2012 8th IEEE International Conference on Distributed Computing in Sensor Systems

Network Lifetime Maximization in Delay-Tolerant Sensor Networks With a Mobile Sink

Zichuan Xu Weifa Liang
Research School of Computer Science
Australian National University
Canberra, ACT 0200, Australia
Email: edward.xu@anu.edu.au, wliang@cs.anu.edu.au

Yinlong Xu
School of Computer Science and Technology
University of Science and Technology of China
Hefei, Anhui 230026, P.R China
Email: ylxu@ustc.edu.cn

***Abstract*—In this paper we investigate the network lifetime maximization problem in a delay-tolerant wireless sensor network with a mobile sink by exploiting a nontrivial tradeoff between the network lifetime and the data delivery delay. We formulate the problem as a joint optimization problem that consists of finding a trajectory for the mobile sink and designing an energy-efficient routing protocol to route sensing data to the sink, subject to the bounded delay on data delivery and the given potential sink location space. Due to NP-hardness of the problem, we then propose a novel optimization framework, which not only prolongs the network lifetime but also improves the other performance metrics including the network scalability, robustness, and the average delivery delay. We finally conduct extensive experiments by simulations to evaluate the performance of the proposed algorithm against other heuristics. The experimental results demonstrate that the proposed algorithm outperforms the others significantly in terms of network lifetime prolongation.**

I. INTRODUCTION

Wireless sensor networks (WSNs) consist of several hundreds to thousands of battery-powered sensors that are endowed with a multitude of sensing modalities including multimedia (*e.g.*, video, audio) and scalar data (*e.g.*, temperature, pressure, light, infrared). The strong demand for WSNs has been spurred by numerous applications that require in-situ, unattended, high-precision, and real-time observation over the monitored region [1], [16]. Although there have been significant progress in sensor fabrications including processing design and computing, advances of battery technology still lag behind, making energy resource the fundamental constraint in WSNs. To maximize the network lifetime, energy conservation in such networks is of paramount importance. In conventional sensor networks, there is only a single stationary sink (a base station), that has unlimited power supply serving as the gateway between the network and users. The sink functionalities typically include gathering data from the sensors via multi-hop relays, performing data processing, and returning query results to users. The sink is often placed in a strategic location in the monitoring region to enable the network operating as long as possible.

Most early studies in WSNs focused on the improvement of network performance based on this stationary sink architecture. However, this stationary sink data gathering paradigm suffers the following major drawbacks that degrade the network performance. One is the *single sink neighborhood problem*, where the sensors within one-hop from the sink have to relay data for other remote sensors. As a result, these sensors consume much more energy than that of the others. Once they run out of energy, the network is partitioned and the sink will be disconnected from these remote sensors. In particular, with the increase of network size, the single sink neighborhood problem becomes worse. Another is network connectivity. It is compulsory that a network consisting of a stationary sink and the sensors should be connected; otherwise, the data generated by the sensors in a fragment different from the one in which the sink is located will not be collected by the sink ultimately. In some sparse sensor deployment scenarios, it is very difficult to ensure that all sensors and the sink are in a single fragment due to the restriction of physical obstacles or other geographic constraint (e.g., a water pond) on the sink deployment.

To cope with the single sink neighboring problem, one strategy is to deploy a mobile sink instead of stationary sink for data gathering. Thus, the energy consumption of each sensor can be balanced through the motion of the sink. Also, each sensor can send its data to the sink within a certain number of hops when the sink approaches the sensor. Consequently, the relaying data workload of each sensor will be decreased, and its energy consumption will be reduced. In this paper, we aim to find a trajectory for the mobile sink such that the network lifetime is maximized, subject to the following constraints: (i) the space of potential sojourn locations of the mobile sink. In practice, whether the mobile sink will sojourn at a location is determined by several factors. For instance, it is inappropriate to let the sink stop at a barrier location that obstructs the wireless communication between itself and sensors or an inaccessible location like a water pond. Therefore, we assume that the sink is only allowed to stop at some anchor locations and we refer to these locations as the *potential sink locations*. Such information is usually given by users *a priori*. (ii) The tolerant delay on data delivery. As the buffer size of each sensor is very small, the amount of data it can store therefore is limited. To minimize the data loss due to data overflow in the buffer, the maximum tolerant data delivery delay for each sensor must be bounded. To establish the relationship between the network lifetime and the tolerant delay on data delivery, a controllable parameter h will be used, which is the upper bound on the number of hops from each

978-0-7695-4707-7/12 $26.00 © 2012 IEEE
DOI 10.1109/DCOSS.2012.17

sensor to its nearest sink location. The choice of h thus plays a key role in exploring the tradeoff between the network lifetime and the delay on data delivery. Intuitively, the smaller the value of h, the more the number of sojourn locations, and the longer the trajectory will be. Consequently, a longer delay on data delivery follows. In this paper we strive to achieve the finest tradeoff between the network lifetime and the tolerant delay on data delivery when employing a mobile sink for data gathering. Specifically, we aim to find an optimal trajectory for the mobile sink and to devise a routing protocol that routes sensing data to the mobile sink when the sink traverses along the trajectory such that the network lifetime is maximized, subject to the maximum tolerant delay on data delivery. The novelty of this study provides a joint optimization framework that strives a non-trivial tradeoff between the network lifetime and the tolerant delay on data delivery.

