Distributed Algorithm for Lifetime Maximization in Delay-Tolerant Wireless Sensor Network with Mobile Sink

YoungSang Yun*, Ye Xia*, Behnam Behdani† and J. Cole Smith†
* Computer and Information Science and Engineering Department,
email: {yyun, yx1}@cise.ufl.edu
† Department of Industrial and Systems Engineering,
University of Florida, Gainesville, FL 32611-6595
email: {behdani@ufl.edu, cole@ise.ufl.edu}

***Abstract*—We propose an algorithm for maximizing the lifetime of a wireless sensor network when there is a mobile sink and the underlying application can tolerate some amount of delay in delivering the data to the sink. The algorithm is distributed, and in addition, mostly uses local information. Such an algorithm can be implemented by parallel and/or distributed execution and the overhead of message passing is low. It is also possible to embed the algorithm into a network protocol so that the sensor nodes and the sink can run it directly as part of the network operation. We give a proof of the algorithm's optimality and the boundedness of the queue sizes, both in the long-run average sense. The proof is based on analyzing a Lyapunov drift.**

***Index Terms*—Wireless Sensor Network, Lifetime Maximization, Distributed Algorithm, Delay-Tolerant Applications, Mobile Sink**

I. INTRODUCTION

A. Motivation

A wireless sensor network (WSN) typically consists of a sink node and a large number of sensor nodes, each of which gathers information from its vicinity and delivers collected data to the sink for further processing in a possibly multi-hop fashion. The sensor nodes usually operate with batteries and are often deployed into a harsh environment. Once deployed, it is hard or even impossible to recharge or replace the batteries of the sensor nodes. Therefore, extending the network lifetime by efficient use of energy is a critical requirement for a WSN.

The network lifetime is usually defined as the time until the first node fails because of energy depletion [1], [2]. Due to the multi-hop routing from the sensor nodes to the sink, the sensor nodes close to the sink usually are the most burdened with relaying data from distant nodes. The traffic imbalance can cause early energy depletion for the nodes near the sink, creating an "energy hole" around the sink. The result may be an early disconnection of the sink from the remaining sensor nodes, which may still have plenty of energy [3], [4], [5].

Recently, the exploitation of mobility to improve the lifetime of a WSN, especially the mobility of the sink, has attracted the interest of researchers [6], [7], [8], [9], [10], [11], [12], [13], [14], [15]. By making stops at different places in the network to receive collected data from the sensor nodes, a mobile sink can better balance the traffic load across the sensor nodes, and as a result, mitigate the energy-hole problem and increase the network lifetime. The use of a mobile sink can also be found naturally in some application scenarios. For instance, in habitat monitoring, a mobile robot may be used to collect information from the sensor nodes in the field. If much of the habitat area is inaccessible by the robot or if it is desirable to minimize disturbance to the targeted animal species, the robot will follow predetermined paths and stop by a set of pre-arranged locations regularly for data collection (see [15] for more examples).

B. Related Work and Contributions

In [15], the authors propose a framework of improving the network lifetime by taking advantage of not only sink mobility but also application delay tolerance. The resulting model is called *Delay Tolerant Mobile Sink Model (DT-MSM)*. DT-MSM is suitable to those applications where some amount of delay in data delivery to the sink is permitted [9]. The sensor nodes may delay the transmission of the collected data and wait for the mobile sink to arrive at the location most favorable for improving the network lifetime. However, finding an efficient algorithm for DT-MSM is not the focus of [15].

The goal of this paper is to find a distributed algorithm that solves the lifetime maximization problem associated with DT-MSM. The decisions to be made include how long the sink should stay at each potential stop, and how to route the data to the sink when it stops (including deciding the amount of data transmitted at various nodes), subject to a maximum tolerable delay. The paper has two main contributions. First, our algorithm is both distributed and mostly local. The overall solution is broken down into smaller decision problems and each decision can be done locally in a sensor node. For the most part, only local information at a node itself and at its neighbors is needed. In general, distributed algorithms are more useful and preferable for networking problems because they can be readily built into network protocols and become network control algorithms. Local algorithms have the additional benefit of restricting the control traffic to be among locally interacting nodes. Second, we analyze our algorithm and show that it converges to the optimal objective value for

the lifetime maximization problem in the long-run average sense, and that the long-time average of the virtual queue sizes are bounded. The proof is based on analyzing a Lyapunov drift.

We briefly survey related work. The authors of [7] study the lifetime maximization problem in a WSN with a mobile sink. They propose a distributed subgradient algorithm for solving the problem. However, their problem is quite different from ours. Furthermore, our algorithm is motivated by a subgradient algorithm; but the deviations have substantial consequences. Unlike their algorithm, standard convergence results for the subgradient algorithms do not apply to our algorithm and the analysis about convergence and algorithm performance in our case relies on a different framework. In addition, their approach relaxes the energy constraints, whereas we relax the flow conservation constraints.

The authors of [6] formulate a linear programming problem for determining how to move the mobile sink and how long the sink should stay at a stop to gather data, with the same objective of maximizing the network lifetime. However, how to route the collected data to the sink is not an interest in their study. In contrast, traffic routing is an important decision in our case.

Regarding the analytical method, the Lyapunov drift technique is widely used for studying the stability issue of control and optimization algorithms for a network of queues. A representative work is [16], where the authors apply this technique to the study of a link scheduling algorithm in multi-hop packet radio networks that proves to be stable and achieve the entire throughput region. Our method is more closely related to [17], which is also about a dynamic control algorithm in wireless networks that attains the optimal performance goal as well as the desired stability property.

C. Assumptions

It is worth commenting on our assumptions regarding the sink's movement. First, the sink's traveling time between stops is negligible. This assumption is widely used in the literature of similar mobile sink problems [6], [10], [11], [12], [13], [14], [15]. It is appropriate for applications where other times of interest, e.g., the delay tolerance level, are much longer than the traveling time. Another reason for making this assumption is tractability. If the sink's traveling time cannot be neglected, the problem becomes very hard to solve. It has a component equivalent to the *Traveling Salesman Problem*, which is known to be an NP-hard problem.

Second, we restrict the possible locations where the sink can stop to a given finite set. If the sink can stop at an arbitrary location, the problem becomes fundamentally different. The authors of [14] study the problem of how to determine the locations, which is NP-hard, and present an approximation algorithm. That problem is essentially orthogonal to ours. Furthermore, although more lifetime can be extracted if the locations of the stops are part of the decision variables, we can approach that performance level by letting the given finite set of potential stops be large. Our assumption is also appropriate when the nature of the sensor field or the special feature of the mobile sink require the locations of the stops to be constrained.

The rest of the paper is organized as follows. Section II describes the system model of DT-MSM and the problem formulation. The proposed distributed algorithm for the problem is given in Section III. In Section IV, we show that the algorithm converges to the optimal value of the problem and the queue sizes are bounded by applying the Lyapunov drift technique. In Section V, we present the experimental results that verify the convergence of our algorithm to the optimal value and the boundedness of the queues in the system. The conclusion is given in Section VI.

II. 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 initial 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 $c : \mathcal{A} \to \mathbb{R}^+$ 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.

As stated earlier, we assume that the traveling time between the sink locations is negligible. With this assumption, the order of visits to the sink locations does not matter for the optimization problem in this paper. The same conclusion has been found true for related problems in several other papers [6], [7], [12], [14]. Thus, for simplicity of explanation, we assume that the order of visits is given as $1 \to 2 \to \cdots \to L$.

