
\text{s. t.} \quad & \sum_{l=1}^{L} \sum_{j \in N_l(i)} e_{ij}^{(l)} x_{ij}^{(l)} \leq zE_i, \quad \forall i \in \mathcal{N} \tag{5-10} \\
& \left\{ \begin{aligned} 
& \sum_{j: i \in N_l(j)} x_{ji}^{(l)} - \sum_{j \in N_l(i)} x_{ij}^{(l)} + y_i^{(l-1)} - y_i^{(l)} = 0, \quad \forall l \in \mathcal{L}, \forall i \in \mathcal{N} \\
& \sum_{l=1}^{L} \sum_{j: s \in N_l(j)} x_{js}^{(l)} - M = 0 
\end{aligned} \right. \tag{5-11} \\
& y_{i}^{(0)} = Dd_i, \quad y_{i}^{(L)} = 0, \quad \forall i \in \mathcal{N} \tag{5-12} \\
& x_{ij}^{(l)} \geq 0, \quad \forall l \in \mathcal{L}, \forall i \in \mathcal{R}_l, \forall j \in N_l(i) \tag{5-13} \\
& y_{i}^{(l)} \geq 0, \quad \forall i \in \mathcal{N}, \forall l \in \{2, 3, \ldots, L-1\} \tag{5-14} \\
& z \geq 0. \tag{5-15}
\end{align*}
\]

Note that \(M\) is an upper bound of any traffic volume, a term that also includes the buffered data. We will use the terms \textit{flow} and \textit{volume} interchangeably. The new formulation has the interpretation that it minimizes the maximum energy consumption among all nodes in a single round, normalized with respect to \(E_i, (\sum_{l=1}^{L} \sum_{j \in N_l(i)} e_{ij}^{(l)} x_{ij}^{(l)} / E_i)\) while satisfying flow conservation.

Instead of tackling the problem (5–10) - (5–16), we would like to consider the following problem, where equalities in flow conservations are changed into less-than-or-equal (\(\leq\)) inequalities. Later we will prove that these different formulations are, in fact, equivalent.
\[
\min z \\
\text{s. t. } \sum_{l=1}^{L} \sum_{j \in N(l)} e_{ij}^{(l)} x_{ij}^{(l)} \leq zE_i, \quad \forall i \in \mathcal{N} \\
\begin{cases}
\sum_{j: i \in N(j)} x_{ij}^{(l)} - \sum_{j \in N(i)} x_{ij}^{(l)} + y_i^{(l-1)} - y_i^{(l)} \leq 0, \quad \forall l \in \mathcal{L}, \forall i \in \mathcal{N} \\
\sum_{l=1}^{L} \sum_{j: i \in N(j)} x_{ij}^{(l)} - M \leq 0 \\
y_i^{(0)} = Dd_i, \quad y_i^{(L)} = 0, \quad \forall i \in \mathcal{N} \\
x_{ij}^{(l)} \geq 0, \quad \forall l \in \mathcal{L}, \forall i \in \mathcal{R}, \forall j \in N(i) \\
y_i^{(l)} \geq 0, \quad \forall i \in \mathcal{N}, \forall l \in \{2, 3, \ldots, L-1\} \\
z \geq 0.
\end{cases}
\] (5–17, 5–18, 5–19, 5–20, 5–21, 5–22, 5–23)

We use an ordered-pair notation \((i, l)\) to denote the index of the flow constraint (5–19) of the node (except the sink) \(i\) when the sink is at \(l\). For a given \((x, y)\), if the flow constraint \((i, l)\) satisfies equality, we say that constraint \((i, l)\) is binding. (Some authors may use the term tight or active instead.) That is, the binding constraint \((i, l)\) implies that
\[
\sum_{j: i \in N(j)} x_{ij}^{(l)} = \sum_{j \in N(i)} x_{ij}^{(l)} + y_i^{(l-1)} - y_i^{(l)} = 0.
\] (5–24)

**Lemma 1.** \((x, y)\) is feasible to constraint (5–19) if only if all constraint (5–19) for \((x, y)\) are binding.

**Proof.** It is clear that any \((x, y)\) satisfies equality in constraint (5–19) is feasible solution. To prove the opposite direction of the lemma, let's suppose \((x, y)\) is feasible and some constraint of (5–19) are not binding. Let \(K\) be the set of an ordered pair \((i, l)\), such that
\[
\sum_{j: i \in N(j)} x_{ij}^{(l)} = \sum_{j \in N(i)} x_{ij}^{(l)} + y_i^{(l-1)} - y_i^{(l)} < 0.
\] (5–24)

We may write a unbinding constraint \((i, l)\) as follows.
\[
\sum_{j: i \in N(j)} x_{ij}^{(l)} - \sum_{j \in N(i)} x_{ij}^{(l)} + y_i^{(l-1)} - y_i^{(l)} = 0 - \sigma_i^{(l)}.
\] (5–24)
where $\sigma_i^{(l)} > 0$. Now, after summing constraint (5–19) over $i \in \mathcal{N}$, $l \in \mathcal{L}$, we get

$$
\sum_{l=1}^{L} \sum_{i=1}^{N} \left( \sum_{j \in N(i)} x_{ji}^{(l)} - \sum_{j \in N(i)} x_{ij}^{(l)} + y_i^{(l-1)} - y_i^{(l)} \right) = 0 - \sum_{(u,v) \in K} \sigma_u^{(v)} \tag{5–25}
$$

After canceling common terms in LHS of (5–25), we have

$$
\sum_{l=1}^{L} \left( - \sum_{j \in N(j)} x_{js}^{(l)} \right) + \sum_{i=1}^{N} y_i^{(0)} - \sum_{i=1}^{N} y_i^{(L)} = - \sum_{l=1}^{L} \sum_{j \in N(j)} x_{js}^{(l)} + M \tag{5–26}
$$

Recall that $y_i^{(0)} = d_i$ and $y_i^{(L)} = 0$ for $i \in \mathcal{N}$. However, since $\sum_{(u,v) \in K} \sigma_u^{(v)} > 0$, the RHS of (5–25) is negative. Hence we have the following inequality.

$$
- \sum_{l=1}^{L} \sum_{j \in N(j)} x_{js}^{(l)} + M < 0
$$

$$
\iff \sum_{l=1}^{L} \sum_{j \in N(j)} x_{js}^{(l)} > M \tag{5–27}
$$

This is a contradiction to the second part of constraint (5–19).

Therefore, for $(x, y)$ to be feasible, all constraints in the first part of (5–19) must be binding. And as a consequence, $\sum_{l=1}^{L} \sum_{j \in N(j)} x_{js}^{(l)} = M$.

