WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. 2013; **13**:1263–1280
Published online 15 August 2011 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/wcm.1179

RESEARCH ARTICLE

Network lifetime maximization for time-sensitive data gathering in wireless sensor networks with a mobile sink

Weifa Liang[1*], Jun Luo[2] and Xu Xu[1]

[1] Research School of Computer Science, The Australian National University, Canberra, ACT 0200, Australia
[2] School of Computer Science, National University of Defense Technology, Changsha, Hunan, China

ABSTRACT

With the advances of more and more mobile sink deployments (e.g., robots and unmanned aerial vehicles), mobile sinks have been demonstrated to play an important role in the prolongation of network lifetime. In this paper, we consider the network lifetime maximization problem for time-sensitive data gathering, which requires sensing data to be sent to the sink as soon as possible, subject to several constraints on the mobile sink. Because the mobile sink is powered by petrol or electricity, its maximum travel distance per tour is bounded. The mobile sink's maximum moving distance from its current location to the next must also be bounded to minimize data loss. As building a new routing tree rooted at each new location will incur an overhead on energy consumption, the mobile sink must sojourn at each chosen location at least for a certain amount of time. The problem, thus, is to find an optimal sojourn tour for the mobile sink such that the network lifetime is maximized, which is subject to a set of constraints on the mobile sink: its maximum travel distance, the maximum distance of each movement, and the minimum sojourn time at each sojourn location. In this paper, we first formulate this novel multiple-constrained optimization problem as the *distance-constrained mobile sink problem* for *time-sensitive* data gathering. We then devise a novel heuristic for it. We finally conduct extensive experiments by simulation to evaluate the performance of the proposed algorithm. The experimental results demonstrate that the performance of the proposed algorithm is very promising, and the solution obtained is fractional of the optimal one. Copyright © 2011 John Wiley & Sons, Ltd.

KEYWORDS
wireless sensor networks; sink mobility; data gathering; network lifetime maximization; distance-constrained shortest path; multiple-constrained joint optimization; load-balanced tree construction; network flow; sojourn time schedule

***Correspondence**
Weifa Liang, Research School of Computer Science, The Australian National University, Canberra, ACT 0200, Australia.
E-mail: wliang@cs.anu.edu.au

1. INTRODUCTION

Wireless sensor networks consist of several hundreds to thousands of battery-powered tiny sensors that are endowed with a multitude of sensing modalities including multimedia (e.g., video and audio) and scalar data (e.g., temperature, pressure, light, and infrared). The strong demand for these networks is spurred by numerous applications that require *in situ*, unattended, high-precision, and real-time observation over the monitored region [1]. 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 wireless sensor networks. In maximizing the network lifetime, energy conservation is of paramount importance.

Most existing studies on data gathering in wireless sensor networks assumed that there is a fixed base station (sink). The sensed data is relayed to the sink through multihop relays. The sensors near the sink, thus, become the bottlenecks of energy consumption because of the relaying of messages for others. Once they run out of energy, the sink will be disconnected from the rest of the network, and the network service will be interrupted. New strategies that exploit the mobility of network components have been developed to prolong the network lifetime by mitigating the deficiency of the fixed sink. One strategy is to allow a fraction of sensors to be movable. However, because of the severe energy constraint on sensors, this approach is infeasible in most application scenarios where sensors are stationary and powered by tiny batteries. Another strategy is to employ mobile sinks instead of fixed sinks to collect data by traversing the monitoring area, and recent studies have demonstrated that use of mobile sinks can improve various network performance significantly. These include the network lifetime, network connectivity, network throughput, and so on [2–13]. A comprehensive survey on this topic can be found in [14].

1.1. Related work

Sink mobility can be further classified into *uncontrolled mobility* [4,10] and *controlled mobility* [5–7,15]. For the former, mobile sinks can move randomly in the monitoring region, whereas for the latter, they can only move along pre-defined trajectories. Recent advance in *uncontrolled mobility* showed that if a mobile sink can sojourn at all given locations, there is a polynomial solution for network lifetime maximization [10]. However, handling *controlled mobility* is much more challenging and needs more efforts. For example, Luo and Hubaux [6] considered the problem in a circle where the sensors are uniformly distributed. They formulated the network lifetime maximization problem as a min–max problem, and derived a nice solution. That is, keeping the sink moving along the external perimeter of the circle will achieve a much longer network lifetime in comparison with the case where the sink only stays at the center of the circle. To improve network lifetime further by deploying multiple mobile sinks, Gandham *et al.* [3] presented an integer linear programming (ILP) model to determine the locations of multiple mobile sinks. They aimed to minimize the energy consumption per sensor node and the total energy consumption within each round, assuming that the movement of multiple sinks is scheduled round by round and each sink at each round is assigned with equal amount of sojourn time. Within each round, they assume that the maximum energy consumption of each sensor node is proportional to a fixed fractional of its residual energy. Wang *et al.* [4] considered a joint optimization problem of determining the sink movement and its sojourn time at certain network locations (co-located with the sensor nodes) in a grid network so that the network lifetime is maximized. For this special network, they proposed an ILP solution by finding the sojourn time of the mobile sink at each node, assuming that half the work load (the number of messages generated and received) of each node flows along its horizontal and vertical links toward the current location of the mobile sink. Because the grid network is a special network, the load at each node toward the sink can be calculated easily. Thus, they are able to calculate the exact energy consumption of each node when the mobile sink is at each possible location. Note that in all of these mentioned approaches, they adopt *flow-based routing protocols* to route sensing data to the mobile sink. Although the flow-based routing approaches are theoretically attractive by solving mixed ILP [3–5,10,16], they may not be applicable to real sensor networks because of inherent difficulties in flow control at each sensor at each time instance. Furthermore, this approach is computationally infeasible with the growth of network size. By contrast, in this paper we will adopt the well-known *routing tree* structure for data gathering [6,7,17,18], as the tree structure is naturally suitable for distributive wireless sensor networks. For tree-based data gathering, Luo *et al.* [7] considered a two-stage joint optimization framework: First, the mobile sink visits all ‘anchor’ points (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 a profile for each anchor point. At the end of this stage, the sink calculates an ILP formula and drops those anchor points at which the sojourn time of the sink is below a given threshold, because it is not worthwhile to keep them in the sojourn tour as the sojourn time is not long enough to amortize the energy overhead on building trees. What follows is to solve the ILP to find the exact sojourn times at the chosen anchor points. Basagni *et al.* [5] considered a more realistic model by incorporating two realistic *bottleneck* constraints on the mobile sink: the maximum moving distance at each movement and the minimum sojourn time at each sojourn location. They first formulated the network lifetime maximization problem as a mixed integer linear program and then proposed a simple, distributed heuristic. Liang *et al.* [19] recently considered the network lifetime maximization problem for time-sensitive data gathering by incorporating the maximum travel distance constraint on a mobile sink, and a breadth-first search (BFS) tree was built at each chosen location for data gathering. Yun and Xia [20] considered the network lifetime maximization with the tolerant delay constraint. They derived the relationship between the network lifetime and the data delivery delay. However, they did not take into account the maximum travel distance on the mobile sink. Xu *et al.* [21] dealt with event collections in sensor networks with a mobile sink. They aimed to minimize the total travel distance of the mobile sink per tour to collect all events by exploring spatial and temporal data correlations. In summary, existing studies in the literature focused on the network lifetime optimization under various bottleneck constraints on the mobile sink. These include bounding the number of sojourn locations [7], the minimum sojourn time at each sojourn location, and the maximum moving distance at each movement [5]. However, none of the works has taken into account an important *additive* constraint on the mobile sink—the maximum travel distance. Incorporating this additive constraint into the problem formulation makes the problem much more realistic but poses a great challenge, too. In this paper, a novel heuristic, which exhibits low computational complexity and high scalability, will be proposed to cope with this joint optimization problem under both additive and bottleneck constraints on the mobile sink. The heuristic will deliver a near optimal solution through experimental simulations.