DT-MSM was introduced in [15]. 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 in 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 $l \in \mathcal{L}$ within each tour, denoted by t_l, and what the rate of data transmission from node i to j should be while the sink is at l, denoted by $a_{ij}^{(l)}$, for each node pair i and j. 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.

In our problem formulation, t_l and $a_{ij}^{(l)}$ always show up together in the form $t_l a_{ij}^{(l)}$. We can define $x_{ij}^{(l)} = t_l a_{ij}^{(l)}$ to replace $t_l a_{ij}^{(l)}$. 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[1].

In DT-MSM, a sensor node can temporarily delay data transmission and store the data in its local buffer; this is in contrast to the non-delay-tolerant case, where each sensor node must not buffer data. The network can deliberately take advantage of delay tolerance and data buffering as follows. When the sink is at location l, it collects data from a set of sensors R_l, where $R_l \subseteq \mathcal{N}$, through multi-hop communication. The set R_l is called the *coverage* of sink location l. The nodes from outside R_l do not attempt to transmit or relay data when the sink is at location l. We assume R_l is given for each l and $\cup_{l\in\mathcal{L}} R_l = \mathcal{N}$. In a degenerate case, each R_l may be the same as $\mathcal{N}$.[2]

Thus, for each location l, there is a graph $G^l = (\mathcal{N} \cup \{l\}, \mathcal{A}^l)$, where $\mathcal{A}^l = \{(i,j) \in \mathcal{A} | i \in R_l, j \in R_l \cup \{l\}\}$. Recall that when the sink is at location l, the nodes outside the coverage R_l do not participate in the data transfer process, and hence, their incoming or outgoing links are not usable for data transfer. The graph G^l contains only those links that are usable when the sink is at location l, plus those links from a sensor node in the coverage R_l to location l. When the sink is at location l, the (downstream) neighbor set of node i is denoted by $N_l(i) = \{j | (i,j) \in \mathcal{A}^l\}$.

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

1) Start each column with G^l, for all $l \in \mathcal{L}$.
2) Relabel node i in G^l 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^l.
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:

$$e_{ij}^{(l)} = \begin{cases} c(i,j), & \text{if} \quad i,j \in R_l, j \neq s \\ c(i,l), & \text{if} \quad i \in R_l, j = s \\ \infty, & \text{otherwise.} \end{cases} \tag{1}$$

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 such an edge are the same physical node and real communication does not occur.

[1]Once $x_{ij}^{(l)}$ for the link (i,j) is solved, we can assign an arbitrary positive value to t_l, as along as $\sum_{l=1}^{|L|} t_l \leq D$, and assign $a_{ij}^{(l)} = x_{ij}^{(l)}/t_l$. The root reason why we can do this is that there is no upper bound on the link rate $a_{ij}^{(l)}$. If such an upper bound were in our formulations, the situation would be very different and the problem would become very hard.

[2]Each R_l will be chosen by the network operator for added flexibility. Depending on the operator's preference, each R_l may be as large as the entire set of the nodes, or alternatively, it may contain only a small number of nodes around location l. There are practical reasons why the operator may prefer the latter arrangement. When the nodes in R_l are close to location l, any coordination and control can be done 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.

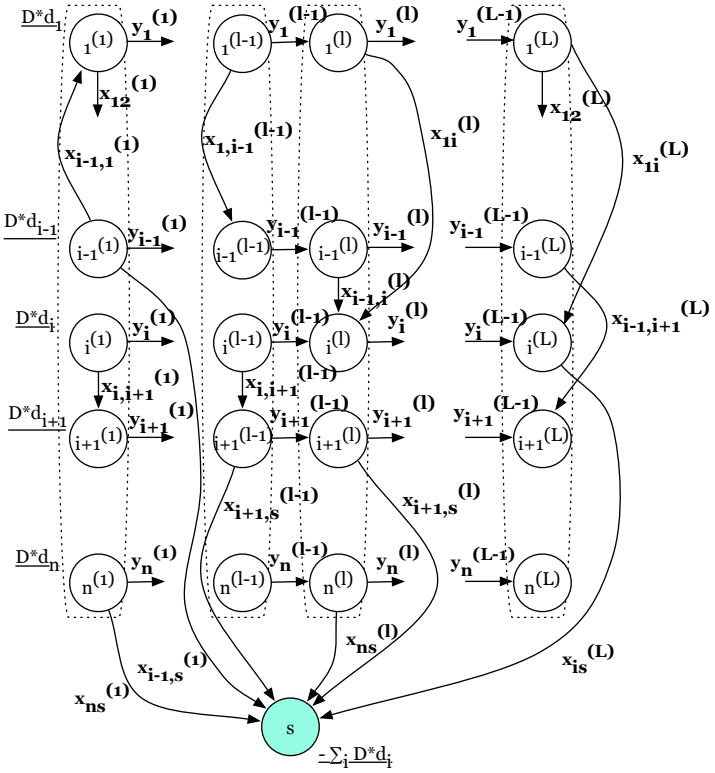

Fig. 1. Expanded graph of DT-MSM

With the introduction of node s in the expanded graph, the definition of $\mathcal{A}^l$ is modified in a natural way for each $l \in \mathcal{L}$. The edges of the form (i,l) in the original $\mathcal{A}^l$ are replaced with (i,s). $N_l(i)$ is still defined as $N_l(i) = \{j | (i,j) \in \mathcal{A}^l\}$ but under the new $\mathcal{A}^l$.

We make the following connectivity assumption: For any sensor node $i \in \mathcal{N}$, there is a path from $i^{(1)}$ to s in the expanded graph. This ensures that there is a way to deliver data from node i to the sink.

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.

$$\begin{cases} \sum_{j:i\in N_1(j)} x_{ji}^{(1)} - \sum_{j\in N_1(i)} x_{ij}^{(1)} - y_i^{(1)} = -D \cdot d_i, \\ \qquad\qquad \text{if } l = 1, \forall i \in \mathcal{N} \\ \sum_{j:i\in N_l(j)} x_{ji}^{(l)} - \sum_{j\in N_l(i)} x_{ij}^{(l)} + y_i^{(l-1)} - y_i^{(l)} = 0, \\ \qquad\qquad \text{if } l \in \{2, \ldots, L-1\}, \forall i \in \mathcal{N} \\ \sum_{j:i\in N_L(j)} x_{ji}^{(L)} - \sum_{j\in N_L(i)} x_{ij}^{(L)} + y_i^{(L-1)} = 0, \\ \qquad\qquad \text{if } l = L, \forall i \in \mathcal{N}. \end{cases} \tag{2}$$

These flow conservation equations ensure that at each node, the total traffic going into a node must be equal to the total traffic coming out of the node. The first equation in (2) corresponds to the nodes in the first column in Fig. 1, which are the source nodes. The second equation is for any node in the columns 2 to $L-1$. The third equation is for the nodes

in the last column.

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

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

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 (2) and (3), subject to the energy constraints at the sensor nodes. More precisely, the problem can be written as follows.

$$\max\ T \tag{4}$$
$$\text{s.t. } (2), (3) \tag{5}$$
$$\left(\sum_{l=1}^{L} \sum_{j\in N_l(i)} e_{ij}^{(l)} x_{ij}^{(l)}\right) T \le E_i, \quad \forall i \in \mathcal{N} \tag{6}$$
$$x_{ij}^{(l)} \ge 0, \quad \forall l \in \mathcal{L}, \forall i \in R_l, \forall j \in N_l(i) \tag{7}$$
$$y_i^{(l)} \ge 0, \quad \forall i \in \mathcal{N}, \forall l \in \mathcal{L} \tag{8}$$
$$T \ge 0. \tag{9}$$

Constraints (6) ensure 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.