**Theorem 5.1.** The problems as formulated in (5–10) - (5–16) and (5–17) - (5–23) are equivalent.

**Proof.** This is a direct consequence from the Lemma 1.

Next, we show that the removal of the second constraint in (5–19) does not affect the optimal objective value. The modified formulation is the final one which we
eventually solve and is as follows.

\[
\begin{align*}
\min & \quad z \\
\text{s. t.} & \quad \sum_{l=1}^{L} \sum_{j \in N_l(i)} e_{ij}^{(l)} x_{ij}^{(l)} \leq zE_i, \quad \forall i \in \mathcal{N} \\
& \quad \sum_{j : i \in N_l(j)} x_{ij}^{(l)} - \sum_{j \in N_l(i)} x_{ji}^{(l)} + y_i^{(l)} - y^{(l-1)}_i \leq 0, \quad \forall l \in \mathcal{L}, \forall i \in \mathcal{N} \\
& \quad y_i^{(0)} = Dd_i, \quad y_i^{(L)} = 0, \quad \forall i \in \mathcal{N} \\
& \quad x_{ij}^{(l)} \geq 0, \quad \forall l \in \mathcal{L}, \forall i \in R_l, \forall j \in N_l(i) \\
& \quad y_i^{(l)} \geq 0, \quad \forall i \in \mathcal{N}, \forall l \in \{2, 3, \ldots, L-1\} \\
& \quad z \geq 0.
\end{align*}
\] (5–28)

Let be \( \hat{z} \) and \( z^* \) be the optimal value of the formulation (5–28) - (5–34) and formulation (5–17) - (5–23) respectively.

**Theorem 5.2.** Let be \( \hat{z} \) and \( z^* \) be the optimal value of the formulation (5–28) - (5–34) and formulation (5–17) - (5–23) respectively. Then, \( \hat{z} = z^* \).

**Proof.** It is obvious that \( \hat{z} \leq z^* \), because the feasible solution set of the formulation (5–28) - (5–34) is bigger than the feasible set of (5–17) - (5–23).

Since Theorem 5.1 says that the formulations (5–10) - (5–16) and (5–17) - (5–23) are equivalent, we compare the optimal objective value of (5–28) - (5–34) to that of (5–10) - (5–16). Let \((z^*, x^*, y^*)\) be one of optimal solution of the formulation (5–10) - (5–16). The inequality constraint (5–30) can be transformed into equality constraint by adding slack variables.

\[
\begin{align*}
\sum_{j : i \in N_l(j)} x_{ij}^{(l)} - \sum_{j \in N_l(i)} x_{ji}^{(l)} + y_i^{(l)} - y^{(l-1)}_i & \geq 0 \\
\iff \sum_{j : i \in N_l(j)} x_{ij}^{(l)} - \sum_{j \in N_l(i)} x_{ji}^{(l)} + y_i^{(l)} - y^{(l-1)}_i & = s_i^l,
\end{align*}
\] (5–35)
where $s_i^l$ is a nonnegative slack variable for $(i, l)$. Let $(\hat{z}, \hat{x}, \hat{y}, \hat{s})$ be the optimal solution of the formulation (5–28) - (5–34) with constraint (5–30) replaced by the above equality constraint with slack variables.

Suppose that $\hat{z} < z^\ast$. From a given optimal solution $(\hat{z}, \hat{x}, \hat{y}, \hat{s})$ that satisfies (5–36) we can construct a path flow. Let $P_i^l$ be the set of path flows who have the $(i, l)$ as the origin. In the path flow formulation, (5–36) can be rephrase as follows.

$$\sum_{p \in P_i^l} \hat{f}(p) = d_i^l + s_i^l, \quad (5–37)$$

where $\hat{f}(p)$ is the amount of flow along the path $p$ and $d_i^l = Dd_i$ if $l = 1$, otherwise $d_i^l = 0$.

Now we construct another path flow $f$ from $\hat{f}$ by the following flow assignment.

$$f(p) = \frac{\hat{f}(p)(1 - s_i^l/\sum \hat{f}(p))}{(5–38)}$$

Here, if $s_i^l = 0$, then inequality constraint becomes equality constraint. Thus, we suppose that $s_i^l > 0$. The above assignment satisfies the following property.

$$\sum_{p \in P_i^l} f(p) = d_i^l \quad (5–39)$$

If we construct edge flow from the new path flow assignment, $f$, the edge flow assignment satisfies the flow conservation constraint as (5–12).

The energy constraint (5–29) can be also represented as a path flow as follows.

$$\sum_{l=1}^L e_{ij}^{(l)} \left( \sum_{p \cdot (i,j) \in p, p \in P_i^l} \hat{f}(p) \right) \leq \hat{z}E_i \quad (5–40)$$

Putting $\hat{f}(p) = f(p) + \hat{f}(p)(s_i^l/\sum \hat{f}(p))$ into the energy constraint, we get

$$\sum_{l=1}^L e_{ij}^{(l)} \left( \sum_{p \cdot (i,j) \in p, p \in P_i^l} f(p) \right) \leq \hat{z}E_i - \sum_{l=1}^L e_{ij}^{(l)} \left( \hat{f}(p)(s_i^l/\sum \hat{f}(p)) \right) \quad (5–41)$$

103
Therefore we have the following inequality.

$$\sum_{l=1}^{L} e_{ij}^{(l)}(\sum_{p,(i,j)\in p, p\in P_i} f(p)) < \hat{z}E_i - \sum_{l=1}^{L} e_{ij}^{(l)}(\hat{f}(p)(s_i/\sum \hat{f}(p)))$$

(5–42)

Hence, our new path flow (or equivalent edge flow) can have an objective value, say \(z\), such that \(z < \hat{z}\), which is the contradiction to our premise.

Therefore, \(\hat{z} = z^*\).  

\[\square\]

Now, we turn our attention to deriving the algorithm by Lagrangian relaxation.

Let \(\pi_i^{(l)}\) be the Lagrange multipliers associated with the constraints in (5–30). The Lagrangian function of (5–28) is

$$L(z, x, y, \pi) = z + \sum_{l=1}^{L} \sum_{i=1}^{N} \sum_{(i,j)\in A} \pi_i^{(l)} e_{ij}^{(l)} - \sum_{j\in N(i)} x_{ij}^{(l)} - y_i^{(l-1)} - y_i^{(l)},$$

(5–43)

where \(\pi = (\pi_i^{(l)})\), over \(i \in N, l \in L\).

After grouping the terms based on the primal variables \(x\) and \(y\), we get

$$L(z, x, y, \pi) = z + \sum_{l=1}^{L} \sum_{(i,j)\in A} (\pi_j^{(l)} - \pi_i^{(l)}) x_{ij}^{(l)} +$$

$$\sum_{i=1}^{N} \sum_{l=1}^{L-1} (\pi_i^{(l+1)} - \pi_i^{(l)}) y_i^{(l)} + D \sum_{i=1}^{N} \pi_i^{(1)} d_i.$$  

(5–44)

Since the last term of (5–44) is a constant for a given \(\pi\), we can ignore it when deriving the dual function. The Lagrangian dual function \(\theta(\pi)\) is now given by

$$\theta(\pi) = \min L(z, x, y, \pi)$$

(5–45)

s. t. \(\sum_{l=1}^{L} \sum_{(i,j)\in A} x_{ij}^{(l)} e_{ij}^{(l)} - zE_i \leq 0, \quad \forall i \in N\)  

(5–46)

\(x_{ij}^{(l)} \geq 0, \quad \forall l \in L, \forall i \in N, \forall j \in N_i(i)\)  

(5–47)

\(0 \leq y_i^{(l)} \leq M, \quad 1 \leq l \leq L - 1, \forall i \in N\)  

(5–48)

\(z \geq 0\).  