The main contributions of this paper are as follows. We first formulate a joint optimization problem, namely the hop-constrained mobile sink problem. Due to the NP-hardness of the problem, we then propose a novel optimization framework, consisting of finding a feasible trajectory for the mobile sink and devising an energy-efficient routing protocol for data collection. The proposed optimization framework not only improves network performance including the network lifetime, network scalability, and the average delay on data delivery, but also enhances the robustness of the network, since all sensing data generated by sensors can be collected by the mobile sink regardless of network connectivity. We finally conduct extensive experiments by simulations to evaluate the performance of the proposed algorithm against the other heuristics. The experimental results demonstrate that the proposed algorithm outperforms the others significantly in terms of network lifetime prolongation.

The remainder of the paper is organized as follows. Section II provides the literature survey on the sink mobility for network lifetime prolongation. Section III introduces the system model and defines the problem precisely. Section IV proposes a novel heuristic algorithm for the problem, and Section V evaluates the performance of the proposed algorithm through experimental simulations, and Section VI concludes the paper.

II. Related work

Extensive studies of network lifetime maximization by exploiting sink mobility have been conducted in the past several years. For example, Luo and Hubaux [14] formulated the network lifetime maximization problem into a min-max problem in a circle with sensor uniform-distribution and derived a nice solution. Gandham *et al.* [7] made the first attempt to determine specific sink movements for energy conservation, by employing $K \geq 1$ mobile sinks to collect sensing data for a monitoring region. They presented an ILP (integer linear program) model to determine the locations of the K mobile sinks within one round with an objective to minimize the maximum energy consumption among nodes or the total energy consumption, assuming that the potential sink location space is given. Wang *et al* [20] considered a joint optimization problem of determining the sink movement and its sojourn time at certain nodes in a grid network so that the network lifetime is maximized. They proposed an ILP solution for the problem, assuming that a half the workload of each node flow along its horizontal and another half flow along its vertical links towards the current location of the sink. Luo *et al.* [15] later considered a joint optimization problem of data gathering by proposing a two-stage scheduling: First, the mobile sink visits the "anchor" locations one by one and sojourns at each of them for a short sampling period. During this stage, the sink collects the power consumption of all nodes and builds the sojourn time profile at the anchor point. The sink then solves an ILP formula, using the given sojourn time profiles. Basagni *et al.* [3] considered a more realistic model for network lifetime maximization by incorporating two realistic constraints on mobile sinks: the maximum distance at its each movement and the minimum sojourn time at each sojourn location. To reduce the data loss due to the sink motion from one location to another, it is assumed that the moving distance of the sink from its current location to its next one is bounded, and the sink sojourns at each chosen location for at least a certain amount of time. Then the problem is to find a trajectory for the sink that maximizes the network lifetime. They first formulated the problem as a mixed integer linear program, and then proposed a simple, distributed heuristic.

In contrast, to eliminate the energy consumption of sensors on relaying data, Sugihara and Gupta [18], [19] considered one-hop data collection and the travel time scheduling of a single mobile sink. They formulated the problem as a traveling salesman problem and schedule the travel time at each edge in the tour to maximize the amount of collected data. Xing *et al.* [21] proposed a rendezvous-based data collection approach that exploits the controlled sink mobility and the capacity of in-network data caching through bounding the total travel distance of the mobile sink. They developed two approximation algorithms to minimize the sum of the consumed energy of all involved sensors, assuming that the collected data at each relay node is aggregated into a single packet prior to its transmission. Guney *et al.* [9] formulated the sink trajectory problem as a joint optimization problem that aims to identify the optimal sink locations and information flow path between sensors and sinks. They formulated the problem as a mixed integer linear programming and developed several heuristics for it. Liang *et al.* [11], [12] incorporated the travel distance of the mobile sink into the network lifetime maximization problem and proposed heuristics. Liang *et al.* [10] later extended their work on a single mobile sink to multiple sinks. Gatzianas and Georgiadis [8] formulated the optimal trajectory problem for a mobile sink as a linear programing problem and provided a distributed solution by utilizing Lagrangian duality and the sub-gradient method. Note that the running time of the proposed distributed algorithm depends on the algorithm's convergence rate. Yun and Xia [23] considered the network lifetime maximization problem, assuming that each sensor does not require to send its data immediately when they are generated, instead the data can be stored at the sensor temporarily and be transmitted when the mobile sink is at the most favorable location to achieve the maximum network lifetime. They formulated the problem as a mixed integer programming problem, subject to the bounded delay constraint, and proposed a flow-based optimization framework.

III. Preliminaries

A. System model

In this paper we consider a wireless sensor network $G = (\mathcal{N} \cup \mathcal{L}, \mathcal{E})$ consisting of $n = |\mathcal{N}|$ stationary sensors, where $\mathcal{L}$ is the set of potential sink locations. Notice that for a given network, we assume that the mobile sink only stops at some strategic locations and avoids obstacles like big rocks, water ponds, and so on. $\mathcal{N}$ is the set of sensors. $\mathcal{E}$ is the set of links. There is a link between two sensors or a sensor and the sink if they are within the transmission range of each other, assuming that $|\mathcal{L}| < n$. For the sake of simplicity, we assume that the transmission ranges of both the sink and the sensors are identical. The locations of sensors are fixed and known *a priori*. Assume that all sensors have identical data generation rates r_a. We further assume that the mobile sink has unlimited energy supply and can sojourn at any location in $\mathcal{L}$ for data gathering. Without loss of generality, for each sensor node, only its energy consumption on wireless communication will be considered, i.e., the energy consumption on data transmission and data reception will be taken into account, its other energy consumptions including sensing and computation will be ignored [17]. The network lifetime is defined as the first sensor failure time due to its energy depletion [4].