III. Decomposition by the Lagrange Method

In this section, we illustrate a distributed algorithm to solve the problem defined in Section II. The following is the equivalent linear problem, which is obtained from the maximization problem of (4) - (9) by replacing $1/T$ with z. The decision variables are the vectors x and y and the scalar z; z is also the optimization objective. For convenience, we also define $M = \sum_{i=1}^{N} Dd_i$, $y_i^{(0)} = Dd_i$ and $y_i^{(L)} = 0$ for all $i \in \mathcal{N}$.

$$\min\ z \tag{10}$$
$$\text{s.t. } \sum_{l=1}^{L} \sum_{j\in N_l(i)} e_{ij}^{(l)} x_{ij}^{(l)} \le z E_i, \quad \forall i \in \mathcal{N} \tag{11}$$
$$\begin{cases} \displaystyle\sum_{j:i\in N_l(j)} x_{ji}^{(l)} - \sum_{j\in N_l(i)} x_{ij}^{(l)} + y_i^{(l-1)} - y_i^{(l)} = 0, \\ \qquad\qquad \forall l \in \mathcal{L}, \forall i \in \mathcal{N} \\ \displaystyle\sum_{l=1}^{L} \sum_{j:s\in N_l(j)} x_{js}^{(l)} - M = 0 \end{cases} \tag{12}$$
$$y_i^{(0)} = Dd_i, \qquad y_i^{(L)} = 0, \quad \forall i \in \mathcal{N} \tag{13}$$
$$x_{ij}^{(l)} \ge 0, \quad \forall l \in \mathcal{L}, \forall i \in R_l, \forall j \in N_l(i) \tag{14}$$
$$y_i^{(l)} \ge 0, \quad \forall i \in \mathcal{N}, \forall l \in \{2, 3, \ldots, L-1\} \tag{15}$$
$$z \ge 0. \tag{16}$$

Note that M is an upper bound of any traffic volume or any buffered data. We will use the terms *flow* and *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, while satisfying flow conservation.

Now, we turn to deriving an algorithm by Lagrangian relaxation. Let $\pi_i^{(l)}$ and π_s be the Lagrange multipliers associated with the constraints in (12). The Lagrangian function of (10) is

$$L(z,x,y,\pi) = z + \pi_s\Big(\sum_{l=1}^{L} \sum_{j:s\in N_l(j)} x_{js}^{(l)} - M\Big) + \sum_{l=1}^{L}\sum_{i=1}^{N} \pi_i^{(l)}\Big(\sum_{j:i\in N_l(j)} x_{ji}^{(l)} - \sum_{j\in N_l(i)} x_{ij}^{(l)} + y_i^{(l-1)} - y_i^{(l)}\Big), \tag{17}$$

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

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

$$L(z,x,y,\pi) = z + \sum_{l=1}^{L} \sum_{(i,j)\in\mathcal{A}^l} (\pi_j^{(l)} - \pi_i^{(l)}) x_{ij}^{(l)} + \sum_{i=1}^{N}\sum_{l=1}^{L-1} (\pi_i^{(l+1)} - \pi_i^{(l)}) y_i^{(l)} - D\sum_{i=1}^{N} (\pi_s - \pi_i^{(1)}) d_i. \tag{18}$$

For convenience, we define $\pi_s^{(l)} \triangleq \pi_s$ for $l = 1, \ldots, L$, and these appear in the second term of equation (18). The Lagrangian dual function $\theta(\pi)$ is now given by

$$\theta(\pi) = \min L(z,x,y,\pi) \tag{19}$$
$$\text{s.t. } \sum_{l=1}^{L} \sum_{j\in N_l(i)} x_{ij}^{(l)} e_{ij}^{(l)} - z E_i \le 0, \quad \forall i \in \mathcal{N} \tag{20}$$
$$x_{ij}^{(l)} \ge 0, \quad \forall l \in \mathcal{L}, \forall i \in \mathcal{N}, \forall j \in N_l(i) \tag{21}$$
$$0 \le y_i^{(l)} \le M, \quad 1 \le l \le L-1, \forall i \in \mathcal{N} \tag{22}$$
$$z \ge 0. \tag{23}$$

We can decompose the problem (19) - (23) into the following two subproblems.

$$S_1 : \min \sum_{i=1}^{N} \sum_{l=1}^{L-1} (\pi_i^{(l+1)} - \pi_i^{(l)}) y_i^{(l)}$$
$$\text{s.t.} \quad 0 \le y_i^{(l)} \le M, \quad \forall i \in \mathcal{N}, 1 \le l \le L-1. \qquad (24)$$

$$S_2 : \min \left\{ z + \sum_{l=1}^{L} \sum_{(i,j)\in\mathcal{A}^l} (\pi_j^{(l)} - \pi_i^{(l)}) x_{ij}^{(l)} \right\}$$
$$\text{s.t.} \quad 0 \le x_{ij}^{(l)} \le M, \quad \forall i \in \mathcal{N}, \forall l \in \mathcal{L}, \forall j \in N_l(i)$$
$$\sum_{l=1}^{L} \sum_{j\in N_l(i)} e_{ij}^{(l)} x_{ij}^{(l)} - z E_i \le 0, \quad \forall i \in \mathcal{N}$$
$$z \ge 0. \qquad (25)$$

Note that we have added the upper bound M to the flow variables x and y.

A. Algorithms for Subproblems

The solution for subproblem S_1 is shown in Algorithm 1. If $(\pi_i^{(l+1)} - \pi_i^{(l)})$ is negative, then we assign the largest value $(= M)$ to the variable $y_i^{(l)}$. Otherwise, $y_i^{(l)}$ should be 0.

Algorithm 1 Solution for S_1

```
if (π_i^(l+1) − π_i^(l)) ≥ 0 then
    y_i^(l) ⇐ 0
else
    y_i^(l) ⇐ M
end if
```

Algorithm 1 can be implemented in a distributed and local manner. The value of $y_i^{(l)}$ can be decided locally in the sensor node i. Also, node i only needs to have the knowledge of π_j from its neighbor node set $N_l(i)$ for all $l \in \mathcal{L}$.

Now, we turn to the subproblem S_2. For ease of exposition, we change the original S_2 to a maximization problem. Suppose, for the moment, z is fixed at $\bar{z}$ and define $f(\bar{z})$ as follows:

$$f(\bar{z}) = \max \left\{ -\bar{z} + \sum_{i=1}^{N} \sum_{l=1}^{L} \sum_{j:j\in N_l(i)} (\pi_i^{(l)} - \pi_j^{(l)}) x_{ij}^{(l)} \right\}$$
$$\text{s.t.} \quad 0 \le x_{ij}^{(l)} \le M, \quad \forall i \in \mathcal{N}, \forall l \in \mathcal{L}, \forall j \in N_l(i)$$
$$\sum_{l=1}^{L} \sum_{j\in N_l(i)} e_{ij}^{(l)} x_{ij}^{(l)} \le \bar{z} E_i, \quad \forall i \in \mathcal{N}.$$

Let

$$f_i(\bar{z}) = \max \sum_{l=1}^{L} \sum_{j:j\in N_l(i)} (\pi_i^{(l)} - \pi_j^{(l)}) x_{ij}^{(l)}$$
$$\text{s.t.} \quad 0 \le x_{ij}^{(l)} \le M, \quad \forall l \in \mathcal{L}, \forall j \in N_l(i)$$
$$\sum_{l=1}^{L} \sum_{j\in N_l(i)} e_{ij}^{(l)} x_{ij}^{(l)} \le \bar{z} E_i.$$

Then, $f(\bar{z}) = -\bar{z} + \sum_{i=1}^{N} f_i(\bar{z})$.