(5–49)
We can decompose the problem (5–45) - (5–49) into the following two subproblems.

\[ S_1 : \min \sum_{i=1}^{N} \sum_{l=1}^{L-1} (\pi_{i,l+1}^{(l)} - \pi_{i,l}^{(l)}) y_{i,l}^{(l)} \]
\[ \text{s.t.} \quad 0 \leq y_{i,l}^{(l)} \leq M, \quad \forall i \in \mathcal{N}, 1 \leq l \leq L - 1. \]  
(5–50)

\[ S_2 : \min \left\{ z + \sum_{l=1}^{L} \sum_{(i,j) \in \mathcal{A}_l} (\pi_{j,l}^{(l)} - \pi_{i,l}^{(l)}) x_{i,j}^{(l)} \right\} \]
\[ \text{s.t.} \quad 0 \leq x_{i,j}^{(l)} \leq M, \quad \forall i \in \mathcal{N}, \forall l \in \mathcal{L}, \forall j \in N_l(i) \]
\[ \sum_{l=1}^{L} \sum_{j \in N_l(i)} e_{i,j}^{(l)} x_{i,j}^{(l)} - z \leq 0, \quad \forall i \in \mathcal{N} \]
\[ z \geq 0. \]  
(5–51)

Note that we add the upper bound \( M \) to the flow variables \( x \) and \( y \).

5.3.1 Algorithms for Subproblems

The solution for the subproblem \( S_1 \) is very obvious. If \((\pi_{i,l+1}^{(l)} - \pi_{i,l}^{(l)})\) is negative, then we assign the largest value \((= M)\) to the variable \( y_{i,l}^{(l)} \). Otherwise, \( y_{i,l}^{(l)} \) should be 0. This is shown in Algorithm 5-1.

---

**Algorithm 5-1 Solution for \( S_1 \)**

if \((\pi_{i,l+1}^{(l)} - \pi_{i,l}^{(l)}) \geq 0\) then

\[ y_{i,l}^{(l)} \leftarrow 0 \]

else

\[ y_{i,l}^{(l)} \leftarrow M \]

end if

---

Algorithm 5-1 can be implemented in a distributed and local manner. The value of \( y_{i,l}^{(l)} \) can be decided locally in the sensor node \( i \). Also, node \( i \) only needs to have the knowledge of \( \pi_j \) from its neighbor node set \( N_l(i) \) for all \( l \in \mathcal{L} \).
Now, we turn to the subproblem $S_2$. For the ease of exposition, we intentionally change the original $S_2$ to the maximization problem. Suppose, for a moment, $z$ is fixed at a certain point $\bar{z}$ and define $f(\bar{z})$ as follows:

$$f(\bar{z}) = \max \left\{ -\bar{z} + \sum_{i=1}^{N} \sum_{l=1}^{L} \sum_{j \in N_l(i)} (\pi_i^{(l)} - \pi_j^{(l)}) x_{ij}^{(l)} \right\}$$

s.t.  $0 \leq x_{ij}^{(l)} \leq M$, $\forall i \in N$, $\forall l \in L$, $\forall j \in N_l(i)$

$$\sum_{l=1}^{L} \sum_{j \in N_l(i)} e_{ij}^{(l)} x_{ij}^{(l)} \leq \bar{z} E_i, \quad \forall i \in N.$$

Let $f_i(\bar{z}) = \max \{ \sum_{l=1}^{L} \sum_{j \in N_l(i)} (\pi_i^{(l)} - \pi_j^{(l)}) x_{ij}^{(l)} \}$ subject to $x_{ij}^{(l)} \in [0, M]$ for all $l \in L, j \in N_l(i)$ and $\sum_{l=1}^{L} \sum_{j \in N_l(i)} e_{ij}^{(l)} x_{ij}^{(l)} \leq \bar{z} E_i$. Then, $f(\bar{z}) = -\bar{z} + \sum_{i=1}^{N} f_i(\bar{z})$.

The last equality states that the maximization problem for finding $f(\bar{z})$ can be further decomposed into smaller maximization problems in which each node $i$ tries to find $f_i(\bar{z})$. The problem to find $f_i(\bar{z})$ at each node $i$ corresponds to the fractional knapsack problem, which has a polynomial time greedy algorithm [20].

Suppose there are $N$ knapsacks and knapsack $i$ has a weight capacity of $\bar{z} E_i$. For knapsack $i$, we can pack items, denoted by $(i, l, j)$ such that $j \in \{ j | (i, j) \in A_l, \forall l \in L \}$. We assume that each item can be infinitely divisible. Suppose there is a reward $(\pi_i^{(l)} - \pi_j^{(l)})$ when we pack a unit of item $(i, l, j)$. Also, consider $e_{ij}^{(l)}$ as the weight of one unit of item $(i, l, j)$. Again, recall that the maximum available amount of an item is limited by $M$. The profit of an item $(i, l, j)$ is defined as the reward per unit weight of that item, or $(\pi_i^{(l)} - \pi_j^{(l)}) / e_{ij}^{(l)}$. The fractional knapsack problem is to select the items to pack subject to the capacity constraint of the knapsack such that the total reward is maximized.

The solution is straightforward. We greedily pack the most profitable item among the remaining ones until that item is exhausted or the knapsack capacity is reached. This operation is repeated until all profitable (that is, with positive profit) items are packed or the knapsack is full. Details are listed in Algorithm 5-2. In some cases, the knapsack
may not be fully packed in the optimal solution when all items with positive rewards are packed.

Note that this algorithm can be implemented in a distributed and local manner. A node $i$ only requires the knowledge of $\pi_l^{(i)}$ of each neighbor $j \in N_l(i)$, for all $l$.

**Algorithm 5-2** Fractional Knapsack ($\bar{z}$) for Finding $f_i(\bar{z})$

```plaintext
sort $(i, l, j)$ in the decreasing order of $(\pi_l^{(i)} - \pi_l^{(j)})/e_l^{(i)}$
$(i, l, j) \leftarrow$ the first element in the sorted list
$U \leftarrow \bar{z}E_i$
while $U > 0$ do
  if $(\pi_l^{(i)} - \pi_l^{(j)} < 0)$ then
    break
  else if $(U - Me_l^{(i)}) < 0$ then
    $x_l^{(i)} \leftarrow U / e_l^{(i)}$
    break
  else
    $x_l^{(i)} \leftarrow M$
  end if
$(i, l, j) \leftarrow$ the next element of the sorted list
end while
```

The tricky part in solving subproblem $S_2$ is how to choose the right value for $z$, so that the overall objective function, $-z + \sum_{i=1}^{N} f_i(z)$, is maximized. Note that $f_i(z)$ is a concave, nondecreasing, and piecewise linear function of $z$ and $f(z)$ is a concave and piecewise linear function of $z$. We will search for an optimal solution by increasing $z$ and we only need to care about those points what mark the beginning or end of a linear segment. Let $z^*$ be the first optimal solution encountered in the search. To the left of $z^*$, the function $f(z)$ must be increasing (except the trivial case where $f(z)$ is identically 0, which can be discovered separately); to the right, the function is non-increasing. This is also a sufficient condition for optimality.