1.2. Contributions

The main contributions of this paper are as follows. We first formulate the network maximization problem for time-sensitive data gathering with multiple constraints on a mobile sink as a multiple-constrained optimization problem. Because of its NP-hardness, we then devise a novel heuristic to find an optimal sojourn tour and sojourn time scheduling for the mobile sink. We finally conduct extensive experiments by simulations to evaluate the performance of the proposed algorithms. We also investigate the impacts of different constraints on the network lifetime. The experimental results demonstrate that the performance of the proposed heuristic is very promising, and the solution obtained is fractional of the optimal one.

The rest of the paper is organized as follows. We first introduce the system model and the problem definition in Section 2. We then introduce a greedy algorithm for the constructions of load-balanced routing trees (LBTs) in Section 3. Thirdly, we propose a novel heuristic that not only finds a sojourn tour for the mobile sink but also determines the actual sojourn time of the sink at each sojourn location in Section 4. We finally evaluate the performance of the proposed algorithms through experimental simulations in Section 5, and we conclude in Section 6.

2. PRELIMINARIES

In this section, we first introduce the system model, which includes introducing terminologies and the energy cost model. We then define the problem precisely.

2.1. System model

We consider a wireless sensor network consisting of n stationary, homogeneous *sensor nodes* to monitor a region of interest. There is one mobile sink, which may not initially be in the monitoring region, for data collection. The location of each sensor is fixed and known *a priori*. Each sensor equipped with an omnidirectional antenna has an identical transmission range. We further assume that although the mobile sink has unlimited energy supplies in comparison with the energy supplies of sensors, its energy consumption on mechanical movement is *proportional to* its travel distance, as it is powered by petrol or electricity, which is not an infinite resource and must be refueled over time. We also assume that the mobile sink can sojourn at each sojourn location for a certain amount of time to collect sensing data through a routing tree rooted at the location. For the sake of convenience, we assume that the potential sojourn locations of the mobile sink are co-located with the sensors. Unless otherwise specified, in this paper we only count the transmission and reception energy consumptions of each sensor by ignoring its other energy consumptions including sensing and computation energy consumptions, as it is well known that the radio frequency (RF) transmission is the dominant energy consumption in wireless communications [22].

Given a sensor network $G = (V, E)$, where V is the set of $n = |V|$ stationary sensor nodes and E is the set of $m = |E|$ links. There is a link between two nodes if they are within the transmission range of each other. The network lifetime is defined as the time of the *first sensor failure* due to the expiration of its energy [23]. A mobile sink (e.g., a robot or a moving vehicle) usually is powered by petrol or electricity to support its mechanical movement. Its maximum travel distance L per tour, thus, is bounded by the volume of petrol or the capacity of power it can carry. On the other hand, the sensing data by sensors may be lost because of the sink movement from one location to another. It is expected that the distance of the sink at the next location should not be far from its current location to minimize the data loss by its movement. With the assumption that the potential locations of the mobile sink are identical to the locations of sensors, the sensor at the current location of the mobile sink may be assigned as the *temporary sink* to collect data from other sensors during this transition period of the sink to avoid the data loss. Let $\Delta\tau$ be the maximum duration of the sink moving from its current location to its next location in addition to the setting up time of a new routing tree rooted at that location. Then, the buffer size of the sensor serving as the temporary sink is at least $n \cdot ra \cdot \Delta\tau$ to ensure that there will be no sensing data lost, where ra is the data generation rate. Once the sink resumes to work at the next location, the temporary sink will forward all collected data to the sink directly. Notice that we assume that the energy consumption for the buffer data transfer is negligible, compared with the regular data gathering session. In other words, the maximum distance at each sink movement is bounded by a given value $R_{\max}$ to minimize data loss because of sink mobility. Although the residual energy among the sensors can be balanced through the frequent movement of the mobile sink, the overhead associated with the sink movement must be taken into account, too. It is required that the mobile sink stays at each sojourn location at least a certain amount of time $T_{\min}$ [5] to make such a movement profitable.

2.2. Problem definition

The *distance-constrained mobile sink problem* for *time-sensitive* data gathering in a wireless sensor network is defined as follows.

Given a sensor network $G(V, E)$ consisting of n stationary sensors and a mobile sink, the mobile sink starts from its depot site v_0 and eventually returns to the depot to recharge its petrol or electricity for its next tour, where v_0 may be outside of the monitoring region. Assume that all potential sojourn locations of the mobile sink are exactly the locations of the n sensors and a routing tree rooted at each sink sojourn location will be used for data gathering. As we deal with time-sensitive data gathering, it requires that the sensing data from each sensor must be relayed to the sink within the minimum number of hops. Thus, the number of hops from each sensor to the root of the routing tree must be minimized. For the mobile sink, (i) its maximum travel distance per tour is bounded by L; (ii) its maximum moving distance at each movement is bounded by $R_{\max}$; and (iii) its minimum sojourn time at each sojourn location is at least $T_{\min}$ time units. The problem is to find a sojourn tour (also referred to as the sink tour) for the mobile sink such that the network lifetime is maximized, which is subject to the mentioned three constraints on the mobile sink.

In other words, let $v_1, v_2, \ldots, v_k$ be the sequence of visited sensor locations by the mobile sink, and let t_i be the sojourn time of the sink at the location v_i, $1 \leq i \leq k$. The problem is to find a sensor sequence and the sojourn time scheduling at each sensor location in the sequence such that the network lifetime $\sum_{i=1}^{k} t_i$ is maximized, provided that the following constraints on the mobile sink must be met: the maximum travel distance $\sum_{i=0}^{k-1} d(v_i, v_{i+1}) + d(v_k, v_0) \leq L$, the maximum moving distance $d(v_i, v_{i+1}) \leq R_{\max}$, and the minimum sojourn time $T_{\min}$, where $d(u, v)$ is the Euclidean distance between two locations of sensors u and v. For the sake of convenience, all symbols in the paper are listed in Table I.

3. LOAD-BALANCED ROUTING TREE

In this section, we assume that the mobile sink has chosen a specific location v as its current sojourn location. The problem then is to determine the sojourn time profile of the sink at v. That is, to determine the amount of energy consumed of each sensor per time unit when the sink located at v for data gathering. To maximize the sojourn time of the sink at v, we will construct a routing tree rooted at v, and the load among the children of the tree root must be balanced, because these children will be the energy bottlenecks of the entire network. We assume that no data aggregation will be performed at each relay node. As we deal with time-sensitive data gathering, it is required that each sensor can reach the tree root in the shortest

Table I. Notations.