Hence, the maximization problem for finding $f(\bar{z})$ can be further decomposed into smaller maximization problems in which each node i finds $f_i(\bar{z})$. The problem to find $f_i(\bar{z})$ at each node i corresponds to the *fractional knapsack problem*, which is solvable in polynomial time [18].

Suppose there are N knapsacks and knapsack i has a weight capacity of $\bar{z}E_i$. For knapsack i, we wish to pack items, denoted by (i, l, j) for $j \in N_l(i)$ and $l \in \mathcal{L}$. 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). Recall that the maximum available amount of an item is M. The profit of an item (i, l, j) is defined as the reward per unit weight of that item, $(\pi_i^{(l)} - \pi_j^{(l)})/e_{ij}^{(l)}$. 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 listed in Algorithm 2, which greedily packs the most profitable item among the remaining ones until that item is exhausted or the knapsack capacity is reached. This operation stops if all profitable (that is, with a positive profit) items are packed or the knapsack is full.

Algorithm 2 can be implemented in a distributed and local manner. A node i only requires the knowledge of $\pi_j^{(l)}$ of each neighbor $j \in N_l(i)$, for all l.

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

```
sort (i, l, j) in the decreasing order of (π_i^(l) − π_j^(l))/e_ij^(l)
U ⇐ z̄E_i
for each of (i, l, j) in the sorted list do
    if (π_i^(l) − π_j^(l) < 0) then
        break
    else if (U − Me_ij^(l)) < 0 then
        x_ij^(l) ⇐ U/e_ij^(l)
        break
    else
        x_ij^(l) ⇐ M
        U ⇐ U − Me_ij^(l)
    end if
end for
```

The tricky part in solving subproblem S_2 is how to choose the right value for z, so that the overall objective function, $f(z)$, is maximized. Note that each $f_i(z)$ is a concave, nondecreasing, and piecewise linear function of z, and hence, $f(z)$ 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 that 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

$$\begin{cases} \sum_{i\in\mathcal{N}} f_i'(z) > 1, & z < z^* \\ \sum_{i\in\mathcal{N}} f_i'(z) \le 1, & z > z^*. \end{cases} \tag{26}$$

Also note that $f'(z)$ may change only when at least one of the $f_i'(z)$ changes. From Algorithm 2, we see that $f_i'(z)$ is determined by the last item packed from the ordered list, as z is increased to $z + \Delta z$ for some small positive Δz. Suppose this item is (i, j, l). Then, $f_i'(z) = (\pi_i^{(l)} - \pi_j^{(l)})E_i/e_{ij}^{(l)}$. Furthermore, after $f_i'(z)$ takes a new value, it may change again only when z is incremented by $Me_{ij'}^{(l')}/E_i$, where (i, j', l') is the next item on the ordered list.

Hence, the procedure for searching z^* is to keep track of the sequence of points where $f'(z)$ may change, which requires keeping track of the sequence of points where $f_i'(z)$ may change, for each i. Consider a fixed i. Suppose $(\pi_i^{(l)} - \pi_j^{(l)})/e_{ij}^{(l)}$ is sorted in decreasing order and suppose any item (i, l, j) with $(\pi_i^{(l)} - \pi_j^{(l)}) \le 0$ is discarded. Starting with $z_0 = 0$, we can generate a sequence $z_k = z_{k-1} + Me_{ij}^{(l)}/E_i$ 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 3 describes an implementation of the above idea for finding an optimal z^*. For each i, the array $P_i[\,]$ records the sequence of z_k and $f_i'(z_k)$. Algorithm 3 eventually solves the subproblem S_2 by calling Algorithm 2 using the optimal z^*.

Algorithm 3 Solution for S_2

for each $i \in \mathcal{N}$ **do**
  sort (i, l, j) in decreasing order of $(\pi_i^{(l)} - \pi_j^{(l)})/e_{ij}^{(l)}$
  discard any item (i, l, j) if $(\pi_i^{(l)} - \pi_j^{(l)}) \le 0$
  $k \Leftarrow 0$; $z_k \Leftarrow 0$
  for each of (i, l, j) in the sorted list **do**
    $z_k \Leftarrow z_k + Me_{ij}^{(l)}/E_i$
    $P_i[k] \Leftarrow (z_k, (\pi_i^{(l)} - \pi_j^{(l)})E_i/e_{ij}^{(l)})$
    $k \Leftarrow k + 1$
  end for
end for
find z^* which satisfies (26) by searching $(P_i)_{i\in\mathcal{N}}$
each node i applies Algorithm 2 with z^*

The description of Algorithm 3 is ambiguous on how to make the algorithm distributed and (partially) local. There are different possibilities. In one version, each node i executes what is inside 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 2.

B. Main Algorithm

We now combine various algorithms into our main algorithm. We assume that the system operates in a time-slotted way. The Lagrangian dual problem of (10) is

$$\text{Dual:} \quad \max \quad \theta(\pi). \tag{27}$$

Consider the subgradient projection method to solve problem (27).[3] The update of π at each iteration is given by the following equations.

$$\pi_i^{(l)}(k+1) = [\pi_i^{(l)}(k) - \delta(\sum_{j\in N_l(i)} x_{ij}^{(l)}(k) + y_i^{(l)}(k) - \sum_{j:i\in N_l(j)} x_{ji}^{(l)}(k) - y_i^{(l-1)}(k))]^+, \quad \forall i \in \mathcal{N}, \forall l \in \mathcal{L} \tag{28}$$

$$\pi_s(k+1) = [\pi_s(k) - \delta(M - \sum_{l=1}^{L} \sum_{(j,s)\in\mathcal{A}^l} x_{js}^{(l)}(k))]^+, \tag{29}$$

where $[b]^+ = \max\{0, b\}$ and δ is a sufficiently small positive number.[4]

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

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

Main Algorithm

$$y(k) \in \arg\min\{\sum_{i=1}^{N}\sum_{l=1}^{L-1}(q_i^{(l+1)}(k) - q_i^{(l)}(k))y_i^{(l)}\} \tag{30}$$
$$\text{s.t.} \quad y_i^{(l)} \in [0, M(\epsilon)], i \in \mathcal{N}, 1 \le l \le L-1.$$

[3]We briefly describe the subgradient algorithm [19]. Consider a convex optimization problem: $\min_{x\in X} f(x)$, subject to $g_j(x) \le 0, j = 1, \ldots, r$, where $f : \mathbb{R}^n \to \mathbb{R}$, $g_j : \mathbb{R}^n \to \mathbb{R}$, for $j = 1, \ldots, r$, are convex functions, and $X \subseteq \mathbb{R}^n$ is a convex and closed set. Let the Lagrangian function be $L(x, \mu) = (f(x) + \mu' g(x))$, where μ' is the transpose of μ. The dual problem to the above minimization problem is $\max_{\mu\ge 0} q(\mu)$, where $q(\mu)$ is the dual function defined by $q(\mu) = \min_{x\in X} L(x, \mu)$. Let $x(k)$ and $\mu(k)$ be the primal and dual variables after the k-th iteration of the algorithm, and let $\alpha(k) \ge 0$ be the step size in the k-th iteration. Then, the subgradient projection algorithm is: $x(k) \in \arg\min_{x\in X} L(x, \mu(k))$; $\mu(k+1) = [\mu(k) + \alpha(k)g(x(k))]^+$. It can be shown that $g(x(k))$ is a subgradient of the dual function $q(\mu)$ at the point $\mu(k)$. Hence, at each iteration, the dual variables move in the direction of a subgradient (followed by projection to $\mathbb{R}_+^r$).