Consider the right derivatives of these piecewise linear functions. Note that the (right) derivative of the function $f(z)$ can be written as $f'(z) = \sum_{i \in \mathcal{N}} f_i(z) - 1$. The
optimality condition is
\[
\begin{align*}
\sum_{i \in N} f_i(z) & > 1, \quad z < z^* \\
\sum_{i \in N} f_i(z) & \leq 1, \quad z > z^*.
\end{align*}
\]
(5–52)

Also note that \( f_i(z) \) changes only when one of the \( f_j(z) \) changes. From Algorithm 5-2, we see that, for each \( i \), \( f_i(z) \) changes only when we select the most profitable item in the list of remaining items. Suppose item \((i, j, l)\) is selected. The new \( f_i(z) \) is given by
\[ f_i(z) = (\pi_i^{(l)} - \pi_j^{(l)})/e_y^{(l)} \]
Furthermore, the next time when \( f_i(z) \) changes again is when \( z \) is incremented by \( Me_y^{(l)}/E_i \).

To summarize, the procedure for searching \( z^* \) is to keep track of the sequence of points where \( f_i(z) \) changes, which requires keeping track of the sequence of points where \( f_i(z) \) changes, for each \( i \). Consider a fixed \( i \). Suppose \((\pi_i^{(l)} - \pi_j^{(l)})/e_y^{(l)}\) is sorted in decreasing order and suppose any item \((i, l, j)\) with \((\pi_i^{(l)} - \pi_j^{(l)}) \leq 0\) is discarded.

Starting with \( z_0 = 0 \), we can generate a sequence \( z_k = z_{k-1} + Me_y^{(l)}/E_i \) iteratively, where \((i, j, l)\) used in the update to get \( z_k \) is the \( k \)th item in the list. Then, \( f_i(z) \) can change only at each of the points \( z_k \). Algorithm 5-3 describes an implementation of the above idea, as well as the solution to the subproblem \( S_2 \). For each \( i \), the array \( P_i[\] \) records the sequence of \( z_k \) and \( f_i(z_k) \).

**Algorithm 5-3 Solution for \( S_2 \)**

```plaintext
for each \( i \in N \) do
    sort \((i, l, j)\) in decreasing order of \((\pi_i^{(l)} - \pi_j^{(l)})/e_y^{(l)}\)
    discard any item \((i, l, j)\) if \((\pi_i^{(l)} - \pi_j^{(l)}) \leq 0\)
    \(k \leftarrow 0; z_k \leftarrow 0\)
    for each of \((i, l, j)\) in the sorted list do
        \(z_k \leftarrow z_k + Me_y^{(l)}/E_i\)
        \(P_i[k] \leftarrow (z_k, (\pi_i^{(l)} - \pi_j^{(l)})/e_y^{(l)})\)
        \(k \leftarrow k + 1\)
    end for
end for
find \( z^* \) which satisfies (5–52) by searching \((P_i)_{i \in N}\)
for each node \( i \) applies Algorithm 5-2 with \( z^* \)
```

108
The description of Algorithm 5-3 is ambiguous on how to make the algorithm distributed and (partially) local. There are different possibilities. In one version, each node $i$ executes the outer for-loop in parallel, and for that, it requires only local information. After the one-dimensional array $P_i[,]$ is computed, node $i$ can broadcast this array to all other nodes. After a node collects the complete two-dimensional array, it can compute $z^*$ by itself. Another possibility is that each node $i$ sends the array $P_i[,]$ to the sink; the sink computes $z^*$ and sends it back to every node, which goes on to execute Algorithm 5-2.

5.3.2 Main Algorithm

We now assume that the system operates in a time-slotted way. The Lagrange dual problem of (5–28) is

\[
\text{Dual: } \max \theta(\pi) \\
\text{s.t } \pi \geq 0
\] (5–53)

Consider the subgradient projection method to solve problem (5–53). The update of $\pi$ at each iteration is given by the following equations.

\[
\pi_i^{(l)}(k+1) = [\pi_i^{(l)}(k) - \delta(\sum_{j \in \mathcal{N}(i)} x_j^{(l)}(k) + y_j^{(l)}(k) - \\
\sum_{j: i \in \mathcal{N}(j)} x_j^{(l)}(k) - y_j^{(l-1)}(k))]^+, \quad \forall i \in \mathcal{N}, \forall l \in \mathcal{L}
\] (5–54)

where $[b]^+ = \max\{0, b\}$ and $\delta(>0)$ is a sufficiently small number.

Our algorithm is motivated by the subgradient algorithm, but not exactly identical. The standard convergence results of the subgradient algorithm do not apply. In Section 5.4, we will use a different analytical framework to prove the optimality of the algorithm.

Let $\delta q_i^{(l)}(k) = \pi_i^{(l)}(k)$. We have the following algorithm. For technical reasons, the upper bound of the flow variables is modified from $M$ to $M(\epsilon) \triangleq M + NL\epsilon$, where $\epsilon$ is a small positive value.
Main Algorithm

\[
y(k) = \arg \min_{y(l) \in [0, M(\epsilon)], \ i \in \mathcal{N}, 1 \leq l \leq L-1} \left\{ \sum_{i=1}^{N} \sum_{l=1}^{L-1} (q_i(l+1) - q_i(l))y_i(l) \right\}
\]  

(5–55)

\[
(z(k), x(k)) = \arg \min_{x(l) \in [0, M(\epsilon)], \ i \in \mathcal{N}, \ j \in \mathcal{L}, \ i \in \mathcal{N}(j)} \left\{ \frac{Z}{\delta} + \sum_{l=1}^{L} \sum_{(i,j) \in \mathcal{A}} (q_j^{(l)}(k) - q_i^{(l)}(k))x_{ij}^{(l)} \right\}
\]

(5–56)

\[
q_i^{(l)}(k+1) = [q_i^{(l)}(k) - \left( \sum_{j \in \mathcal{N}(i)} x_{ij}^{(l)}(k) + y_i^{(l)}(k) - \sum_{j:i \in \mathcal{N}(j)} x_{ji}^{(l)}(k) - y_i^{(l-1)}(k) \right)]^+, \quad \forall i \in \mathcal{N}, \forall l \in \mathcal{L}
\]

(5–57)

\[
q_s(k+1) = 0.
\]

(5–58)

Note that (5–55) and (5–56) are solved by Algorithm 5-1 and 5-3, respectively, with suitable modification of the notations. We can consider \(q_i^{(l)}\) as a virtual queue at node \(i^{(l)}\) in Figure 5-1, and (5–57) can be understood as the queue dynamic. That is, the queue length of the node at time slot \(k+1\) is equal to the queue length at time slot \(k\) plus the new arrivals \(\sum_{j:i \in \mathcal{N}(j)} x_{ij}^{(l)} + y_i^{(l-1)}\) and minus the total service \(\sum_{j:i \in \mathcal{N}(j)} x_{ji}^{(l)} + y_i^{(l)}\). Since \(q_s^{(k+1)} = 0\), all flow reaching the sink should be drained out.