Notation	Description
$G(V, E)$	The sensor network with sensor set V and link set E
n	Number of nodes in G, $n = \lvert V \rvert$
m	Number of links in G, $m = \lvert E \rvert$
e_t	Transmission energy consumption per bit in G
e_r	Reception energy consumption per bit in G
$nd_T(v)$	Number of descendants of sensor v in a routing tree T
L	End-to-end distance of the sink tour
$R_{\max}$	Maximum distance of mobile sink at each movement
$T_{\min}$	The minimum sojourn time of the sink at each sojourn location
l	Length of sensing data per sensor at each data gathering session
V_l	Set of sensors whose distances to the sink are l, $0 \leq l \leq h$
n_l	$n_l = \lvert V_l \rvert$, $0 \leq l \leq h$
$G_l(X_l, Y_l, E_l, w)$	A node-weighted bipartite graph, where $X_l = V_1$ and $Y_l = V_{l+1}$ and $w(x) \neq 0$ and $w(y) = 1$
$N_l(s, t, G_l)$	An induced flow network from G_l, $2 \leq l \leq h$
B	The capacity assigned to any link incident to t in N_l
B_{opt}	The optimal capacity value assigned to any link incident to t in N_l
f	The value of flow f that goes through each link in N_l
d	Number of children of the tree root
$G_{l,l+1}$	A bipartite graph that is used to assign nodes in V_{l+1} to nodes in V_l such that each node in V_l is load balanced
$d(v_i, v_j)$	Euclidean distance between sensors v_i and v_j with $1 \leq i$ and $j \leq n$
$RE(v_j)$	The residual energy of sensor v_j
$c_{v_i}(v_j)$	Number of descendants of v_j in the routing tree rooted at sensor v_i
v_0	Initial and final location of the mobile sink
t_i	The sojourn time of the sink at location of sensor v_i, $1 \leq i \leq n$
$t_{\max}$	$t_{\max} = \max_{1 \leq i \leq n}\{t_i\}$
M	$M = nt_{\max}$
ρ	$\rho = 1/nt_{\max}$
$P_{u,v}$	A sink tour starting from sensor u and ending at sensor v
$D(P_{u,v})$	$D(P_{u,v}) = \sum_{e \in P_{u,v}} d(e)$
$T(P_{u,v})$	The sum of sojourn times of sensor nodes in $P_{u,v}$
t'_i	The actual sojourn time of the sink at location of v_i

distance (in terms of number of hops). Thus, sensors in the routing tree can be partitioned into different layers by their distances to the tree root. Meanwhile, it is also desirable that the load (energy consumption) among the sensor nodes in every other layer besides the first layer is also load-balanced. The motivation behind is that, consider a scenario where the nodes in a specific layer are heavily imbalanced, if one of the nodes later is chosen as a sojourn location of the sink, it is likely that some of the nodes in the same layer will become the children of the chosen node, while the residual energy among these sensors is heavily imbalanced, the sojourn time of the sink at this new location will be significantly shortened. To maximize the sojourn time of the sink at any given location, we will construct an LBT. However, it has been shown that finding an optimal LBT is NP-complete by a reduction from the set cover problem and a heuristic to find such a tree is devised [24].

3.1. Heuristic algorithm for load-balanced spanning trees

In this subsection, we introduce a novel heuristic for finding a load-balanced spanning tree rooted at the sink by employing the network flow technique. We assume that the nodes in the network have been partitioned into h layers, according to their distance (the minimum number of hops) to the sink. Let V_i be the set of sensor nodes in layer i, then $\cup_{i=1}^{h} V_i = V$, and $V_i \cap V_j = \emptyset$ if $i \neq j$, where V_0 contains just the sink and V_1 contains all the sensor nodes that the sink is within their transmission ranges, $1 \leq i, j \leq h$. The tree is constructed layer by layer in a top-down fashion. Assume that a partial, load-balanced tree spanning the nodes from layer 0 to layer l has been constructed, we now expand the tree further by including the nodes in layer $l+1$ as follows.

Let $v_1, v_2, \ldots, v_{n_l}$ be the nodes in layer l where $n_l = |V_l|$ and $1 \leq l < h$. We first construct a node-weighted, bipartite graph $G_l = (X_l, Y_l, E_l, w)$, where Y_l is the set of nodes in layer $(l+1)$, that is, $Y_l = V_{l+1}$. $X_l \subseteq V_1$ is obtained as follows. The nodes in V_l are grouped by their ancestors in V_1, that is, the nodes in V_l are partitioned into at most d subsets, where d is the number of children of the tree root r. All the nodes in the same subset are the descendants of a child of the root. Let $x_1, x_2, \ldots, x_{d'}$ be the d' ($\leq d$) ancestors in V_1 of the nodes in V_l that are incident to nodes in Y_l. Associated with each $x_i \in X_l$, there is a weight $w(x_i)$, which is the number of descendants of x_i in the current tree. And each node $y \in Y_l$ is assigned a weight $w(y) = 1$. There is an edge $(x, y) \in E_l$ if $(v_i, y) \in E$ and $v_i \in V_l$ is a descendant of $x \in V_1$.

The load-balanced tree problem then is to choose a node $x_i \in X_l$ as its ancestor for each node $y_j \in Y_l$ such that the maximum number of descendants (the load) among the nodes in X_l in the resulting tree is minimized. We will transform this problem into a maximum flow and minimum cut problem. In the end, each sensor $y_j \in Y_l$ will choose a $x_i \in X_l$ as its ancestor, we refer to x_i as the ancestor of y_i in the resulting tree and denote by $anc(y_j) = x_i$.

Following the definition of the load-balanced spanning tree, we aim to balance not only the load among the children of the tree root but also the load among the nodes in other layers. Because the tree is expanded layer by layer, we assume that the load among the nodes in the first $l-1$ layers is balanced already, we proceed to balance the load among the nodes in layer l as follows.

A bipartite graph $G_{l,l+1} = (V'_l, V_{l+1}, E'_l)$ is constructed based on the results in G_l, where V'_l is the subset of V_l, which is the set of nodes whose ancestors are in X_l. There is an edge $(u, v) \in E'_l$ if $(u, v) \in E$, $u \in V'_l$, $v \in V_{l+1}$, and $anc(u) = anc(v)$. To balance the load among the nodes in layer l, we then find a node $u \in V'_l$ such that u is the parent of v in the expanded tree and the maximum load among the nodes in V'_l is minimized. This can be solved using the similar technique to balance the load among the children of the tree root.

In the following, we propose a solution for the tree expansion by transforming the load balancing tree problem among the children of the tree root and the nodes in layer l into the maximum flow and minimum cut problem in the corresponding auxiliary flow networks.