Given a potential sink location $s \in \mathcal{L}$, let $N_1(s) = \{u \mid (u, s) \in \mathcal{E}, u \in \mathcal{N}\}$ be the set of one-hop neighboring sensors of the sink located at s and $N_h(s)$ the set of neighboring sensors of the sink at s within h-hop, where $N_h(s) = \{v \mid$ the number of hops from $v \in \mathcal{N}$ to s is no more than $h\}$. Given any sink location $s \in \mathcal{L}$, the calculation of $N_h(s)$ is as follows. A partial Breath-First-Search tree rooted at s is constructed layer by layer (hop by hop), and the expansion continues until all nodes in layer h are explored. The set of sensors contained in the partial BFS tree is refereed to as $N_h(s)$. To bound the delay on data delivery, we assume that there is a given value D_{max}, which is the maximum tolerant delay of any sensor. For a given sensor $v_i \in \mathcal{N}$, its buffer size Buff usually is fixed, the maximum amount of data it can hold without overflow thus is $r_a t_i \leq$ Buff, then, the maximum tolerant delay on data delivery by sensor v_i without data loss is upper bounded by $t_i \leq \frac{\text{Buff}}{r_a}$, $1 \leq i \leq n$. Thus, we may set $D_{max} = \min\{\ t_i \mid 1 \leq i \leq n\}$ if no data loss is expected when the mobile sink traverses along the trajectory for data gathering.

B. Problem definition

The hop-constrained mobile sink problem in a wireless sensor network $G(\mathcal{N} \cup \mathcal{L}, \mathcal{E})$ is to find an optimal trajectory for the mobile sink consisting of sink locations in $\mathcal{L}$ such that the network lifetime is maximized, subject to the following constraints: (i) each sensor can send its data to one sink location in the trajectory within $h \geq 1$ hops; (ii) the maximum tolerant delay on data delivery is bounded by a given value D_{max}, or the length of the trajectory bounded by L, assuming that the sink travels at a constant speed v, i.e., $D_{max} = \frac{L}{v}$.

To collect sensing data from sensors to the mobile sink, we here adopt the forest consisting of routing trees, where each tree can be treated as cluster and the tree root is the cluster head. A routing tree rooted at a sink location s is formed by including all sensors in $N_h(s)$. Thus, the sensors in G are organized into a forest of routing trees, in which each tree is rooted at a sink location on the trajectory of the mobile sink, and the depth of each tree is bounded by a parameter h, the sink only sojourns at tree roots (sink locations), and the total travel time of the sink except sojourning is bounded by D_{max}. Assume that the mobile sink is able to collect all the data stored at each cluster on no time when it approaches the cluster head. Notice that parameter h is a controllable parameter, which quantifies the extent of multi-hop routing and represents the maximum number of hops from a sensor to the sink. In other words, assuming that there are $|\mathcal{L}|$ potential sink locations $\mathcal{L} = \{s_1, s_2, \ldots, s_{|\mathcal{L}|}\}$, the hop-constrained mobile sink problem is to find a trajectory consisting of sink locations in a subset $S' \subseteq \mathcal{L}$, such that the network lifetime is maximized, provided that the trajectory length is bounded by L, each sensor is no more than h hops from its root, and all sensors are *covered* by the sink locations in the trajectory, *i.e.*, $\bigcup_{s \in S'} N_h(s) = \mathcal{N}$, where a sensor is "covered" if the number of hops from it to one of the sink locations in the trajectory is no more than h. Let $k = |S'|$, the sensors in the network are partitioned into $k = |S'|$ clusters, and a routing tree T_s rooted at cluster head $s \in S'$ will be used for collecting the sensing data in the cluster.

Since we assume that there are unlimited energy supplies to the mobile sink in comparison with the initial capacity of sensors, the network lifetime will be determined by the energy consumption of sensors, while the sensors near to a sink location are usually the *bottleneck sensors* in terms of energy consumption, since they have to relay data for other remote sensors and their energy consumption on relaying is proportional to the amount of data relayed. To prolong the network lifetime is thus to balance the energy consumption (workload) of bottleneck sensors. Let $\mathcal{F} = \{T_s \mid s \in S'\}$ be the forest of routing trees, where T_s is a routing tree rooted at sink location s. Denote by $dt_{T_s}(v)$ the number of descendants of sensor $v \in \mathcal{N}$ in T_s and $C_T(s)$ the set of children of s in T_s. Recall that the data generation rate of each sensor is r_a. Then, the energy consumption of sensor v in T_s on wireless communication per time unit is

$$ec_{T_s}(v) \quad = \quad r_a \cdot [(dt_{T_s}(v) + 1) \cdot e_t + dt_{T_s}(v) \cdot e_r], \quad (1)$$

when the sink is located at s, where e_t and e_r are the amounts of power consumed by transmitting and receiving a unit-length of data, assuming that no data aggregation at each relay node is allowed. It can be seen that the value of $ec_{T_s}(v)$ is proportional to the value of $dt_{T_s}(v)$.

The hop-constrained mobile sink problem is NP-hard, by considering one of its special cases where $h = 1$, $\mathcal{L} = \mathcal{N}$ and the length L of the sink trajectory is given, the problem becomes the Traveling Salesman Problem with Neighborhoods (TSPN) that a salesman sends his products to his customers directly or to the neighbors of the customers with an aim to minimize the total travel cost. It is well known that TSPN is NP-complete [2], the problem of concern thus is NP-hard, too.

IV. Heuristic algorithm

Since the problem is NP-hard, we instead propose a novel heuristic for it that takes both the maximum tolerant delay and the network lifetime into consideration through a controllable parameter h. To this end, we decouple this joint optimization problem into two following subproblems: finding a feasible trajectory and devising an energy-efficient routing protocol.