[4]Although there is no non-negativity requirement on π in the optimization problem (27), the subgradient algorithm tries to find a non-negative optimal π, which exists.

$$(z(k),\ x(k)) \in \arg\min\{\frac{z}{\delta}+ \sum_{l=1}^{L}\sum_{(i,j)\in\mathcal{A}^l}(q_j^{(l)}(k)-q_i^{(l)}(k))x_{ij}^{(l)}\} \quad (31)$$

$$\text{s.t.} \quad x_{ij}^{(l)} \in [0, M(\epsilon)], i \in \mathcal{N}, l \in \mathcal{L}, j \in N_l(i)$$
$$\sum_{l=1}^{L}\sum_{j\in N_l(i)} e_{ij}^{(l)} x_{ij}^{(l)} \le zE_i, i \in \mathcal{N}$$
$$z \ge 0$$

$$q_i^{(l)}(k+1) = [q_i^{(l)}(k) - (\sum_{j\in N_l(i)} x_{ij}^{(l)}(k) + y_i^{(l)}(k) - \sum_{j:i\in N_l(j)} x_{ji}^{(l)}(k) - y_i^{(l-1)}(k))]^+, \quad \forall i \in \mathcal{N}, \forall l \in \mathcal{L} \quad (32)$$

$$q_s(k+1) = 0. \quad (33)$$

Note that (30) and (31) are solved by Algorithms 1 and 3, respectively, with suitable modification of the notation. We can consider $q_i^{(l)}$ as a virtual queue at node $i^{(l)}$ in Fig. 1, and (32) 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 ($\sum_{j:i\in N_l(j)} x_{ji}^{(l)} + y_i^{(l-1)}$) minus the total service ($\sum_{j\in N_l(i)} x_{ij}^{(l)} + y_i^{(l)}$). Since $q_s(k+1) = 0$, all flow reaching the sink should be drained out.

IV. Convergence 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 [17] and [20].

We first define an ϵ-perturbed problem to the original problem in (10) - (16), which will be used later.[5] Here, ϵ is the same small positive constant in the definition of $M(\epsilon)$.

$$\min\ z \quad (34)$$

$$\text{s.t.} \sum_{l=1}^{L}\sum_{j\in N_l(i)} e_{ij}^{(l)} x_{ij}^{(l)} \le zE_i, \quad \forall i \in \mathcal{N} \quad (35)$$

$$\begin{cases} \sum_{j:i\in N_l(j)} x_{ji}^{(l)} - \sum_{j\in N_l(i)} x_{ij}^{(l)} + y_i^{(l-1)} - y_i^{(l)} = -\epsilon, \\ \qquad\qquad \forall l \in \mathcal{L}, \forall i \in \mathcal{N} \\ \sum_{l=1}^{L}\sum_{j:s\in N_l(j)} x_{js}^{(l)} = M + NL\epsilon \end{cases} \quad (36)$$

$$y_i^{(0)} = Dd_i, \qquad y_i^{(L)} = 0, \quad \forall i \in \mathcal{N}. \quad (37)$$

The usual non-negativity constraints of the variables are still required. In the above problem, we inject extra supply in the amount ϵ 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: A crucial fact is that any feasible flow to the perturbed problem still satisfies $y_i^{(l)} \in [0, M(\epsilon)], i \in \mathcal{N}, 1 \le l \le L-1$; $x_{ij}^{(l)} \in [0, M(\epsilon)], i \in \mathcal{N}, l \in \mathcal{L}, j \in N_l(i)$; $\sum_{l=1}^{L}\sum_{j\in N_l(i)} e_{ij}^{(l)} x_{ij}^{(l)} \le zE_i, i \in \mathcal{N}$; and $z \ge 0$. Hence, the vector y is feasible to the optimization problem in (30), and (z, x) is feasible to the problem in (31).

In the following lemma, we discuss some useful properties of the optimal objective value function of the ϵ-perturbed problem. For simplicity of discussion, we consider the standard linear programming problem:

$$(P)\ \min\ c^T x$$
$$\text{s.t. } Ax = d$$
$$x \ge 0.$$

In the problem ($P(\epsilon)$) below, the right hand side of the equality constraint in (P) is perturbed by ϵ along the direction Δd.

$$(P(\epsilon))\ \min\ c^T x$$
$$\text{s.t. } Ax = d + \epsilon\Delta d$$
$$x \ge 0. \quad (38)$$

Suppose problem (P) is feasible and problem $(P(\epsilon))$ is feasible for $\epsilon \in [0, \epsilon_o]$, where ϵ_o is a positive constant. Let f^* and $f^*(\epsilon)$ be the optimal objective values for problem (P) and $(P(\epsilon))$, respectively, where $\epsilon \in [0, \epsilon_o]$.

Lemma 1. *$f^*(\epsilon)$ is continuous, convex, and piecewise linear on $[0, \epsilon_o]$.*

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

$$(D)\ \max\ d^T\pi$$
$$\text{s.t. } A^T\pi \le c,$$

where π is the dual variable associated with the equality constraint of (P). Similarly, the dual problem for $(P(\epsilon))$ is:

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

Note that the set of feasible solutions for $(D(\epsilon))$ is the same as that for (D). Let $\mathcal{D}$ denote this set, which does not depend on ϵ. By strong duality [21], we know that $f^*(\epsilon) = \max\{(d+\epsilon\Delta d)^T\pi | \pi \in \mathcal{D}\}$ for $\epsilon \in [0, \epsilon_o]$. For any $\epsilon \in [0, \epsilon_o]$, the optimal objective value can be obtained at one of the extreme points of $\mathcal{D}$. Let Ω be the set of extreme points of $\mathcal{D}$, which is a finite set. Hence, $f^*(\epsilon) = \max\{(d+\epsilon\Delta d)^T\pi | \pi \in \Omega\}$. For each $\pi \in \Omega$, $(d+\epsilon\Delta d)^T\pi$ is a linear function of ϵ. Therefore, $f^*(\epsilon) = \max\{(d+\epsilon\Delta d)^T\pi | \pi \in \mathcal{D}\}$ is a continuous, convex, and piecewise linear function of ϵ on $[0, \epsilon_o]$ [22]. ■

Let ϵ_o be the largest ϵ for which the perturbed problem (34) - (37) is feasible.

Therorem 2. *Let $\hat{z}(\epsilon)$ be the optimal value for the problem in (34) - (37), where $\epsilon \in [0, \epsilon_o]$, and z^* be the optimal value for the unperturbed problem. Then, $\hat{z}(\epsilon) \to z^*$ as $\epsilon \to 0$.*

[5] For ease of presentation, we assume $R_l = \mathcal{N}$ for all $l \in \mathcal{L}$ from now on. If this is not the case, we need to modify the ϵ-perturbed problem as follows. From the expanded graph, we remove any node that does not have a path to the sink s. Then, we inject ϵ amount of extra supply to each of the remaining nodes, except s. Suppose the total number of remaining nodes, excluding s, is K. We change the right hand side of the second equation in (36) from $M + NL\epsilon$ to $M + K\epsilon$.