To balance the load among the nodes in the first layer (the children of the tree root), we transform the load balancing problem in $G_l(X_l, Y_l, E_l, w)$ into a maximum flow and minimum cut problem in an auxiliary flow network N_l through assigning its links with different capacities dynamically. $N_l = (X_l \cup Y_l \cup \{s, t\}, E_l \cup \{\langle s, x_i\rangle\} \cup \{\langle s, y\rangle \mid y \in Y_l\} \cup \{\langle x, t\rangle \mid x \in X_l\}), c)$, where s is the source node and t is the destination node. There is a directed edge from s to y with capacity 1 for each $y \in Y_l$, that is, $c(s, y) = 1$. For each x_i, there is a directed edge from s to x_i with capacity $c(s, x_i) = w(x_i)$. For each edge $(y, v_j) \in E$ in the original network G, there is a directed edge $\langle y, x_k\rangle$ from y to x_k with capacity 1, that is, $c(y, x_k) = 1$, assuming that v_j is a descendant of x_k in the constructed tree. There is a directed edge from each $x_i \in X_l$ to the destination node t with capacity $c(x_i, t) = B$, where B is the maximum load among the nodes in X_l and will be determined later. Thus, given a positive integer B, apply the maximum flow and minimum cut algorithm in N_l to find a flow f from s to t, and check whether $|f| = \sum_{1 \leq i \leq d} w(x_i) + |Y_l|$. It can be seen that $\max_{1 \leq i \leq d'}\{w(x_i) \mid x_i \in X_l\} \leq B \leq \max_{1 \leq i \leq d'}\{w(x_i) \mid x_i \in X_l\} + |Y_l|$. To determine the optimal value B_{opt} of B, we can use the binary search on the interval $[\max_{1 \leq i \leq d'}\{w(x_i) \mid x_i \in X_l\}, \max_{1 \leq i \leq d'}\{w(x_i) \mid x_i \in X_l\} + |Y_l|]$. As a result, the proposed maximum flow algorithm performs at most $\lceil \log |V_{l+1}| \rceil = \lceil \log n_{l+1} \rceil$ times to find the optimal load B_{opt} for the current tree expansion, where $n_{l+1} = |V_{l+1}|$. As a result, each node $y \in Y_l = V_{l+1}$ has been assigned an ancestor x_i if $f(y, x_i) = 1$, where f is the value of the flow and $(y, x_i) \in E_l$ and $anc(y) = x_i$, which means that x_i is an ancestor of y in the expanded tree.

What follows is the balancing of the load among the nodes in $V'_l = V_l - \{u \mid$ if there is not any edge $(v, u) \in E$

with $u \in V_l$ and $v \in V_{l+1}\}$ through the construction of another bipartite graph $G_{l,l+1}$; because none of the nodes in $V_l - V'_l$ will have a descendant in the future tree expansion, they therefore will not be considered any more. To achieve the load balance among the nodes in V'_l, we can transform the problem into the maximum flow and minimum cut problem in a flow network $N_{l,l+1} = (s, t, G_{l,l+1}, w)$ as follows.

Let $V_{l,i} = \{u \mid anc(u) = x_i,\ x_i \in X_l, u \in V_l\}$ be the set of nodes in V_l sharing the same ancestor $x_i \in V_1$. Then, $V'_l = \cup_{i=1}^{d'} V_{l,i}$. Clearly, set V_{l+1} can be partitioned into d' corresponding subsets $V_{l+1,i} = \{y \mid y \in V_{l+1}$ and $x_i = anc(y)\}$, that is, $V_{l+1} = \cup_{i=1}^{d'} V_{l+1,i}$, and there is not any edge in $G_{l,l+1}$ between nodes in $V_{l,i}$ and $V_{l+1,j}$ if $i \neq j$, $1 \leq i$, and $j \leq d'$. Then, the optimal value of the maximum load among the nodes in V'_l is within the interval between $\max_{1 \leq i \leq d'} \{\lceil |V_{l+1,i}|/|V_{l,i}| \rceil\}$ and $\max_{1 \leq i \leq d'} \{\lceil |V_{l+1,i}|/|V_{l,i}| \rceil + |V_{l+1,i}|\}$. The flow network $N_{l,l+1}$ is then defined, where s and t are the source and destination nodes; there is a directed edge from s to each $v \in V_{l+1}$ with capacity 1. There is a directed edge from s to each $u \in V'_l$ with capacity 1, and there is a directed edge from each $u \in V'_l$ to t with capacity B, while the value of B will be assigned dynamically, where the value range of B is within $[\max_{1 \leq i \leq d'} \{|V_{l+1,i}|/|V_{l,i}|\}, \max_{1 \leq i \leq d'} \{(|V_{l+1,i}|/|V_{l,i}|) + |V_{l+1,i}|\}]$. For each edge $\langle u, v \rangle \in E'_l$, its capacity is 1. If there is a flow f' in $N_{l,l+1}$ from s to t with $|f'| = |V'_l| + |V_{l+1}|$ under the optimal capacity assignment B'_{opt}, then for each edge $f(v, u) = 1$, u will be the parent of v in the expanded tree, where $u \in V'_l$ and $v \in V_{l+1}$. The detailed routine of tree expansion from layer l to layer $l + 1$ is described in Routine 1.

We, thus, have the following lemma.

Lemma 1. *Let T be a partial load-balanced tree including the nodes from layer 1 to layer l. Routine* `Top_Down_Load` *expands the tree by including the*

```
begin
  1.  $U' \leftarrow \{x \mid x\ is\ an\ ancestor\ \ in\ V_1\ of\ a\ node\ v \in V_l\ \&\ \exists\ y \in V_{l+1}\ s.t.\ (v, y) \in E\}$;
  2.  $B_{max} \leftarrow \max_{1 \leq i \leq d'} \{w(x_i) \mid x_i \in U'\}$;
  3.  $B_{new} \leftarrow B_{max}$; /* the balanced load among the nodes in $U'$ */
  4.  $B \leftarrow 0$; /* the initial load */
  5.  $low \leftarrow B_{max}$;
  6.  $top \leftarrow B_{max} + |V_{l+1}|$;
  7.  while ($B \neq B_{new}$) do
  8.      $B \leftarrow B_{new}$;
  9.      assign each link $\langle x_i, t \rangle$ in $N_l(s, t, G_l, c)$ from $x_i$ to $t$ with capacity $B$;
  10.     if  $\exists$ a flow $f$ in $N_l$ from $s$ to $t$ with $(|f| = \sum_{1 \leq i \leq d'} w(x_i) + |V_{l+1}|)$
  11.         then $top \leftarrow B$
  12.         else $low \leftarrow B$;
  13.     endif;
  14.     $B_{new} \leftarrow \lceil \frac{top+low}{2} \rceil$;
  15. endwhile;
  16. for each link $\langle y, x_k \rangle$ in the flow network $N_l$ do
  17.     if  $(f(y, x_k) = 1)$ then
  18.         $anc(y) \leftarrow x_k$;
  19.     endif;
  20. endfor;
  21. construct a bipartite graph $G_{l,l+1} = (V'_l, V_{l+1}, E'_l)$ and
          a corresponding flow network $N_{l,l+1}$;
  22. find a maximum flow $f'$ in $N_{l,l+1}$ from $s$ to $t$ such that
          $|f'| = |V'_l| + |V_{l+1}|$ and $B'_{opt}$ is an optimal one;
  23. for each edge $(u, v) \in E'_l$ with $u \in V'_l$ and $v \in V_{l+1}$ do
  24.     if  $f'(v, u) = 1$ then
  25.         $parent(v) \leftarrow u$;
              /* $u$ becomes the parent of $v$ in the expanded tree */
      endfor; end.
```

Routine 1. `Top_Down_Load`$(G, l, l + 1)$.

nodes in layer $l + 1$. Then, the maximum load among the children of the tree root in the resulting tree is the optimal in terms of the expansion to layer $l + 1$, $1 \le l < h$.