A. *Finding a hop-constrained trajectory*

Given an integer $h \geq 1$, let $\mathcal{C} = \{N_h(s) \mid s \in \mathcal{L}\}$ be the collection of sets derived from set $\mathcal{L}$ of potential sink locations, in which each set $N_h(s)$ is a h-hop neighboring set of the mobile sink at location s. The proposed algorithm proceeds iteratively. Within each iteration, a new sink location will be added to the trajectory if the resulting trajectory still meets the delay constraint. In case there are multiple sink locations to choose, choose one with the maximum 'benefit' with respect to (w.r.t) the current trajectory, where the benefit of a sink location *w.r.t* the current trajectory will be defined later. This procedure continues until either the delay constraint is violated or all the sensors are covered.

Recall that a sensor $v \in \mathcal{N}$ is *covered* by the mobile sink at location s if the number of hops from v to s is no more than h; otherwise, v is *uncovered* if none of any sink locations in the trajectory can cover it. Given a solution, if not all sensors are covered, the solution is *infeasible*. To obtain a feasible solution to the problem, the value of h is increased by one and the above procedure is applied again, it continues until a feasible solution obtained. Specifically, initially all sensors in $\mathcal{N}$ are *uncovered* and the set of the chosen sink locations S' consists of the depot location only, assuming no sensor is covered by the mobile sink at the depot. Within each iteration, the proposed algorithm selects a sink location s in $\mathcal{L} - S'$ if it brings the maximum *benefit*, $b(s)$, which is defined as follows.

$$b(s) = \max_{s' \in S', s \in \mathcal{L}-S'} \{\frac{|N_h(s) - N_h(s) \cap N_h(S')|}{d(s, s')}\}, \qquad (2)$$

where $d(s, s')$ is the Euclidean distance between s and s' and $N_h(S') = \cup_{s' \in S'} N_h(s')$ is the set of covered sensors by sink locations in S'. Once s is chosen, every sensor in $N_h(s)$ now is covered by s if it has not been covered yet.

As a result, a feasible solution is obtained, which can be further refined by adding as many sink locations as possible to it until the delay constraint is violated, because the more the sink locations added to the trajectory, the less the load among bottleneck sensors share, and the longer the network lifetime will be. The detailed algorithm for finding the optimal trajectory will be shown in algorithm `Find_Feasible_Tour`, in which there are two procedures: `Test_Feasible` which returns either a "true" or a " fail" value by examining whether an obtained solution is feasible; and `Augment` which refines the feasible solution (a found trajectory) further by adding more sink locations as long as the delay constraint still holds. These procedure are described as follows.

Algorithm 1 *Test_Feasible*

Input: T_1, S_1, D_{max};
Output: '$true'$ or '$false'$.

Find an MST T_1 in a complete graph $K[S_1]$ in S_1, where the weight of an edge is their Euclidean distance;
Construct a bipartite graph $G_B = (X, Y, E_B)$ based on T_1, where X and Y are sets of odd-degree vertices in T_1. There is an edge E_B between $x \in X$ and $y \in Y$ with weight $d(x, y)$ if $x \neq y$;
Find a minimum weighted perfect matching M in G_B;
Find a trajectory with length L' derived from $T_1 \cup M$ such that each location appears in L' once;
/* Let $D(L') = \frac{L'}{v}$ be the delay of L' */;
if $D(L') > D_{max}$ **then**
  return $false$
else
  return $true.$
end if

Algorithm 2 *Augment*

Input: L', S', D_{max};
Output: L', S'.

$S \leftarrow \mathcal{L} - S'$;
$indicator \leftarrow' true'$; /* can the trajectory be extended? */
while $indicator$ **do**
  Choose a location $s \in S$ such that $\triangle L(s) = min_{s' \in S}\{L(T_{MST}[S' \cup \{s'\}]) - L(T_{MST}[S'])\}$, where $L(T_{MST}[V'])$ is the trajectory length derived from an MST of $K[V']$ by nodes in V'. Clearly, $L' = L(T_{MST}[S'])$;
  $L'' \leftarrow L(T_{MST}[S' \cup \{s'\}])$;
  if $D(L'') < D_{max}$ **then**
    $S \leftarrow S - \{s\}$; $S' \leftarrow S' \cup \{s\}$; $L' \leftarrow L''$;
  else
    $indicator \leftarrow' false'$;
  end if
end while
return (L', S').

Having the above two routines, we now provide the detailed description of algorithm `Find_Feasible_Tour`. For convenience, we refer to this algorithm as `FFT`.

Algorithm 3 *Find_Feasible_Tour*

Input: $\mathcal{N}, \mathcal{L}, D_{max}$;
Output: A solution (L', S', h).

1: $h \leftarrow 1$;
2: $T \leftarrow \emptyset$ /* The trajectory is derived by traversing a tree T and performing shortcut, using the triangle inequality */
3: $S \leftarrow \mathcal{L}$; /* set of available sink locations */
4: $flag \leftarrow' true'$; /* is the solution feasible? */
5: **while** $flag$ **do**
6: $\quad U \leftarrow \mathcal{N}$; /* set of uncovered sensors */
7: $\quad S' \leftarrow \{s_0\}$; /* set of sink locations */
8: $\quad \mathcal{C} \leftarrow \{N_h(s) \mid s \in \mathcal{L}\}$;
9: $\quad$ **while** $(U \neq \emptyset)$ and $(S \neq \emptyset)$ **do**
10: $\quad\quad$ choose a location $s \in S$ with maximum benefit $b(s)$ by Eq. (2);
11: $\quad\quad$ **if** *Test_Feasible*$(T \cup \{(s', s)\}, S' \cup \{s\}, D_{max})$ **then**
12: $\quad\quad\quad T \leftarrow T \cup \{(s', s)\}$; /* Add the edge to T, where s' is such a location in S' that leads to the maximum benefit of location s */
13: $\quad\quad\quad S' \leftarrow S' \cup \{s\}$; $U \leftarrow U - N_h(s)$; $S \leftarrow \mathcal{L} - S'$;
14: $\quad\quad$ **else**
15: $\quad\quad\quad S \leftarrow S - \{s\}$;
16: $\quad\quad$ **end if**
17: $\quad$ **end while**
18: $\quad$ **if** $(U = \emptyset)$ **then**
19: $\quad\quad flag \leftarrow' false'$; /* all sensors are covered */
20: $\quad\quad$ Let L' be the length of a trajectory derived from an MST of the complete graph $K[S']$;
21: $\quad\quad$ **call** *Augment* (L', S', D_{max});
22: $\quad$ **else**
23: $\quad\quad h \leftarrow h + 1$; /* an infeasible solution */
24: $\quad$ **end if**
25: **end while**
26: **return** a solution (L', S', h);