Proof: This is a direct consequence of Lemma 1. ■

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 $V(q) = \sum_{i\in\mathcal{N}}\sum_{l\in\mathcal{L}}(q_i^{(l)})^2$. Let $\Delta(k) \triangleq V(q(k+1)) - V(q(k))$, which is known as the Lyapunov drift.

Lemma 3. *There exists a positive constant B such that for any $\epsilon \in [0, \epsilon_o]$ and any $\delta > 0$, the following condition holds for any time slot k and for any $q(k)$,*

$$\Delta(k) + \frac{2}{\delta}z(k) \leq B + \frac{2}{\delta}\hat{z}(\epsilon) - 2\epsilon\sum_{l\in\mathcal{L}}\sum_{i\in\mathcal{N}} q_i^{(l)}, \tag{39}$$

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

Proof: By squaring (32) and arranging it, we get

$$(q_i^{(l)}(k+1))^2 - (q_i^{(l)}(k))^2 \leq g^2(i,l,k) - 2q_i^{(l)}(k)g(i,l,k), \tag{40}$$

where

$$g(i,l,k) \triangleq \sum_{j\in N_l(i)} x_{ij}^{(l)}(k) - \sum_{j:i\in N_l(j)} x_{ji}^{(l)}(k) + y_i^{(l)}(k) - y_i^{(l-1)}(k).$$

Note that $g(i,l,k) \leq NM$ because $\sum_{j\in N_l(i)} x_{ij}^{(l)}(k) \leq (N-1)M$ and $y_i^{(l)}(k) \leq M$ for all i, l, and k.

Summing the inequality in (40) over all l and i, we have

$$\begin{aligned}
&\sum_{l=1}^{L}\sum_{i=1}^{N}\left[\left(q_i^{(l)}(k+1)\right)^2 - \left(q_i^{(l)}(k)\right)^2\right] \\
\leq&\sum_{l,i} g^2(i,l,k) - 2\sum_{l,i} q_i^{(l)}(k)g(i,l,k) \\
\leq&LN^3M^2 + 2\sum_{l,i} q_i^{(l)}(k)\left(-\sum_{j\in N_l(i)} x_{ij}^{(l)}(k) + \right. \\
&\left.\sum_{j:i\in N_l(j)} x_{ji}^{(l)}(k) - y_i^{(l)}(k) + y_i^{(l-1)}(k)\right) \\
=&B - 2\sum_{l=1}^{L}\sum_{(j,s)\in\mathcal{A}^l} q_j^{(l)}(k)x_{js}^{(l)}(k) \\
&+ 2\sum_{l=1}^{L}\sum_{(i,j)\in\mathcal{A}^l;j\neq s}\left(q_j^{(l)}(k) - q_i^{(l)}(k)\right)x_{ij}^{(l)}(k) \\
&+ 2\sum_{i=1}^{N}\sum_{l=1}^{L-1}\left(q_i^{(l+1)}(k) - q_i^{(l)}(k)\right)y_i^{(l)}(k) \\
&+ 2\sum_{i=1}^{N} q_i^{(1)}(k)y_i^{(0)}(k),
\end{aligned} \tag{41}$$

where $B \triangleq LN^3M^2$. (41) is obtained by regrouping the terms based on variables x and y. Note that the third term on the right hand of (41) excludes the links to the sink. Adding $2q_s(k)(\sum_l\sum_{(j,s)\in\mathcal{A}^l} x_{js}^{(l)}(k)) = 0$ to (41) and also using the fact $y_i^{(0)}(k) = Dd_i$, we have

$$\begin{aligned}
\Delta(k) \leq& B + 2\sum_{i=1}^{N} q_i^{(1)}(k)Dd_i \\
&+ 2\sum_{l}\sum_{(i,j)\in\mathcal{A}^l}\left(q_j^{(l)}(k) - q_i^{(l)}(k)\right)x_{ij}^{(l)}(k) \\
&+ 2\sum_{i=1}^{N}\sum_{l=1}^{L-1}\left(q_i^{(l+1)}(k) - q_i^{(l)}(k)\right)y_i^{(l)}(k).
\end{aligned} \tag{42}$$

Note that the second term now includes the links to the sink.

Adding $(2/\delta)z(k)$ to both sides of inequality (42), we get

$$\begin{aligned}
&\Delta(k) + \frac{2}{\delta}z(k) \leq B + 2\sum_{i=1}^{N} q_i^{(1)}(k)Dd_i \\
&+2\left\{\frac{z(k)}{\delta} + \sum_{l}\sum_{(i,j)\in\mathcal{A}^l}(q_j^{(l)}(k) - q_i^{(l)}(k))x_{ij}^{(l)}(k)\right\} \\
&+ 2\sum_{i=1}^{N}\sum_{l=1}^{L-1}\left(q_i^{(l+1)}(k) - q_i^{(l)}(k)\right)y_i^{(l)}(k) \\
&\leq B + 2\sum_{i=1}^{N} q_i^{(1)}(k)Dd_i \\
&+2\left\{\frac{\hat{z}(\epsilon)}{\delta} + \sum_{l}\sum_{(i,j)\in\mathcal{A}^l}(q_j^{(l)}(k) - q_i^{(l)}(k))\hat{x}_{ij}^{(l)}(\epsilon)\right\} \\
&+ 2\sum_{i=1}^{N}\sum_{l=1}^{L-1}\left(q_i^{(l+1)}(k) - q_i^{(l)}(k)\right)\hat{y}_i^{(l)}(\epsilon),
\end{aligned} \tag{43}$$

In (43), $(\hat{z}(\epsilon), \hat{x}(\epsilon), \hat{y}(\epsilon))$ is an optimal solution of the ϵ-perturbed problem defined in (34) - (37). Based on the earlier remark, $\hat{y}(\epsilon)$ is feasible to the optimization problem in (30), and $(\hat{z}(\epsilon), \hat{x}(\epsilon))$ is feasible to (31). But, $y(k)$ is a minimum to the optimization problem in (30), and $(z(k), x(k))$ is a minimum to (31). Hence, inequality (43) follows.

After regrouping the terms in (43) and using (37), we have

$$\begin{aligned}
&\Delta(k) + \frac{2}{\delta}z(k) \leq B + \frac{2}{\delta}\hat{z}(\epsilon) + \\
&2\sum_{\substack{l\in\mathcal{L}\\ i\in\mathcal{N}}} q_i^{(l)}(k)\left(\sum_j \hat{x}_{ji}^{(l)}(\epsilon) - \sum_j \hat{x}_{ij}^{(l)}(\epsilon) - \hat{y}_i^{(l)}(\epsilon) + \hat{y}_i^{(l-1)}(\epsilon)\right) \\
&= B + \frac{2}{\delta}\hat{z}(\epsilon) - 2\epsilon\sum_{l\in\mathcal{L}}\sum_{i\in\mathcal{N}} q_i^{(l)}(k).
\end{aligned} \tag{44}$$

In (44), the flow conservation constraint (36) is used. ■

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.