Proof. Let $U = \{u_1, u_2, \ldots, u_d\}$ be the set of d children of the root node and the weight $w(u_i)$ of each node $u_i \in U$ be the number of descendants of u_i in the partial load-balanced tree, that is, the tree spans all the nodes from layer 1 to layer l. Let $U' = \{u'_1, u'_2, \ldots, u'_{d'}\}$ be a subset of U. For a node $v \in V_l$, if there is at least a neighboring node $y \in V_{l+1}$ (i.e., $(v, y) \in E$), then the ancestor $u \in V_1$ of v is included by U'. Otherwise, the ancestor of v will not be in U'. It is obvious that the maximum load among the children of the root node in the partial load-balanced tree is $L_{\max}(l, U) = \max\{w(u) \mid u \in U\}$. Denote by $L_{\max}(l, U') = \max\{w(u') \mid u' \in U'\}$. Notice that the nodes in $U - U'$ will not increase their load in the future tree expansion, because they do not have any descendants in layer $l + 1$. Instead, only the nodes in U' will increase their load during the expansion. Thus, the maximum load among the nodes in U' in the resulting tree is within the interval $[L_{\max}(l, U'), L_{\max}(l, U') + |V_{l+1}|]$. Routine `Top_Down_Load` aims to find the optimal maximum load B_{opt} through assigning each outgoing link incident to each node in X_l in flow network N_l with a capacity that is the approximation of the optimal load, while the optimal load can be found through the binary search on its value interval. Because the value of the maximum flow f in N_l from s to t must be equal to $\sum_{x \in U'} w(x) + |V_{l+1}|$, this means that all the nodes in V_{l+1} must be included in the tree expansion. □

3.2. An example

We use here an example to illustrate the tree expansion. From Figure 1, it can be seen that the constructed tree is spanning the nodes in the first two layers ($l = 2$). Now, we expand the tree to include the nodes in the third layer. To do so, we construct an auxiliary bipartite graph $G_2 = (\{x_1, x_3\}, \{y_1, y_2, \ldots, y_7\}, E_2, w)$, where $w(x_1) = w(x_3) = 3$ while $w(y_j) = 1$ for all js with $1 \le j \le 7$. Notice that x_2 is not included in G_2 because it does not have any descendant in the third layer.

To construct a flow network $N_2 = (s, t, G_2, c)$, where s and t are the source and destination nodes, we assign each outgoing link incident to a node in U' a capacity equal to the maximum load B, where the value of B is assigned dynamically and its range is within $[\max\{w(x_1), w(x_3)\}, \max\{w(x_1), w(x_3)\} + |V_3|] = [3, 10]$.

The maximum flow found in N_2 corresponds to an ancestor assignment of the nodes in Y_2. That is, x_1 is the ancestor of nodes y_1, y_2, and y_3, whereas x_3 is the ancestor of nodes y_4, y_5, y_6, and y_7 in the expanded tree. Having this ancestor assignment of the nodes in $Y_l = V_{l+1}$ (see the graph in the right-hand side of Figure 2(a)), an induced subgraph $G_{2,3}$ by the nodes in V'_2 and V_3 is constructed, illustrated in Figure 2(a), and its corresponding flow network $N_{2,3}$ is shown in Figure 2(b). The maximum

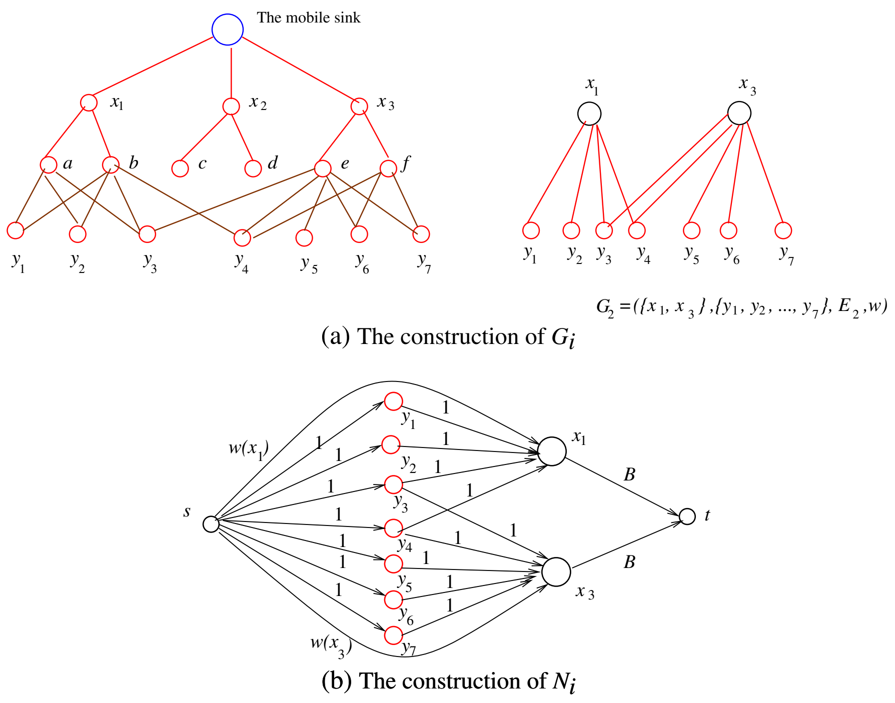
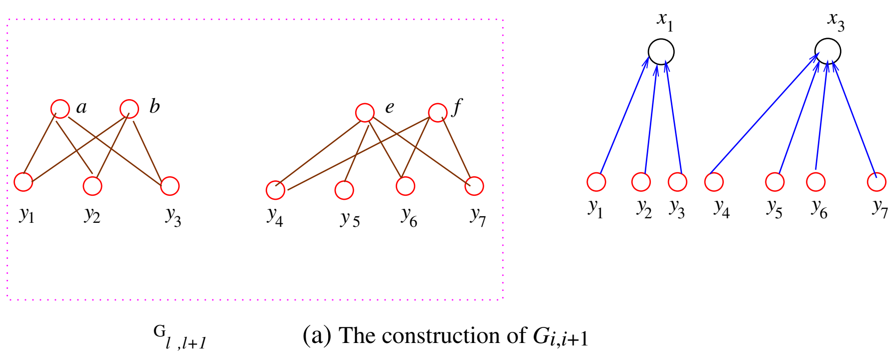
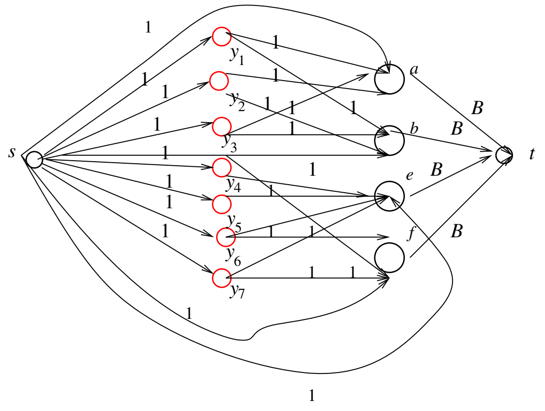

The mobile sink
x_1
x_2
x_3
a
b
c
d
e
f
y_1 y_2 y_3 y_4 y_5 y_6 y_7
$G_2 = (\{x_1, x_3\}, \{y_1, y_2, \ldots, y_7\}, E_2, w)$
(a) The construction of G_i
s
$w(x_1)$
$w(x_3)$
1
t
B
(b) The construction of N_i

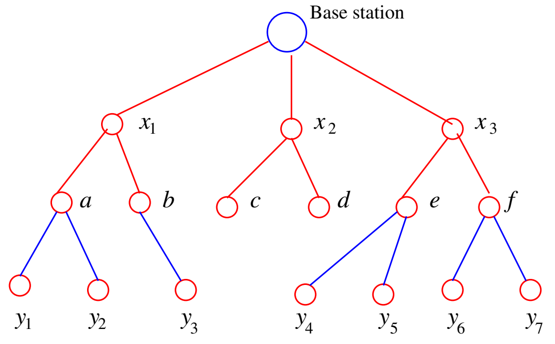
Figure 1. The constructions of G_l and N_l with $l = 2$.