In the following we analyze the running time of algorithm `FFT`.

Lemma 1: Given a wireless sensor network $G(\mathcal{N} \cup \mathcal{L}, \mathcal{E})$ and a tolerant delay of data delivery D_{max}, algorithm `FFT` can deliver a feasible solution that takes $O(n \cdot l^3 \log D + m \cdot l \log D)$ time, where D is the diameter of the network, $l = |\mathcal{L}|$, $n = |\mathcal{N}|$, and $m = |\mathcal{E}|$.

Proof: We analyze the time complexity of algorithm `FFT` by assuming that h is fixed first. Following the proposed algorithm, the calculation of the collection $\mathcal{C}$ at Step 8 takes $O((m+n)l)$ time. Step 10 takes $O(l^2 n)$ time, while the dominant time step is Step 11 calling routine `Test_Feasible` for finding a perfect matching in G_B, which takes $O(l^3)$ time. The number of iterations between Step 9 to Step 17 is l. Routine `Augment` takes $O(l^4)$. Thus, algorithm `FFT` takes $O(l^3 \cdot n \cdot h + l^4 \cdot h + l \cdot m \cdot h)$ time. As can be seen, when h approaches the diameter D of the network, only one sink location suffices, which will cover all sensors within D hops, and the algorithm will terminate. However, the running time of algorithm `FFT` can be further improved by a binary search on h in order to find a minimum h to meet the delay constraint D_{max}, $1 \leq h \leq D$. Thus, algorithm `FFT` takes $O(l^3 \cdot n \log D + m \cdot l \log D)$ time since $l \leq n$. ■

B. A routing protocol for data gathering

Let S' be the set of chosen sink locations in the found trajectory by algorithm `FFT` with $k = |S'|$. We now build a forest of load-balanced routing trees to minimize the energy consumption among the sensors with the aim of maximizing the network lifetime.

Notice that the data generation rate r_a of each sensor is identical, the energy consumption of each sensor is proportional to the number of its descendants in the routing tree rooted at s when the mobile sink is located at s, following Eq. (1). To maximize the network lifetime is equivalent to minimize the maximum energy consumption among bottleneck sensors. To do so, a forest consisting of load-balanced routing trees will be built. Each routing tree rooted at a sink sojourn location has the two properties: (i) the number of hops from each sensor of it to the root is no more than h; and (ii) the maximum energy consumption among the children of the root is minimized. Such a load-balanced routing tree can be constructed by an algorithm in [22], which briefly is described as follows. A virtual node r is created and the k sink locations in the found trajectory are compressed into the virtual node, any neighbor (sensor) of the sink at a chosen location in the original network now becomes a neighbor of r, a load-balanced routing tree T in the resulting network rooted at r is then found.

To find k load-balanced routing trees rooted at the k chosen sink locations such that the load among them is balanced, we reduce this problem to a load-balanced semi-matching problem [13] which is defined as follows. Given a node-weighted bipartite graph $G_{XY} = (X, Y, E_{XY}, w)$, where X and Y are the sets of nodes, E_{XY} is the set of edges between nodes in X and Y, and w is a non-negative weight function on nodes in Y, assuming $|X| \leq |Y|$. The load-balanced semi-matching problem in G_{XY} is to find a semi-matching M_{XY} such that each node $y \in Y$ has a parent node $x \in X$ and the maximum weight among the nodes in X is minimized, where the weight of a node $x \in X$ is $w(x) = \sum\{w(y_i) \mid y_i \in Y, (x, y_i) \in M_{XY}\}$. Note that the load-balanced semi-matching problem is NP-hard, and there is an approximation algorithm for it [13].

To reduce our problem to the load-balanced semi-matching problem, a node-weighted bipartite graph $G_B = (S', V_1, E', w')$ is constructed, where V_1 is the set of children of the virtual node r in T. There is an edge $(s, v) \in E'$ if the sink at location $s \in S'$ is within the transmission range of sensor $v \in V_1$. For each $v \in V_1$, $w'(v)$ is the number of nodes in *the subtree* T_v of T rooted at v including node v itself. Let M_B be an approximate solution of the load-balanced semi-matching in graph G_B by applying the approximation algorithm due to Low [13]. Then, for each matched sensor v which is an endpoint of a matched edge in M_B, its another endpoint in S' is the root of a load-balanced tree, and the subtree T_v of T rooted at v will be part of the load-balanced routing tree. Thus, the sensors in the network are partitioned into k load-balanced routing trees rooted at the k chosen sink locations. Each sensor can relay data to its root (a sink location) within h hops, and the load at each sink location is well balanced. In summary, algorithm `Find_Load_Bala_Forest` for finding an energy-efficient routing protocol is described as follows.