Therorem 4. *There exists a positive constant B such that for any $\epsilon \in (0, \epsilon_o]$ and any $\delta > 0$, the following holds.*

$$\limsup_{T\to\infty}\frac{1}{T}\sum_{k=0}^{T-1} z(k) \leq \frac{\delta B}{2} + \hat{z}(\epsilon), \tag{45}$$

$$\limsup_{T\to\infty} \frac{1}{T}\sum_{k=0}^{T-1} Q(k) \le \frac{B}{2\epsilon} + \frac{1}{\delta\epsilon}\hat{z}(\epsilon). \quad (46)$$

Proof: Summing the inequality in (39) for $k = 0, 1, \cdots, T-1$, we have

$$V(q(T)) - V(q(0)) + \frac{2}{\delta}\sum_{k=0}^{T-1} z(k) \le BT + \frac{2}{\delta}T\hat{z}(\epsilon) - 2\epsilon\sum_{k=0}^{T-1} Q(k) \quad (47)$$

After arranging the terms, we get

$$\begin{aligned}\frac{1}{T}\sum_{k=0}^{T-1} z(k) &\le \frac{\delta B}{2} + \hat{z}(\epsilon) \\ &- \frac{\delta\epsilon}{T}\sum_{k=0}^{T-1} Q(k) - \frac{\delta V(q(T))}{2T} + \frac{\delta V(q(0))}{2T} \\ &\le \frac{\delta B}{2} + \hat{z}(\epsilon) + \frac{\delta V(q(0))}{2T}. \end{aligned} \quad (48)$$

After taking the limit in T, we get (45).

Next, from (47), we have

$$\begin{aligned} 2\epsilon &\sum_{k=0}^{T-1} Q(k) \\ &\le BT + \frac{2T}{\delta}\hat{z}(\epsilon) + V(q(0)) - V(q(T)) - \frac{2}{\delta}\sum_{k=0}^{T-1} z(k) \\ &\le BT + \frac{2T}{\delta}\hat{z}(\epsilon) + V(q(0)). \end{aligned} \quad (49)$$

The above inequality is the same as

$$\frac{1}{T}\sum_{k=0}^{T-1} Q(k) \le \frac{B}{2\epsilon} + \frac{1}{\delta\epsilon}\hat{z}(\epsilon) + \frac{V(q(0))}{2T\epsilon}. \quad (50)$$

After taking the limit in T, we get (46). ∎

Let (x^*, y^*, z^*) be an optimal solution to the original problem in (10) - (16). Note that z^* is also the optimal objective value.

Therorem 5. *There exists a positive constant B such that for any positive δ, the following holds.*

$$\limsup_{T\to\infty} \frac{1}{T}\sum_{k=0}^{T-1} z(k) \le \frac{\delta B}{2} + z^*, \quad (51)$$

$$\limsup_{T\to\infty} \frac{1}{T}\sum_{k=0}^{T-1} Q(k) \le \frac{B}{2\epsilon_o} + \frac{1}{\delta\epsilon_o}\hat{z}(\epsilon_o) \quad (52)$$

Proof: In (45), let $\epsilon \to 0$. Since by Theorem 2, $\hat{z}(\epsilon) \to z^*$ as $\epsilon \to 0$, we get (51). ∎

By Theorem 5, we can take δ 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.

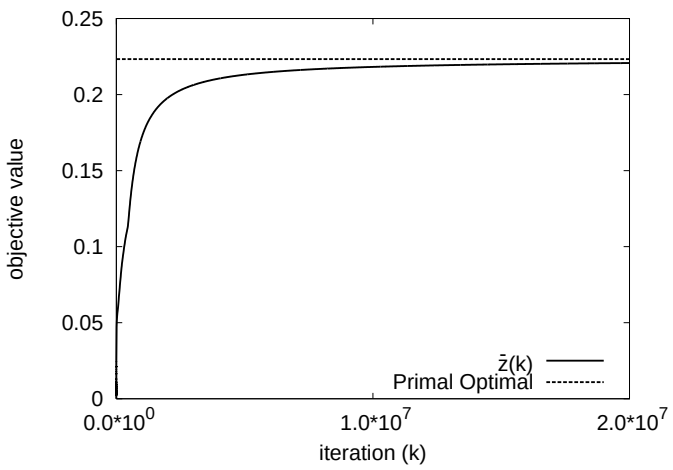

Fig. 2. Convergence to the optimal value, z^*

V. Experimental Results

In this section, we present the results of the simulation/numerical experiments to verify the validity of our algorithm. First, we show that our algorithm achieves the optimal objective value for the problem in (10) - (16). Then, we show how the Lyapunov drift and the queue size evolve.

For the 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 transmitting one bit of data between two nodes depends on the distance between them as in the first order radio model [23]:

$$c(i,j) = \beta d(i,j)^2, \quad (53)$$

where $\beta = 100$ pJ/bit/m^2 [23]. The data generation rate of each node is randomly selected from $[0, 500]$ bps and each node has 500 J of initial energy.

Transmission can only occur within a limited range, which is assumed to be 7.5m. For the algorithm, we use $\delta = 10^{-8}$ and $\epsilon = 10^{-8}$.

Fig. 2 shows the convergence result of our algorithm to the primal optimal value. As a reference, the optimal solution of the primal problem (10) - (16) 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. Fig. 2 verifies the first part of our main theorem, Theorem 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.

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

VI. Conclusion

In this paper, we propose an algorithm for maximizing the WSN lifetime when there is a mobile sink and the underlying application can tolerate some degree of delay in delivering the data to the sink. Although known linear or convex optimization

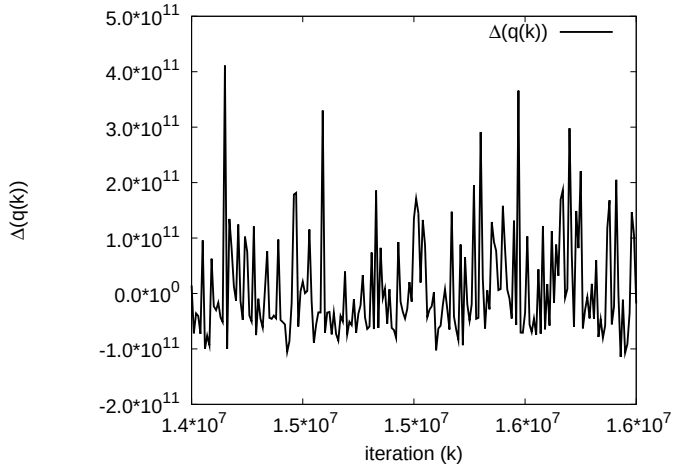

Fig. 3. Lyapunov drift of the algorithm over time

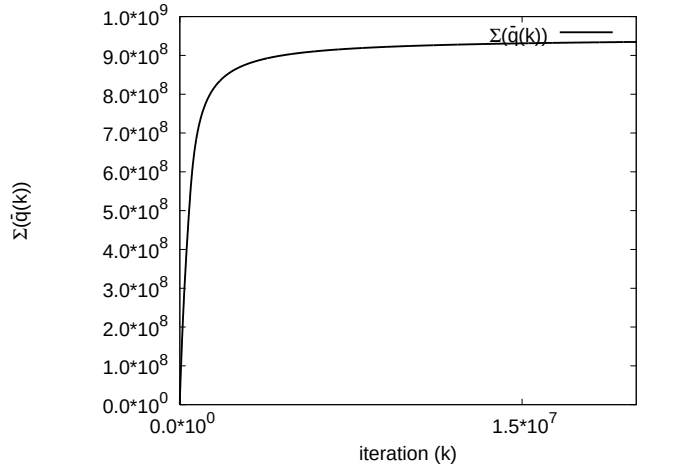

Fig. 4. Time average of total virtual queue size over time

algorithms may be able to solve the lifetime optimization problem using a centralized solver, our goal is to develop a distributed, and preferably, local algorithm. Such an algorithm can be implemented by parallel and/or distributed execution and the overhead of message passing is low. It is also possible to embed the algorithm into a network protocol so that the sensor nodes and the sink can run it directly as part of the network operation.