(a) The construction of $G_{i,i+1}$



(b) The construction of $N_{i,i+1}$



(c) The resulting tree

Figure 2. The constructions of $G_{l,l+1}$ and $N_{l,l+1}$ with $l = 2$ and the expanded resulting tree including the nodes in layer 3.

flow and minimum cut problem in $N(2, 3)$ are then solved, and the optimal capacity B'_{opt} is obtained. Consequently, the resulting tree including the nodes in layer 3 is shown in Figure 2(c).

The load-balanced spanning tree rooted at location r can then be constructed layer by layer. We refer to this algorithm as algorithm `Balanced_Load_Tree`(G, r). The rest is to analyze the time complexity of the proposed algorithm by the following theorem.

Theorem 1. *Given a wireless sensor network $G(V, E)$ with the sink r at a specified location, there is a heuristic algorithm* `Balanced_Load_Tree` *for finding a load-balanced spanning tree rooted at r, which takes $O(mn \log n)$ time, where $n = |V|$ is the number of sensors and $m = |E|$ is the number of links in the network.*

Proof. The nodes in the network are partitioned into h disjoint subsets, and the node partitioning takes $O(m + n)$ time, using the BFS technique. The problem then is to construct a load-balanced spanning tree rooted at r. Assume that the tree constructed so far contains all the nodes from layer 0 to layer l. To expand the tree by including the nodes in layer $(l + 1)$, we construct a node-weighted corresponding bipartite graph $G_l(X_l, Y_l, E_l, w)$, which contains $|X_l| = d' \leq d = |V_1| = n_1$ children of the root node and $n_{l+1} = |V_{l+1}| = |Y_l|$ sensor nodes. We claim that $|E_l| \le |(V_l \times V_{l+1}) \cap E|$ by the following observation. Consider four links $(v_a, y_i) \in E$, $(v_b, y_j) \in E$, $(v_a, y_j) \in E$, and $(v_b, y_i) \in E$. If both v_a and v_b share the same ancestor x in V_1, then there are only two edges $(x, y_i) \in E_l$ and $(x, y_j) \in E_l$ in G_l by the construction of G_l. Let $m_l = |E_l|$. Then, the construction of G_l takes $O(n_1 + n_{l+1} + m_l)$ time because the size of G_l is $O(n_{l+1} + d' + m_l) = O(n_1 + n_{l+1} + m_l)$.

The corresponding flow network $N_l(s, t, G_l, c)$ of G_l contains $O(n_1 + n_{l+1})$ nodes and $O(m_l + n_1 + n_{l+1})$ links. To find an optimal balanced load B_{opt} for this layer expansion, we will perform the maximum flow algorithm in N_l at most $\lceil \log |V_{l+1}| \rceil$ times, while this algorithm takes $O(n'm')$ time in a graph with n' nodes and m' links [25]. Thus, each call of the maximum flow algorithm in N_l takes $O((n_{l+1} + n_1)m_l)$ time. The constructions of $G_{l,l+1}$ and $N_{l,l+1}$ can be performed in $O(|V_l| + |V_{l+1}| + |E_l|) = O(n_l + n_{l+1} + m_l)$ time, whereas the semi-matching computation in $G_{l,l+1}$ can be performed in $O((n_l + n_{l+1})|E'_l| \log n_{l+1})$ time, by applying the maximum flow and minimum cut algorithm in $N_{l,l+1}$ for $\lceil \log |V_{l+1}| \rceil$ times, while $|E'_l| \le |(V_l \times V_{l+1}) \cap E| \le m_l$ and $\sum_{l=1}^{h-1} m_l = m$. Following algorithm `Balanced_Load_Tree`, the construction of load-balanced spanning tree takes $\sum_{l=1}^{h-1} O((n_{l+1} + n_1)m_l \log n_{l+1}) = \sum_{l=1}^{h-1} O((n_1 + n_{l+1})m_l \log n) = \sum_{l=1}^{h-1} O(n_1 m_l \log n) + \sum_{l=1}^{h-1} O(n_{l+1}) * \sum_{l=1}^{h-1} O(m_l \log n) = O(mn \log n)$ time, because $n_{l+1} \le n$, $\sum_{l=1}^{h} n_l = n$, and $\sum_{l=1}^{h-1} m_l = \sum_{l=1}^{h-1} |E_l| \le \sum_{1=l}^{h-1} |(V_l \times V_{l+1}) \cap E| = |\cup_{l=1}^{h-1} [(V_l \times V_{l+1}) \cap E]| \le |E| = m$. □

4. HEURISTIC ALGORITHMS FOR FINDING SOJOURN TOURS FOR THE MOBILE SINK

As the distance-constrained mobile sink problem is NP-hard, in this section we focus on developing heuristic algorithms by proposing a basic algorithm first, followed by presenting an improved algorithm based on the basic algorithm.

4.1. Overview of the basic algorithm

The idea behind the proposed algorithm is to fully utilize the load-balanced spanning tree to prolong the sojourn time of the sink at each location, which then serves as the sojourn time profile of the sink at that location. It finally finds a sojourn tour and actual time schedule at each chosen location for the sink such that the network lifetime is maximized, provided that all specified constraints on the mobile sink are met. Specifically, the proposed algorithm consists of the following three stages.

It first builds a LBT rooted at each sensor location and determines the sojourn time profile of the sink at that location, assuming that the sink will sojourn at the locations of all sensors. It then finds a sojourn tour (also referred to as the sink tour or a visiting sensor sequence) based on the sojourn time profile at each sensor location, provided that maximum travel distance per tour is no greater than L, the maximum moving distance at each movement is no more than $R_{\max}$, and the minimum sojourn time at each sojourn location is no less than $T_{\min}$. It finally determines the exact sojourn time at each location in the sojourn tour.