Algorithm 4 *Find_Load_Bala_Forest*

Input: $\mathcal{N}, S', h$;

Output: A load balanced forest with the trees rooted at chosen sink locations.

1: Construct a new network G' by compressing all nodes in S' into a virtual node r, and all neighbors of nodes in S' become the neighbors of r in G';
2: Construct an approximate, load-balanced tree T in G' rooted at r, using an algorithm by Xu and Liang [22];
3: Construct a node-weighted bipartite graph G_B;
4: Find an approximate, load-balanced semi-matching M_B in G_B, using an algorithm by Low [13]. Thus, each sensor matches with a sink location in S';
5: Form a load-balanced tree rooted at s for each $s \in S'$ by merging related subtrees according to the matched edges;
6: **return** $\{T_s \mid s \in S'\}$.

We thus have the following theorem.

Theorem 1: Given a wireless sensor network $G(\mathcal{N} \cup \mathcal{L}, \mathcal{E})$ and the maximum tolerant delay on data delivery D_{max}, there is an algorithm for the hop-constrained mobile sink problem in G, which takes $O(l^3 \cdot n \log D + m \cdot l \log D + m \cdot n \log n)$ time, where D is the diameter of the network, $n = |\mathcal{N}|$ is the number of sensors, and $m = |\mathcal{E}|$ is the number of links.

Proof: Following Lemma 1, the trajectory finding takes $O(l^3 \cdot n \log D + m \cdot l \log D)$ time. The time complexity of algorithm `Find_Load_Bala_Forest` for finding a forest consisting of load-balanced routing trees is $O(mn \log n)$ time [22], the theorem follows. ■

For the sake of convenience, we refer to the proposed heuristic for the hop-constrained mobile sink problem as algorithm `HCMK` for short.

V. PERFORMANCE EVALUATION

In this section we evaluate the performance of the proposed algorithm for the hop-constrained mobile sink problem and investigate the impact of constraint parameters: the potential sink location space $\mathcal{L}$, the maximum number of hops from each sensor to a sink location h, and the maximum tolerant delay on data delivery D_{max}, on the network lifetime through experimental simulations.

A. Simulation environment

We consider a wireless sensor network consisting of 100 to 600 sensors which are randomly deployed in a $100m \times 100m$ square region in our default setting. The potential sink locations in $\mathcal{L}$ are also randomly generated with the default setting $|\mathcal{L}| = 100$. The transmission range R of each sensor is fixed at 25 meters and its initial energy capacity IE is $100 Jules$. In all our experiments we adopt the energy consumption parameters of real sensors - MICA2 motes [6], where $e_t = 14.4 \times 10^{-6} J/bit$ and $e_r = 5.76 \times 10^{-6} J/bit$. We assume that the data generation rate is $r_a = 1 bit/s$. Together with these parameters, the network lifetime can be calculated using Eq. (1). Each value in figures is the mean of the results by applying each mentioned algorithm to 15 different network topologies of the same size. We assume that the speed of the sink is fixed to be $1m/s$.

B. Other heuristics

We here propose the other two heuristics for the hop-constrained mobile sink problem, which will serve as the benchmark purpose. The only difference between these two heuristics and algorithm `HCMK` lies in the trajectory finding stage.

The first heuristic `Delete_Sink` is described as follows. Assume that an initial trajectory contains all locations in $\mathcal{L}$, which may not be feasible due to the violation of the delivery delay constraint. We modify this solution to make it become either a feasible solution or an infeasible solution. The algorithm keeps removing a *redundant sink location* iteratively from the current solution until either the delay on the resulting trajectory is no greater than D_{max} or no feasible solution exists. Specifically, it proceeds as follows. An initial solution is $S' = \mathcal{L}$, assuming that all sensors are covered by the sink locations in $\mathcal{L}$ within h hops with a certain h, $1 \leq h \leq n-1$. It then finds an MST $T_{MST}(K[S'])$ in a complete graph $K[S']$ induced by the nodes in S', the weight associated with each edge in $K[S']$ is the Euclidean distance between its two endpoints. It thirdly transforms $T_{MST}(K[S'])$ into a trajectory, consisting of nodes in S', using the triangle inequality and the minimum weighted perfect matching. It is well known that the length of the trajectory is at most twice of the weighted sum of the edges in $T_{MST}(K[S'])$. Recall that $L(T_{MST}(K[S']))$ is the length of the trajectory transformed from tree $T_{MST}(K[S'])$. It then checks whether the delay constraint is violated. If not, the solution is a feasible solution, we are done. Otherwise, more sink locations in the current solution need to be removed. To this end, we check whether there is a redundant sink location in the solution, where a sink location is referred to *redundant* if the sensors covered by it within h-hops are still been covered by the other sink locations in the solution, i.e., a sink location $s \in S'$ is a redundant sink location if $N_h(s) \cap N_h(S' - \{s\}) = N_h(s)$. If there is no such a redundant sink location, the procedure terminates and the solution is infeasible. We increase h by 1, and repeat the above procedure again. Otherwise, if there are multiple redundant sink locations to choose, choose the one s whose removal results in the maximum delay reduction $\nabla D(s)$, where

$$\nabla D(s) = \frac{L(T_{MST}K[S']) - L(T_{MST}K[S' - \{s\}])}{v} \quad (3)$$

It finally examines whether $\frac{L(T_{MST}(K[S'-\{s\}])}{v} \leq D_{max}$.

The second heuristic `Set_Cover` is almost identical to algorithm `Delete_Sink`. The only difference is the choice of set S'. Here S' is obtained by applying the set cover algorithm [5] to identify a subset of sink locations of set $\mathcal{L}$, assuming that $N_h(s)$ is given for all $s \in \mathcal{L}$. Let $S' \subseteq \mathcal{L}$ be the set of chosen locations, the rest is identical to algorithm `Delete_Sink`, omitted.