5.4 Performance Analysis

In this section, we show that our algorithm converges to the optimal solution and the virtual queues are bounded, both in the long-run average sense. The analytical technique is in part borrowed from [29].
We first define an \(\epsilon\)-perturbed problem, which will be used later. Here, \(\epsilon\) is the same small positive constant in the definition of \(M(\epsilon)\).

\[
\min_z \quad z \\
\text{s. t.} \quad \sum_{l=1}^{L} \sum_{j \in N(i)} e^{(l)}_{ij} x^{(l)}_{ij} \leq zE_i, \quad \forall i \in \mathcal{N} \\
\sum_{j \in N(i)} x^{(l)}_{ij} = \sum_{j \in N(i)} x^{(l-1)}_{ij} + y^{(l)}_{i} - y^{(l-1)}_{i} = -\epsilon, \quad \forall l \in \mathcal{L}, \forall i \in \mathcal{N} \\
\sum_{l=1}^{L} \sum_{j \in N(i)} x^{(l)}_{js} = M + NL\epsilon \\
y^{(0)}_{i} = Dd_i, \quad y^{(L)}_{i} = 0, \quad \forall i \in \mathcal{N}.
\]

(5–62)

The usual non-negativity constraints of the variables are still required. In the above problem, we inject extra supply in the amount \(\epsilon\) at each node \(i^{(l)}, i \in \mathcal{N}, l \in \mathcal{L}\). The demand at the destination node \(s\) is now \(M + NL\epsilon\), so that there exists a feasible flow.  

Remark: The crucial fact is that any feasible flow to the perturbed problem still satisfies 
\(y^{(l)}_{i} \in [0, M(\epsilon)], i \in \mathcal{N}, 1 \leq l \leq L - 1; x^{(l)}_{ij} \in [0, M(\epsilon)], i \in \mathcal{N}, l \in \mathcal{L}, j \in N(i); \)
\[
\sum_{l=1}^{L} \sum_{j \in N(i)} e^{(l)}_{ij} x^{(l)}_{ij} \leq zE_i, i \in \mathcal{N}; \text{ and } z \geq 0.
\]
Hence, the vector \(y\) is feasible to the optimization problem in (5–55), and \((z, x)\) is feasible to (5–56).

In the following lemma, we discuss the properties of the optimal objective value function of the \(\epsilon\)-perturbed problem. For simplicity of discussion, we consider the standard linear programming problem:

\[
(P) \min c^T x \\
\text{s.t.} \quad Ax = d \\
x \geq 0.
\]

(5–63)
In the problem \((P(\epsilon))\) below, the right hand side of \((P)\) is perturbed by \(\epsilon\) along the direction \(\Delta d\).

\[
(P(\epsilon)) \min \ c^T x \\
\text{s.t. } Ax = d + \epsilon \Delta d \\
x \geq 0.
\]

(5–64)

Let \(f^*\) and \(f^* (\epsilon)\) be the optimal objective values for problem \((P)\) and \((P(\epsilon))\), respectively.

**Lemma 2.** \(f^* (\epsilon)\) is continuous, convex, and piecewise linear.

**Proof.** Let \((D)\) be the dual linear problem of \((P)\).

\[
(D) \max \ d^T \pi \\
\text{s.t. } A^T \pi \leq c,
\]

where \(\pi\) is the dual variable associated with the constraints of \((P)\). Similarly, the dual problem for \((P(\epsilon))\) is defined as

\[
(D(\epsilon)) \max \ (d + \epsilon \Delta d)^T \pi \\
\text{s.t. } A^T \pi \leq c.
\]

Since the constraints for \((D(\epsilon))\) are not changed from \((D)\), the feasible solution set of \((D(\epsilon))\) is the same as that of \((D)\). Let \(D\) be this feasible solution set, which does not depend on \(\epsilon\). By the strong duality [9], we know that \(f^* (\epsilon) = \max\{(d + \epsilon \Delta d)^T \pi | \pi \in D\}\).

For any \(\epsilon\), the optimal objective value can be obtained at one of the extreme points of \(D\). Let \(\Omega\) be the set of extreme points of \(D\), which is a finite set. Hence, \(f^*(\epsilon) = \max\{(d + \epsilon \Delta d)^T \pi | \pi \in \Omega\}\). For each \(\pi \in \Omega\), we have

\[
(d + \epsilon \Delta d)^T \pi = d^T \pi + \epsilon \Delta d^T \pi,
\]

(5–65)
which is a linear function of $\epsilon$. Therefore, $f^*(\epsilon) = \max\{(d + \epsilon \Delta d)^T \pi | \pi \in \mathcal{D}\}$ is a continuous, convex, and piecewise linear function \[10\].

**Theorem 5.3.** Let $z^*(\epsilon)$ be the optimal value for the problem in (5–59) - (5–62) and $z^*$ be the optimal value for the unperturbed problem. Then, $z^*(\epsilon) \to z^*$ as $\epsilon \to 0$.

**Proof.** This is a direct consequence of Lemma 2.

Next, we want to prove our algorithm converges to the optimal objective value in the time average sense. Let us define a Lyapunov function of the queues by $V(q) = \sum_{i \in N} \sum_{l \in L} (q_i(l)_{k+1})^2$. Let $\Delta(k) \triangleq V(q(k+1)) - V(q(k))$.

**Lemma 3.** There exists a positive constant $B$ such that for any small positive $\epsilon$ and $\delta$, the following condition holds for any time slot $k$ and for any $q(k)$,

$$
\Delta(k) + \frac{2}{\delta} z(k) \leq B + \frac{2}{\delta} \hat{z}(\epsilon) - 2\epsilon \sum_{i \in L} \sum_{i \in N} q_i(l),
$$

where $\hat{z}(\epsilon)$ is part of an optimal solution of the $\epsilon$-perturbed problem.

**Proof.** By squaring (5–57) and arranging it, we get

$$(q_i(l)_{k+1})^2 - (q_i(l)_k)^2 \leq g^2(i, l, k) - 2q_i(l)_k g(i, l, k),$$

where $g(i, l, k) = \sum_{j \in N(i)} x_{ij}(l)_k - \sum_{j \in N(l)} x_{ji}(l)(i, k) + y_{ij}(l)_k - y_{ij}(l-1)_k$. Note that $g(i, l, k) \leq NM$ because $\sum_{j \in N(i)} x_{ij}(l)_k \leq (N-1)M$ and $y_i(l)_k \leq M$ for all $k$. 