4.2. The sojourn time of the sink at the location of each sensor

To calculate the duration of the sink staying at the location of a sensor, we construct a load-balanced spanning tree rooted at that location. Let T_v be the load-balanced tree rooted at the location of sensor $v \in V$ and $c_v(u)$ be the number of descendants of sensor $u \in V$ in T_v. Notice that a node is a descendant of itself. Thus, the amount of energy consumed at u is $ec_v(u) = l * c_v(u)$ in T_v if each sensor sends l-unit-length data to the sink using T_v. Let t_i be the sojourn time of the sink at the location of sensor v_i and $RE(v_i)$ the residual energy of sensor v_i, $1 \le i \le n$. The network lifetime maximization problem without distance constraint on the mobile sink is to

$$\text{maximize} \quad \sum_{i=1}^{n} t_i \tag{1}$$

which is subject to

$$\sum_{i=1}^{n} ec_{v_i}(v_j) \cdot t_i \le RE(v_j), \qquad \text{for all } j, 1 \le j \le n \tag{2}$$

$$t_i \ge 0, \qquad 1 \le i \le n \tag{3}$$

Notice that the above linear programming is polynomially solvable. The amount time t_i at location v_i is a lower bound of the actual sojourn time of the sink at the location if it is chosen as a sojourn location because all sensor locations are counted in the calculation of sojourn time profile.

Inequality (2) implies that the total amount of energy consumed at each sensor v_j is no more than the amount of energy it has. Assume that IE is the initial energy capacity of the sensor and $RE(v_j) = IE$ initially.

4.3. Finding a sojourn tour for the mobile sink

To find a sojourn tour for the mobile sink, we will reduce the problem to a distance-constrained shortest path problem as follows.

A weighted, directed graph $G_D = (V_D, E_D, \omega, d)$ is constructed, where $V_D = \{v_{i,1}, v_{i,2} \mid v_i \in V\}$. Assuming that the sink initially is located at v_0. The sink will start from and return to v_0 when it finishes the tour. Associated with each sensor $v_i \in V$, there are two corresponding nodes $v_{i,1}$ and $v_{i,2}$ in G_D, and there is a directed edge in E_D from $v_{i,1}$ to $v_{i,2}$ with weight $\omega(v_{i,1}, v_{i,2}) = t_i$, which is the sink sojourn time profile at location v_i and $d(v_{i,1}, v_{i,2}) = 0$. There is a directed edge $\langle v_{i,2}, v_{j,1}\rangle$ in E_D from $v_{i,2}$ to $v_{j,1}$ if the distance from v_i to v_j is no more than $R_{\max}$, that is, $d(v_{i,2}, v_{j,1}) \leq R_{\max}$, and the sojourn time at v_j is at least $T_{\min}$, that is, $t_j \geq T_{\min}$, the associated weight is $\omega(v_{i,2}, v_{j,1}) = 0$, and $d(v_{i,2}, v_{j,1})$ $(= d(v_i, v_j))$ is the Euclidean distance between v_i and v_j. G_D has the following important properties. For each node $v_{i,1}$, there are multiple incoming edges but only one outgoing edge $\langle v_{i,1}, v_{i,2}\rangle$. For each node $v_{i,2}$, there is only one incoming edge but multiple outgoing edges. The distance (length) of the two endpoints of each edge is no more than $R_{\max}$.

To find an optimal sojourn tour in $G(V, E)$ for the mobile sink starting from and returning to v_0 is reduced to find a distance-constrained longest path in G_D from a source node $v_{i,1}$ to a destination node $v_{j,2}$ such that the weighted sum (in terms of function ω) of all edges in the path is maximized, while the distance sum (in terms of function d) of all edges in the path is bounded by $L_{v_i,v_j} = L - d(v_i, v_0) - d(v_0, v_j)$, where $L(v_i, v_j)$ is the distance sum of all edges in the segment of the sojourn tour from v_i to v_j. It is well known that the distance-constrained longest path problem is NP-hard, because the well-known Hamiltonian path problem is one of its special cases where no constraints are imposed on its edges. In the following, we instead propose a heuristic for it.

We first reduce the distance-constrained longest path problem in G_D to the distance-constrained shortest path problem in another auxiliary graph $G'_D = (V_D, E_D, \omega', d)$. The latter in turn will return a feasible solution to the former. We then perform local improvement on the feasible solution by including as many qualified locations (sensors) as possible, provided that the specified constraints on the mobile sink are still met.

The definition of G'_D is as follows. For each directed edge $\langle v_{i,1}, v_{i,2}\rangle \in E_D$,

$$\omega'(v_{i,1}, v_{i,2}) = \begin{cases} M & \text{if } t_i = 0, \\ \frac{1}{t_i} - \rho & \text{otherwise.} \end{cases}$$

For each $\langle v_{i,2}, v_{j,1}\rangle \in E_D$, $\omega'(v_{i,2}, v_{j,1}) = 0$, where M and ρ are positive constants, $t_{\max} = \max_{1\leq i\leq n}\{t_i\}$. The value of M is no less than $t_{\max}$, and the value of ρ is no greater than $1/t_{\max}$. For example, $M = nt_{\max}$ and $\rho = 1/nt_{\max}$. The purpose of introducing the term ρ is to break a tie between two equal-length shortest paths between a pair of nodes by favoring the longer sojourn time 1. The rationale behind is illustrated by the following example.

Assume that the assignment of edge weights does not contain the term ρ. Consider a case where there are two distance-constrained shortest paths between a pair of nodes with equal length: one consists of two sensors a and b with sojourn times of $t_a = 2$ and $t_b = 2$, respectively, whereas another consists of four sensors c, d, e, and f with sojourn times of $t_c = t_d = t_e = t_f = 4$, then, the length of both paths is 1. Thus, the proposed algorithm can choose either one. However, it can be seen that the path consisting of sensors c, d, e, and f delivers a longer network lifetime. Now, we incorporate the term ρ into the assignment of edge weights. Then, the length of the path consisting of sensors a and b is $(1/t_a) + (1/t_b) - 2\rho = 1 - 2\rho$, and the length of the path consisting of sensors c, d, e, and f is $(1/t_c) + (1/t_d) + (1/e_t) + (1/t_f) - 4\rho = 1 - 4\rho$. Thus, the proposed algorithm will choose the shorter one consisting of sensors c, d, e, and f, not the one consisting of sensors a and b.

The distance-constrained shortest path problem in G'_D is to find a path from a source node $v_{i,1}$ to a destination node $v_{j,2}$ such that the weighted sum of the edges in the path is minimized, while the distance constraint in the path $L_{v_i,v_j} = L - d(v_0, v_i) - d(v_0, v_j)$ must be met. For convenience, in the rest of this paper, we say that a path P_{v_i,v_j} in G'_D actually means path $P_{v_{i,1},v_{j,2}}$ starting from node $v_{i,1}$ and ending at node $v_{j,2}$. There are several approximation algorithms for distance-constrained shortest path problem, and the state-of-the-art one is due to Chen *et al.* [26], which provides an optimal solution by relaxing the distance constraint to $(1 + \epsilon)L_{v_i,v_j}$ where ϵ is constant with $0 \leq \epsilon < 1$.