C. Impact of h, L, and $\mathcal{L}$ on network performance

In the following we evaluate the performance of the proposed heuristic against the other mentioned heuristics in terms of network lifetime. We study the impact of parameters h, the maximum tolerant delay D_{max}, the space of potential sink locations $\mathcal{L}$, and network size n on the network lifetime.

Impact of L and n on the network lifetime: We first investigate the impact of the length of the trajectory L and network size n on the network lifetime by varying L from 100 to 350 with the increment of 50. Given the maximum tolerant delay D_{max} on data delivery, as $D_{max} = \frac{L}{v}$ and the speed v of the mobile sink is fixed, we can use the length of the trajectory L instead of D_{max} to evaluate its impact on the network lifetime. Fig. 1 illustrates the network lifetime under different values of L and n.

Fig. 1 shows that a longer delay leads to a longer network lifetime since a longer delay implies that there are more sink locations in the trajectory and consequently there are fewer hops from each sensor to the root of its routing tree, which is further illustrated by Fig. 2 as follows.

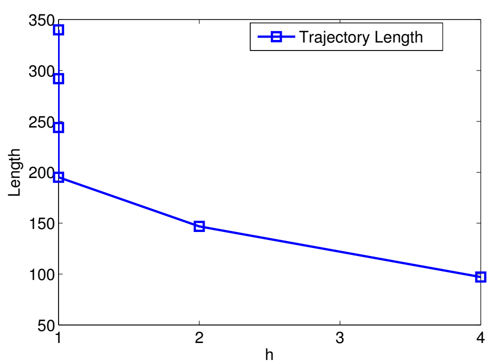


Fig. 2. Impact of parameter h on the trajectory length

Fig. 3 plots the performance curves of different algorithms when n is fixed at 300, from which it can be seen that algorithm `HCMK` achieves a much longer network lifetime than both algorithm `Delete_Sink` and algorithm `Set_Cover`.

Impact of hop h and n on the network lifetime: We then study the impact of the number of hops h and the network size n on the network lifetime, and Fig. 4 shows that when n is fixed, a larger h will lead to a shorter network lifetime since each children of the root of each routing tree will bear a heavier load. Similarly, with the increase of network size n while fixing h, the network lifetime drops, too.

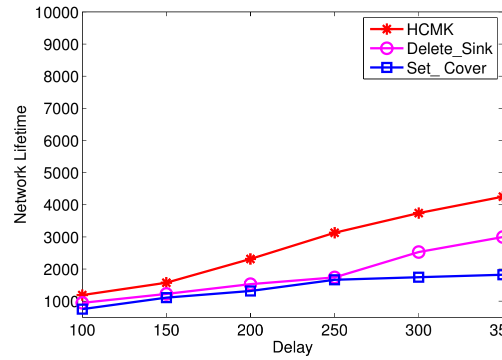


Fig. 3. Performance of different algorithms when $n = 300$.

Fig. 5 plots the performance curves of different algorithms when h is fixed to 3, from which it can be seen that the network lifetime delivered by algorithm `HCMK` is a much longer than these of both algorithm `Delete_Sink` and algorithm `Set_Cover` when the stringent delay constraint is imposed.

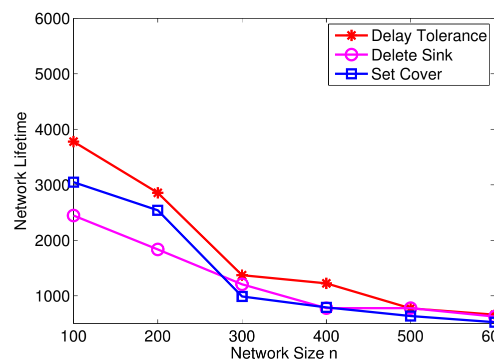
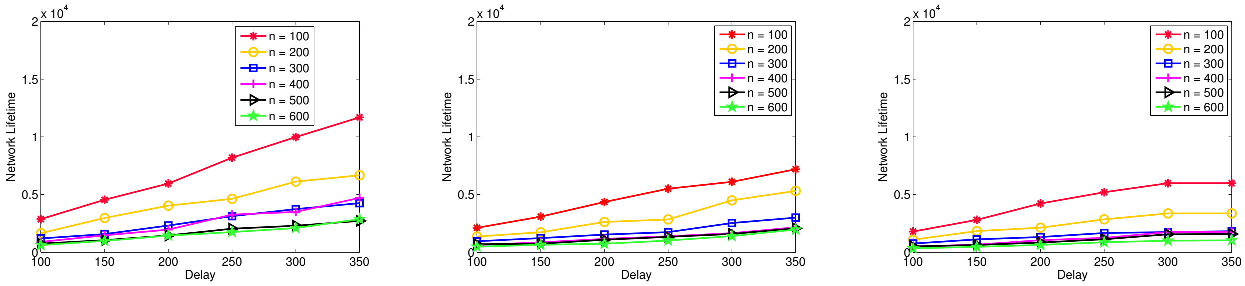


Fig. 5. Performance of different algorithms when $h = 3$.