\[113\]
Summing the above inequality over all $i$ and $l$, we have

$$
\sum_{i=1}^{L} \sum_{l=1}^{N} \left[ \left( q_{i}^{(l)}(k+1) \right)^{2} - \left( q_{i}^{(l)}(k) \right)^{2} \right] \leq \sum_{l,i} g_{i,l,k}^{2} - 2 \sum_{l,i} q_{i}^{(l)}(k) g_{i,l,k} \\
\leq LN^{3}M^{2} + 2 \sum_{l,i} q_{i}^{(l)}(k) \left( - \sum_{i \in M(i)} x_{q}^{(l)}(k) + \sum_{j:i \in M(j)} x_{q}^{(l)}(k) - y_{q}^{(l)}(k) + y_{q}^{(l-1)}(k) \right) \\
= B - 2 \sum_{l=1}^{L} \sum_{(j,s) \in A'} q_{j}^{(l)}(k) x_{js}^{(l)}(k) + 2 \sum_{l=1}^{L} \sum_{(j,s) \in A' : j \neq s} \left( q_{j}^{(l)}(k) - q_{i}^{(l)}(k) \right) x_{q}^{(l)}(k) \\
+ 2 \sum_{l=1}^{N} \sum_{i=1}^{L-1} \left( q_{i}^{(l+1)}(k) - q_{i}^{(l)}(k) \right) y_{i}^{(l)}(k) + 2 \sum_{i=1}^{N} q_{i}^{(1)}(k) y_{i}^{(0)}(k),
$$

(5–67)

where $B \triangleq LN^{3}M^{2}$. (5–67) can be obtained by regrouping the terms based on variables $x$ and $y$. Note that the third term in the last equality excludes links to the sink. Adding $2q_{s}(\sum_{l,i} x_{js}^{(l)}(k)) = 0$ to (5–67), we have

$$
\Delta(k) \leq B + 2 \sum_{i=1}^{N} q_{i}^{(1)}(k) D_{d_{i}} + 2 \sum_{l=1}^{L} \sum_{(i,j) \in A'} \left( q_{j}^{(l)}(k) - q_{i}^{(l)}(k) \right) x_{q}^{(l)}(k) \\
+ 2 \sum_{i=1}^{N} \sum_{l=1}^{L-1} \left( q_{i}^{(l+1)}(k) - q_{i}^{(l)}(k) \right) y_{i}^{(l)}(k).
$$

(5–68)

Note that the third term now includes the links to the sink. We also used the fact $y_{i}^{(0)}(k) = D_{d_{i}}$.

Adding $(2/\delta)z(k)$ to both sides of inequality (5–68), we get

$$
\Delta(k) + \frac{2}{\delta}z(k) \leq B + 2 \sum_{i=1}^{N} q_{i}^{(1)}(k) y_{i}^{(0)}(k) \\
+ 2 \left\{ \frac{z(k)}{\delta} + \sum_{l} \sum_{(i,j) \in A'} \left( q_{j}^{(l)}(k) - q_{i}^{(l)}(k) \right) x_{q}^{(l)}(k) \right\} \\
+ 2 \sum_{i=1}^{N} \sum_{l=1}^{L-1} \left( q_{i}^{(l+1)}(k) - q_{i}^{(l)}(k) \right) y_{i}^{(l)}(k) \\
\leq B + 2 \sum_{i=1}^{N} q_{i}^{(1)}(k) y_{i}^{(0)}(\epsilon) + 2 \left\{ \frac{z(\epsilon)}{\delta} + \sum_{l} \sum_{(i,j) \in A'} \left( q_{j}^{(l)}(k) - q_{i}^{(l)}(k) \right) x_{q}^{(l)}(\epsilon) \right\} \\
+ 2 \sum_{i=1}^{N} \sum_{l=1}^{L-1} \left( q_{i}^{(l+1)}(k) - q_{i}^{(l)}(k) \right) y_{i}^{(l)}(\epsilon),
$$

(5–69)
In (5–69), \((\hat{z}(\epsilon), \hat{x}(\epsilon), \hat{y}(\epsilon))\) is an optimal solution of the \(\epsilon\)-perturbed problem defined in (5–59) - (5–62). Based on the earlier remark, \(\hat{y}(\epsilon)\) is feasible to the optimization problem in (5–55), and \(\hat{z}(\epsilon), \hat{x}(\epsilon)\) is feasible to (5–56). But, \(y(k)\) is a minimum to the optimization problem in (5–55), and \(z(k), x(k)\) is a minimum to (5–56). Hence, inequality (5–69) follows.

After regrouping the terms in (5–69) and using (5–62), we have

\[
\Delta(k) + \frac{2}{\delta} z(k) \leq B + \frac{2}{\delta} \hat{z}(\epsilon) + 2 \sum_{l \in \mathcal{L}} \sum_{i \in \mathcal{N}} q_i^{(l)}(k) \left( \sum_{j} \hat{x}_{ji}^{(l)}(\epsilon) - \sum_{j} \hat{x}_{ji}^{(l)}(\epsilon) - \hat{y}_{i}^{(l)}(\epsilon) + \hat{y}_{i}^{(l-1)}(\epsilon) \right)
\]

\[
= B + \frac{2}{\delta} \hat{z}(\epsilon) - 2\epsilon \sum_{l \in \mathcal{L}} \sum_{i \in \mathcal{N}} q_i^{(l)}(k)
\]

(5–70)

In (5–70), the flow conservation constraint (5–61) is used. \(\square\)

Define \(Q(k) = \sum_{l=1}^{L} \sum_{i \in \mathcal{N}} q_i^{(l)}(k)\), which is the sum of the virtual queue sizes at time slot \(k\).

**Theorem 5.4.** There exists a positive constant \(B\) such that for any small positive \(\epsilon\) and \(\delta\), the following holds.

\[
\limsup_{T \to \infty} \frac{1}{T} \sum_{k=0}^{T-1} z(k) \leq \frac{\delta B}{2} + \hat{z}(\epsilon),
\]

(5–71)

\[
\limsup_{T \to \infty} \frac{1}{T} \sum_{k=0}^{T-1} Q(k) \leq \frac{B}{2\epsilon} + \frac{1}{\delta \epsilon} \hat{z}(\epsilon)
\]

(5–72)

**Proof.** Summing the inequality in (5–66) for \(k = 0, 1, \ldots, T - 1\), we have

\[
V(q(T)) - V(q(0)) + \frac{2}{\delta} \sum_{k=0}^{T-1} z(k) \leq BT + \frac{2}{\delta} T \hat{z}(\epsilon) - 2\epsilon \sum_{k=0}^{T-1} Q(k)
\]

(5–73)

After arranging the terms, we get

\[
\frac{1}{T} \sum_{k=0}^{T-1} z(k) \leq \frac{\delta B}{2} + \hat{z}(\epsilon) - \frac{\delta \epsilon}{T} \sum_{k=0}^{T-1} Q(k) - \frac{\delta V(q(T))}{2T} + \frac{\delta V(q(0))}{2T}
\]

\[
\leq \frac{\delta B}{2} + \hat{z}(\epsilon) + \frac{\delta V(q(0))}{2T}.
\]

(5–74)
After taking the limit in $T$, we get (5–71).

Next, from (5–73), we have

$$2\epsilon \sum_{k=0}^{T-1} Q(k) \leq BT + \frac{2T}{\delta} \hat{z}(\epsilon) + V(q(0)) - V(q(T)) - \frac{2}{\delta} \sum_{k=0}^{T-1} z(k) \leq BT + \frac{2T}{\delta} \hat{z}(\epsilon) + V(q(0)).$$

(5–75)

The above inequality is the same as

$$\frac{1}{T} \sum_{k=0}^{T-1} Q(k) \leq \frac{B}{2\epsilon} + \frac{1}{\delta\epsilon} \hat{z}(\epsilon) + \frac{V(q(0))}{2T\epsilon}.$$  

(5–76)

After taking the limit in $T$, we get (5–72).