Whether it is possible to find such an algorithm for an arbitrary problem depends on the specific structure of the problem. The breakthrough in our work is to recognize our problem as a network flow problem on an expanded graph. Then, we modify a standard subgradient algorithm for the network flow problem and make it into a distributed algorithm. In addition, the resulting algorithm mostly uses local information for execution and control.

The resulting algorithm is not a conventional optimization algorithm because its optimality is in the long-run average sense. The results and the analytical techniques from the standard optimization theory do not apply. We give a proof of the algorithm's optimality and the boundedness of the queue sizes based on analyzing a Lyapunov drift.

Finally, we discuss several possible future research directions. First, our algorithm currently uses virtual queue sizes and other virtual quantities, such as virtual traffic volumes, for control. To make it more like an adaptive network algorithm, it is desirable to incorporate real queue sizes and real traffic volumes into the algorithm. This will allow the nodes to send real traffic while the algorithm is being executed. One next step of research is to make such a modification and analyze the stability and optimality of the modified system. Second, it may be possible to derive simpler, sub-optimal or heuristic algorithms based on the insights gained from studying the optimal algorithm in this paper. These heuristics can be evaluated and compared with the optimal one under more comprehensive evaluation criteria, including various engineering costs. Third, the problem formulation can be incrementally enriched to reflect additional constraints or cost considerations. Examples include models in which the sink traveling time is non-negligible or the sink locations become additional decision variables. These new problems are usually much more difficult to solve optimally. Deriving heuristic algorithms in a principled way, as advocated above, may be a good strategy to meet the challenge.

REFERENCES

[1] J. Chang and L. Tassiulas, "Routing for maximum system lifetime in wireless ad hoc networks," in *37th Annual Allertion Conf. Communication, Control, and Computing*, Monticello, IL, September 1999.

[2] ——, "Maximum lifetime routing in wireless sensor networks," *IEEE/ACM Transactions of Networking*, vol. 12, pp. 609–619, August 2004.

[3] J. Li and P. Mohapatra, "An analytical model for the energy hole problem in many-to-one sensor networks," in *Proc. of Vehicular Technology Conference*, January 2005, pp. 2721– 2725.

[4] ——, "Analytical modeling and mitigation techniques for the energy hole problem in sensor networks," *Pervasive and Mobile Computing*, vol. 3, no. 8, pp. 233–254, 2007.

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

[6] Z. M. Wang, S. Basagni, E. Melachrinoudis, and C. Petrioli, "Exploiting sink mobility for maximizing sensor network lifetime," in *38th Hawaii International Conference on System Science*, 2005.

[7] M. Gatzianas and L. Georgiadis, "A distributed algorithm for maximum lifetime routing in sensor networks with mobile sink," *IEEE Transactions on Wireless Communications*, vol. 7, no. 3, pp. 984–994, March 2008.

[8] J. Luo and J.-P. Hubaux, "Joint mobility and routing for lifetime elongation in wireless sensor networks," in *INFOCOM 05*, 2005.

[9] R. C. Shah, S. Roy, S. Jain, and W. Brunette, "Data MULEs: Modeling a three-tier architecture for sparse sensor networks," in *the First IEEE International Workshop on Sensor Network Protocols and Applications, SNPA 2003*, Anchorage, AK, May 2003, pp. 30–41.

[10] S. Basagni, A. Carosi, E. Melachrinoudis, C. Petrioli, and Z. M. Wang, "A new MILP formulation and distributed protocols for wireless sensor networks lifetime maximization," in *IEEE International Conference on Communications 2006*, June 2006, pp. 3517–3524.

[11] W. Wang, V. Srinivasan, and K.-C. Chua, "Using mobile relays to prolong the lifetime of wireless sensor networks," in *MobiCom '05*, 2005, pp. 270–283.

[12] I. Papadimitriou and L. Georgiadis, "Maximum lifetime routing to mobile sink in wireless sensor networks," in *The 13th IEEE SoftCom, 2005*, 2005.

[13] S. Gandham, M. Dawande, R. Prakash, and S. Venkatesan, "Energy efficient schemes for wireless sensor networks with multiple mobile base stations," in *Global Telecommunications Conference, 2003. GLOBECOM '03. IEEE*, December 2003.

[14] Y. Shi and Y. T. Hou, "Theoretical results on base station movement problem for sensor network," in *IEEE INFOCOM '08*, 2008.

[15] Y. Yun and Y. Xia, "Maximizing the lifetime of wireless sensor networks with mobile sink in delay-tolerant applications," *IEEE Transactions on Mobile Computing*, vol. 9, no. 9, pp. 1308–1318, September 2010.
[16] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 12, no. 12, pp. 1936–1948, December 1992.
[17] M. J. Neely, E. Modiano, and C.-P. Li, "Fairness and optimal stochastic control for heterogeneous networks," *IEEE Transactions on Networking*, vol. 16, no. 2, April 2008.
[18] E. Horowitz, S. Sahni, and S. Rajasekaran, *Computer Algorihtms/C++*. Computer Science Press, 1996.
[19] D. Bertsekas, *Nonlinear Programming*, 2nd ed. Athena Scientific, 2003.
[20] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Foundations and Trends in Networking*, vol. 1, no. 1, pp. 1–144, 2006.
[21] M. S. Bazaraa, J. J. Jarvis, and H. D. Sherali, *Linear Programming and Network Flows*. Wiley-Interscience, 2004.
[22] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.
[23] W. R. Heinzelman, A. Chadrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proc. of the 33rd Hawaii International Conference on System Sciences*, January 2000.

YoungSang Yun received his PhD degree from the University of Florida in 2010 and the MS and BS degrees in Computer Science and Engineering from Pohang University of Science and Technology, Korea in 1994 and 1992, respectively. He was a research and development engineer in LG Electronics, Korea between 1994 and 2003. His research interests include network control algorithm for the wireless sensor network and mathematical optimization of network protocols and algorithms.

Ye Xia is an associate professor at the Computer and Information Science and Engineering department at the University of Florida, starting in August 2003. He has a PhD degree from the University of California, Berkeley, in 2003, an MS degree in 1995 from Columbia University, and a BA degree in 1993 from Harvard University, all in Electrical Engineering. Between June 1994 and August 1996, he was a member of the technical staff at Bell Laboratories, Lucent Technologies in New Jersey. His research interests are in the area of communication networks, including performance evaluation, network resource allocation, and distributed algorithms. He is also interested in probability theory and stochastic processes.

Behnam Behdani is a Ph.D. candidate in Department of Industrial and Systems Engineering at the University of Florida. He received his bachelor's degrees in industrial engineering from Sharif University of Technology in 2005. He also has a master's degrees in industrial engineering from Sharif University of Technology. He joined the Department of Industrial and Systems Engineering at the University of Florida in August 2007 as a Ph.D. student. His research interests lie in the operations research area, including integer programming, network optimization, decomposition approaches to large-scale optimization problems, and robust optimization.

Cole Smith is a Professor of Industrial and Systems Engineering at the University of Florida. He earned in MS in Mathematical Sciences from Clemson University in 1996 and his PhD in Industrial and Systems Engineering from Virginia Tech in 2000. His research interests lie in mathematical programming, particularly in integer programming and network optimization theory and methods. Prof. Smith's current research topics involve large-scale and multi-agent optimization problems, especially in scenarios involving uncertainty and/or discrete decisions.