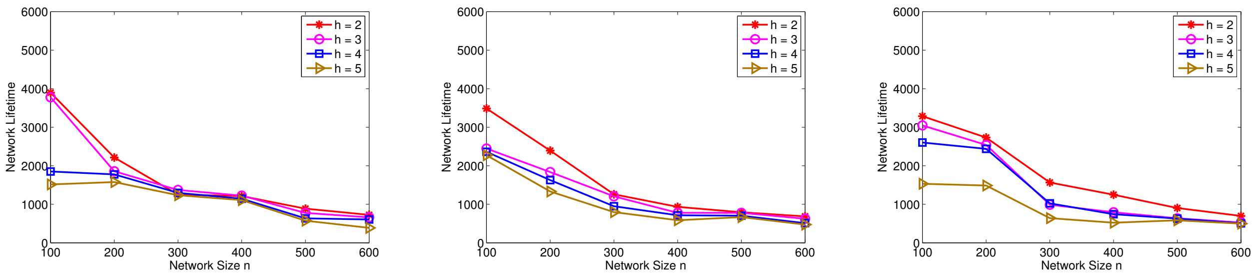
VI. Conclusion

In this paper we have studied the tradeoff between the tolerant delay on data delivery and the network lifetime in a wireless sensor network with a mobile sink, by proposing a novel joint optimization framework. Due to the NP-hardness of the problem, we then devised a heuristic consisting of finding a hop-constrained trajectory for the mobile sink and an energy-efficient routing protocol for routing sensing data to the mobile sink when it traverses along the trajectory, subject to the maximum tolerant delay on data delivery. We finally conducted extensive experiments by simulations to evaluate the performance of the proposed algorithm against the other heuristics. The experimental results demonstrate that the proposed algorithm outperforms the others significantly in terms of network lifetime prolongation.



(a) Algorithm `HCMK` (b) Algorithm `Delete_Sink` (c) Algorithm `Set_Cover`

Fig. 1. Impact of L (or D_{max}) and n on the network lifetime.



(a) Algorithm `HCMK` (b) Algorithm `Delete_Sink` (c) Algorithm `Set_Cover`

Fig. 4. Impact of h and n on the network lifetime

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, Vol. 38, pp. 393–422, 2002.

[2] E. Arkin and R. Hassin. Approximation algorithms for the geometric covering salesman problem. *Discrete Applied Math.*, Vol. 55, pp. 197–218, 1994.

[3] S. Basagni, A. Carosi, E. Melachrinoudis, C. Petrioli, and Z. M. Wang. Controlled sink mobility for prolonging wireless sensor networks lifetime. *Wireless Networks*, Vol. 14, pp.831–858, 2008.

[4] J-H Chang and L. Tassiulas. Energy conserving routing in wireless ad hoc networks. *Proc. INFOCOM'00*, IEEE, 2000.

[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein. *Introduction to Algorithms*. 3rd Edition, MIT Press, 2009.

[6] Crossbow Inc. MPR-Mote Processor Radio Board User's Manual.

[7] S. R. Gandham, M. Dawande, R. Prakask, and S. Venkatesan. Energy efficient schemes for wireless sensor networks with multiple mobile base stations. *Proc. Globecom'03*, IEEE, 2003.

[8] M. Gatzianas and L. Georgiadis. A distributed algorithm for maximum lifetime routing in sensor networks with mobile sink. *IEEE Trans. Wireless Communications*, Vol.7, pp984–994, 2008. .

[9] E. Guney, N. Aras, I. L. Altinel, and C. Ersoy. Efficient integer programming formulations for optimum sink location and routing in heterogeneous wireless sensor networks. *Computer Networks*, 2010.

[10] W. Liang and J. Luo. Network lifetime maximization in sensor networks with multiple mobile sinks. To appear in *Proc LCN'11*, IEEE, Oct., 2011.

[11] W. Liang, J. Luo, and X. Xu. Prolonging network lifetime via a controlled mobile sink in wireless sensor networks. *Proc Globecom'10*, IEEE, Dec., 2010.

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

[13] C. P. Low. On load-balanced semi-matchings for weighted bipartite graphs. *Proc of TAMC*, LNCS, Vol. 3959, pp.159–170, 2006.

[14] J. Luo and J-P Hubaux. Joint mobility and routing for lifetime elongation in wireless sensor networks. *Proc of INFOCOM'05*, IEEE, 2005.

[15] J. Luo, J. Panchard, M. Piorkowski, M. Grossblauser, and J-P Hubaux. Mobiroute: routing towards a mobile sink for improving lifetime in sensor networks. *Proc of DCOSS'06*, LNCS, Vol. 4026, pp. 480–497,2006.

[16] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson. Wireless sensor networks for habitat monitoring. *The 1st Intl Workshop on Wireless Sensor Networks and Applications (WSNA)*, ACM, 2002.

[17] G. J. Pottie. Wireless sensor networks. *Proc. of Information Theory Workshop*, pp 139–140, 1998.

[18] R. Sugihara and R. K. Gupta. Improving the data latency in sensor networks with controlled mobility. *Proc. of DCOSS'08*, IEEE, 2008.

[19] R. Sugihara and R. K. Gupta. Optimizing energy-latency trade-off in sensor networks with controlled mobility. *Proc. of INFOCOM'09*, IEEE, 2009.

[20] Z. M. Wang, S. Basagni, E. Melachrinoudis, and C. Petrioli. Exploiting sink mobility for maximizing sensor networks lifetime. *Proc. HICSS*, IEEE, 2005.

[21] G. Xing, T. Wang, W. Jia, and M. Li. Rendezvous design algorithms for wireless sensor networks with a mobile base station. *Proc. of MobiHoc'08*, ACM, pp.231–240, 2008.

[22] X. Xu and W. Liang. Placing optimal number of sinks in sensor networks for network lifetime maximization. *Proc. of ICC'11*, IEEE, 2011.

[23] Y. Yun and Y. Xia. Maximizing the lifetime of wireless sensor networks with mobile sink in delay-tolerant applications. *IEEE Trans Mobile Computing*, Vol. 9, No.9, pp.1308–1318, 2010.