Let $(x^*, y^*, z^*)$ be an optimal solution to the original problem in (5–17) - (5–23). Note that $z^*$ is also the optimal objective value. Let $\epsilon_o$ be the largest $\epsilon$ for which the perturbed problem (5–59) - (5–62) is feasible.

**Theorem 5.5.** There exists a positive constant $B$ such that for any positive $\delta$, the following holds.

$$\limsup_{T \to \infty} \frac{1}{T} \sum_{k=0}^{T-1} z(k) \leq \frac{\delta B}{2} + z^*,$$  

(5–77)

$$\limsup_{T \to \infty} \frac{1}{T} \sum_{k=0}^{T-1} Q(k) \leq \frac{B}{2\epsilon_o} + \frac{1}{\delta\epsilon_o} \hat{z}(\epsilon_o).$$  

(5–78)

**Proof.** In (5–71), let $\epsilon \to 0$. Since by Theorem 5.3, $\hat{z}(\epsilon) \to z^*$ as $\epsilon \to 0$, we get (5–77).

By Theorem 5.5, we can take $\delta$ small enough so that the long-time average of $z(k)$ is arbitrarily close to the optimum $z^*$. But, this is at the expense of an increase in the provable queue bound.

Now we will prove that the long-time average of $x(k)$ and $y(k)$ eventually satisfies the constraint (5–19). Let $\overline{x}^{(l)}_i(k)$ and $\overline{y}^{(l)}_i(k)$ be $(1/k) \sum_{k=0}^{k-1} x^{(l)}_i(k)$ and $(1/k) \sum_{k=0}^{k-1} y^{(l)}_i(k)$ respectively.
Theorem 5.6. For all \( i \in \mathcal{N} \) and \( l \in \mathcal{L} \), the following condition is satisfied.

\[
\lim_{T \to \infty} \left( \sum_{j \in \mathcal{N}(i)} \bar{x}_{ij}^{(l)}(T) - \sum_{j \in \mathcal{N}(i)} \bar{y}_{ij}^{(l)}(T) + \bar{y}_{i}^{(l-1)}(T) - \bar{y}_{i}^{(l)}(T) \right) \leq 0 \quad (5–79)
\]

Proof. From the (5–57) of main algorithm, for each \( i \in \mathcal{N} \) and \( l \in \mathcal{L} \), we have the following inequality.

\[
q_{i}^{(l)}(k+1) \geq q_{i}^{(l)}(k) - g(i, l, k)
\]
\[
\Leftrightarrow -g(i, l, k) \leq q_{i}^{(l)}(k+1) - q_{i}^{(l)}(k) \quad (5–80)
\]

Since \( Q(k) = \sum_{l=1}^{L} \sum_{i=1}^{N} q_{i}^{(l)}(k) \) is bounded above by the second part of the Theorem 5.5 and each \( q_{i}^{(l)} \) is nonnegative, we know that \( q_{i}^{(l)} \) is also bounded above. Thus, suppose \( q_{i}^{(l)}(k) \leq M_{q} \) for all \( k \).

Summing (5–80) over \( k = 0, 1, \ldots, T - 1 \), we have

\[
\sum_{k=0}^{T-1} -g(i, l, k) \leq q_{i}^{(l)}(T) - q_{i}^{(l)}(0) \leq M_{q} \quad (5–81)
\]

Dividing the above inequality by \( T \), we get

\[
\frac{1}{T} \sum_{k=0}^{T-1} -g(i, l, k) \leq \frac{M_{q}}{T} \quad (5–82)
\]

Letting \( T \to \infty \), we have

\[
\lim_{T \to \infty} \frac{1}{T} \sum_{k=0}^{T-1} -g(i, l, k) \leq 0 \quad (5–83)
\]

(5–83) completes the proof. \( \square \)

In fact, Theorem 5.6 states that the long-time average of \((x, y)\) eventually satisfy the constraints (5–19).

### 5.5 Experimental Results

In this section, we present the results of the numerical experiments to prove the validity of our algorithm. First, we show that our algorithm achieves the optimal objective
value for the problem in (5–17) - (5–23). Next, we show how the Lyapunov drift and the queue size evolve.

For the simulation experiments, we randomly place 50 sensor nodes in a circular region with a radius 25m. We also generate 6 sink locations in the same region for the mobile sink to visit. The cost of transmission between two nodes depends on the distance between them.

\[ c(i, j) = \beta d(i, j)^2, \]  
\hspace{1cm} (5–84)

where \( \beta = 0.0013 \text{pJ/bit/m}^2 \) [19]. The data generation rate of node \( i \) is randomly selected from \([0, 500]\) bps and each node has 500 \text{J} of initial energy.

Transmission can only occur within a limited range, which is defined to be 7.5m in our setting. In the subgradient projection method, we use a constant step size, which is \( \delta = 10^{-8} \). In all simulations, the perturbation parameter is \( \epsilon = 10^{-8} \).

Figure 5-2 shows the convergence result of our algorithm to the primal optimal value. As a reference, the optimal solution of the primal problem (5–17) - (5–23) is obtained by the CPLEX linear programming solver. The curve labeled as \( \bar{z}(k) \) is the time average value of \( z(k) \) at iteration \( k \). Figure 5-2 verifies the first part of our main theorem, Theorem 5.5.
We also measure the Lyapunov drift, \( \Delta(k) = V(q(k + 1)) - V(q(k)) \), at every iteration. As expected by Lemma 3, we can observe that the drift is bounded from above.

Figure 5-4 shows the time averaged value of the total queue size, \( \sum_i \sum_l q^{(l)}_i \). By the second part of Theorem 5.5, this value is bounded from above, which is verified here.

Figure 5-5 show the long-time average value of \( g(i, l, k) = \sum_j x^{(l)}_{ji}(k) - \sum_j x^{(l)}_{ij}(k) + y^{(l-1)}_i(k) - y^{(l)}_i(k) \) for a few selected \( i \) and \( l \). As shown in the figure, it can be observed that long-time average value of \( g(i, l, k) \) converges to 0. It means that the long-time...
average solution of sequence \( \{x(k), y(k)\} \) generated by our algorithm eventually fall into the feasible set.

### 5.6 Implementation Issues

In this section, we discuss several issues relating to the implementation of our algorithm as a network control protocol to maximize the lifetime. There are two ways of implementation for controlling DT-MSM: One is to use the centralized LP solver to control the network and the other is to adopt the distributed algorithm such as our proposed one. Both methods must find how much data for a node to send along its outgoing wireless links in the pursuit of maximizing the network lifetime. The centralized method will find such a solution and must distribute an obtained solution to each node.

In general, the distributed algorithms have several benefits over the centralized algorithms. First, there is no single point of failure. In the centralized implementation, the special node computing the way of controlling the system might be such a point. Therefore, if such a computing node fails, the whole network stops to work. Second, the distributed algorithm typically possesses the scaleability. As the network size becomes larger, the growth rate in computation time of the computing node might be much higher than one in the distributed algorithm. Third, most distributed algorithms are also adaptive and dynamic algorithms as our algorithm. As a consequence, it is easy
Table 5-1. Performance comparison between CPLEX and Algorithm 3

<table>
<thead>
<tr>
<th>network size (# of sensor nodes, # of sink locations)</th>
<th>ratio of computing time</th>
</tr>
</thead>
<tbody>
<tr>
<td>(50,5)</td>
<td>30.333</td>
</tr>
<tr>
<td>(100,5)</td>
<td>25.644</td>
</tr>
<tr>
<td>(200,5)</td>
<td>23.110</td>
</tr>
<tr>
<td>(50,10)</td>
<td>34.474</td>
</tr>
<tr>
<td>(100,10)</td>
<td>24.714</td>
</tr>
<tr>
<td>(200,10)</td>
<td>21.701</td>
</tr>
</tbody>
</table>

for the distributed algorithm to cope with the change of the network. However, when the network changes, the computing node must redo the computation. In other words, the distributed system ensures a certain degree of fault tolerance in nature. For those reasons, distributed algorithm sometimes is preferable as a network control algorithm for the large scale networks.

However, these benefits come at the cost. In our distributed algorithm, every node runs three algorithms presented in the previous sections. Each node requires extra communication for solving two sub-problems. For the sub-problem $S_1$, each node has to exchange information about the current length of virtual queues (which is denoted as $q_i^{(l)}$.) of its all outgoing wireless link in the beginning of time slots. Also, to solve the sub-problem $S_2$, each node disseminates information (which is $P_i$ in the Algorithm 3.) needed to run the Algorithm 3 to all other nodes. In the course of execution Algorithm 3, a node should run the fractional knapsack algorithm, which is Algorithm 2. Note that Algorithm 2 only requires the information about the current queue length of the neighbors and it is the same required arguments for the Algorithm 1.

Since the sub-problem $S_2$ is formulated as Linear Programming problem, it is solved by Linear Programming solver such as CPLEX. We now present the experimental result showing how efficient our Algorithm 3 would be when compared to the Linear Programming solver, which is the CPLEX in this experiment. The efficiency and simplicity of the Algorithm 3 is the key factor to the distributed implementation of the overall problem.
In the Table 5-1, network size is represented by the number of nodes $N$ and the number of sink locations $L$. The measured data is the ratio of the computing time of CPLEX over the computing time of Algorithm 3. As shown in the table, Algorithm 3 shows substantial enhancement in computing the solution for the sub-problem $S_2$.

Our method requires an information exchange in the beginning of the time slots. This information might be conveyed in the form of control message, which is not relevant to the purpose of wireless sensor network. This overhead might not occur in the system which runs centralized problem solver after the system acquires the topology information of the sensor nodes at the very beginning of the operation of the system. In our algorithm, each node should broadcast a single message containing the solution of Algorithm 2 and exchanges the local information about the current length of the virtual queues to its neighbor. Thus, in overall, $N$ broadcast messages and maximum $N^2$ unicast or point-to-point messages are needed as overhead messages. The discussion about efficient way of broadcasting message is beyond the scope of this research. Note that, in general, the cost of broadcast is much higher than the cost of unicast and the message complexity of broadcast dominates the overall message complexity.
CHAPTER 6
CONCLUSION

In this dissertation, we discuss various issues on efficient energy management scheme in wireless sensor network.

First, we show why general techniques, especially the routing protocol used in wired network are inappropriate in wireless sensor network through extensive experiments. The least cost routing algorithms that wired networks prefers sometimes aggravate the situation in wireless sensor network. This is because the least cost routes can be also the most popular routes so that those routes are likely to be used frequently by several nodes. The nodes along the least cost routes tend to exhaust their energy in earlier time. It partitions the network, so that the whole network expires even though the rest of nodes still have plenty of energy. With the help of mathematical optimization, we can obtain the optimal route that will maximize the network lifetime. We compare the performance between the least cost routing and optimal routing in terms of network lifetime. As the size of network increases, the gap between two routing schemes becomes bigger.

Second, we examine how to apply non-uniform deployment of the sensor nodes to resolve the problem of uneven energy consumption rates by the nodes or the energy hole problem in multi-hop wireless sensor networks. More generally, non-uniform deployment with careful density control can be an important technique for achieving a desirable lifetime and system-cost tradeoff of the sensor network. Our main contribution is to present a method for computing the required node density function. As an example, we show that the method enables us to compute the correct densities that achieve an equal energy consumption rate for all nodes, thereby, extending the system lifetime. The method is expected to be widely applicable to other similar objectives and constraints.

Third, we propose a new framework for improving the network lifetime by exploiting sink mobility and delay tolerance. It is expected to be useful in applications that can
tolerate a certain amount of delay in data delivery. We present the mathematical formulations for optimizing the network lifetime under the proposed framework. We identify several properties that our models possess. To validate the framework’s ability for improving network lifetime, we conduct extensive experiments and found that the framework is superior to the models published previously, including the static sink model and the mobile sink model without delay tolerance. The lifetime gain of the proposed model is significant when compared to the previous models. Furthermore, as the number of sink locations increases, the optimal network lifetime increases substantially. The results of the paper can both be applied to practical situations and be used as benchmarks for studying energy-efficient network designs. Furthermore, we can point out three interesting future work directions. The paper has not touched upon the issue of finding efficient algorithms to solve the optimization problems, but has relied on standard, centralized algorithms. The first direction is to find simpler, preferably distributed, algorithms, which are clearly more generally applicable. The goal is likely to be attainable since the problems formulated in this paper are extension of the network-flow problems and many efficient algorithms are known for such problems. The second direction is to relax some of the simplifying assumptions of the formulations. For instance, we can bring the non-zero traveling time by the sink and/or the finite link transmission rate into the formulations. Either one seems to make the problems very difficult, but more relevant at the same time. The third direction is to consider where to choose for sink stops so that the network lifetime can be optimized.

Last, we extend the our research of DT-MSM. In previous work, we show that our model is superior to other models in terms of life time. One of possible direction of further research is to devise a method implementing DT-MSM model in a distributed manner. Among sub flow based model and queue based model, we target the queue based model because it always produces a better result than the other. Basically we apply the dual method because it sometimes reveals nice structural properties.
so that we could decompose the dual problem into several sub-problems. However, in general, dual problems is not differentiable. To deal with non-differentiability, we adopt sub-gradient method. In each iteration, it is required to solve decomposed sub-problems. We proposed (possibly) distributed algorithms for solving them. Moreover, we prove that our approach eventually finds the solution which is arbitrarily close to the optimal solution of the primal problem.
REFERENCES


BIOGRAPHICAL SKETCH

YoungSang Yun was born in ChangHeung, Republic of Korea, in 1970. He received his BS and MS degrees in computer engineering at Pohang University of Science and Technology in Korea in 1992 and 1994, respectively. He worked for the LG Electronics, Inc., Korea from 1994 to 2003. He was involved in development of system software for telecommunication equipments, such as ATM switches. He also managed the IP routing protocol software development team. Since 2003, he has been conducting research with Dr. Ye Xia in the Department of Computer and Information Science and Engineering at the University of Florida. His research interests are wireless sensor networking and mathematical optimization.