To find a sojourn tour for the mobile sink, it finds a distance-constrained shortest path in G'_D such that the sum of the sojourn time of the sink at each location in the path is maximized. Let $P_{u,v}$ be a distance-constrained shortest path in G'_D from node u to node v. Let $C(P_{u,v}) = \sum_{e\in P_{u,v}} \omega'(e)$ be the weighted sum of the edges in $P_{u,v}$ and $D(P_{u,v}) = \sum_{e\in P(u,v)} d(e)$ be the distance sum of the edges in $P_{u,v}$. Then, $D(P_{u,v}) \leq L - d(v_0, u) - d(v_0, v)$. The sum of sojourn times of the sink at the nodes in $P_{u,v}$ is $T(P_{u,v}) = \sum_{\langle v_{i,1}, vi,2\rangle\in P_{u,v}}\{1/[\omega'(v_{i,1}, v_{i,2}) + \rho] \mid \text{if } \omega'(v_{i,1}, v_{i,2}) \neq M\} + \sum_{\langle v_{i,2}, v_{j,1}\rangle\in P_{u,v}} \omega'(v_{i,2}, v_{j,1}) = \sum_{\langle v_{i,1}, v_{i,2}\rangle\in P_{u,v}} t_i$. The original problem is to find a sojourn tour for the mobile sink $P_{v_{i_0},v_{j_0}}$ such that $T\left(P_{v_{i_0},v_{j_0}}\right) = \max_{v_i\in V_D,\, v_j\in V_D}\left\{T\left(P_{v_i,v_j}\right)\right\}$.

Assume that a feasible sink tour $P = \langle v_0, v_1, \ldots, v_k, v_0\rangle$ has been found. Let $D(P) = \sum_{i=0}^{k} d(v_i, v_{i+1})$; clearly, $D(P) \leq L$. We now perform local improvement on path P such that the resulting path P' (if existent) contains more sojourn locations while all specified constraints are met. What follows is to check whether there are a sensor $v_j \notin P$ and a sensor $v_i \in P$ with $i \neq 0$ such that $d(v_j, v_i) \leq R_{\max}$, $d(v_j, v_{i+1}) \leq R_{\max}$, $t_j \geq T_{\min}$, and $D(P) + d(v_j, v_i) + d(v_j, v_{i+1}) - d(v_i, v_{i+1}) \leq L$. If yes, an *improved path* $P' = \langle v_0, v_1, v_2, \ldots, v_i, v_j, v_{i+1}, v_{i+2}, \ldots, v_k, v_0\rangle$ is found. If there are multiple such candidate sensors $v_j \notin P$, the one with the maximum sojourn time t_j among the candidate sensors will be chosen. This local improvement continues until no further improvement is possible. It is obvious that the time spent for local improvement is no more than $O(n^2)$. The detailed algorithm for stage two is described in Routine 2.

4.4. Calculating the sojourn time at each location in the sink tour

Let $P_{v_{i_0},v_{j_0}}$ be a found path in G'_D with the maximum value of $T(P_{v_{i_0},v_{j_0}})$. Then, the sojourn tour of the mobile sink is determined, which is $P_{v_{i_0},v_{j_0}}$. For the sake of simplicity, let $v_1, v_2, \ldots, v_k$ be the sensor sequence of k sensors in $P_{v_{i_0},v_{j_0}}$. Then, the current solution consisting of t_i for all i with $1 \le i \le k$ is a feasible solution to the original problem. To find a better solution to the problem of concern, we calculate the actual sojourn time t'_i of the sink at each location v_i such that $\sum_{i=1}^{k} t'_i \ge \sum_{i=1}^{k} t_i$, where $t'_i = t_i + \Delta t_i$, and the calculation of Δt_i is as follows:

$$\text{maximize} \qquad \sum_{i=1}^{k} \Delta t_i, \tag{4}$$

which is subject to

$$\sum_{i=1}^{k} ec_{v_i}(v_j) \cdot \Delta t_i \le IE - \sum_{i=1}^{k} ev_{v_i}(v_j) \cdot t_i, \quad \text{for all } j, 1 \le j \le n. \tag{5}$$

$$\Delta t_i \ge 0, \qquad 1 \le i \le k. \tag{6}$$

```
begin
        /* Stage two: find a sojourn tour for the mobile sink */
    1.  construct a weighted, directed graph $G'_D(V_D, E_D, \omega', d)$;
    2.  for each pair of nodes $v_{i,1} \in V_D$ and $v_{j,2} \in V_D$ do
    3.      find a distance-constrained shortest path $P_{v_i,v_j}$ in $G'_D$ from $v_i$ to $v_j$
            with the end-to-end distance constraint $L - d(v_0, v_i) - d(v_0, v_j)$;
            by calling an approximation algorithm due to Chen et al. [7];
        endfor;
    5.  let $T(P_{v_{i_0},v_{j_0}}) = \max_{v_i \in V, v_j \in V}\{T(P_{v_i,v_j})\}$, where $T(P_{u,v}) = \sum_{\langle v_{i,1},v_{i,2}\rangle \in P_{u,v}} t_i$.
        then, path $P_{v_{i_0},v_{j_0}}$ is a feasible solution;
        /* let $v_1, v_2, \ldots, v_k$ be the sensors in $P_{v_{i_0},v_{j_0}}$ with $v_{i_0} = v_1$ and $v_{j_0} = v_k$ */
        /* Perform the local improvement on $P_{v_{i_0},v_{j_0}}$ iteratively to find a feasible path */
        /* such that the sum of sojourn times of chosen sensors is maximized. */
    6.  $further\_improvement \leftarrow true$;
    7.  $P \leftarrow P_{v_{i_0},v_{j_0}}$;
    8.  while ($further\_improvement$) do
    9.      $temp_t \leftarrow T_{min}$; /* choose a sensor $v_j \notin P$ with the maximum sojourn time */
    10.     $control \leftarrow true$;
    11.     while $control$ do
    12.         $control \leftarrow false$;
    13.         for each $v_j \notin P$ do
    14.             if ($v_j$ meets the conditions at sensor $v_i$) and ($t_j > temp_t$)
    15.                 $temp_t \leftarrow t_j$; $index \leftarrow j$; $control \leftarrow true$;
                    endif;
                endfor;
    16.         if $control$ then
    17.             $P = \langle v_1, v_2, \ldots, v_i, v_{index}, v_{i+1}, \ldots, v_k\rangle$; /* update path $P$*/
                endif;
            endwhile;
    18.     if (there is no such $v_j$ meeting the conditions) then
    19.         $further\_improvement \leftarrow false$;
            endif;
        endwhile;
end
```

Routine 2. `Find_Sink_Tour`$(G(R_{\max}, T_{\min}, L))$.

4.5. Algorithm

In summary, the proposed heuristic algorithm is described in Algorithm 1.

We refer to Algorithm 1 as algorithm CSPLI and have the following theorem.

Theorem 2. *Given a wireless sensor network $G(V,E)$ and a mobile sink with the maximum travel distance L, the maximum moving distance $R_{\max}$, and the minimum sojourn time $T_{\min}$, there is a heuristic algorithm CSPLI for the distance-constrained mobile sink problem, which takes $O(mn^2\log n+(n^2m+n^3\log n)(L/\epsilon))$ time, where n is the number of sensors, m is the number of edges in G, and ϵ is a constant with $0<\epsilon\le 1$.*

Proof. We first show that algorithm CSPLI delivers a feasible solution to the distance-constrained mobile sink data collection problem. Following the construction of auxiliary directed graph G'_D, let $P_{v_{i_0},v_{j_0}}=\langle v_1,v_2,\ldots,v_k\rangle$ be the sojourn tour for the mobile sink by the routine Find_Sink_Tour; clearly, the end-to-end distance of $P_{v_{i_0},v_{j_0}}$ is $D\left(P_{v_{i_0},v_{j_0}}\right)\le L-d\left(v_0,v_{i_0}\right)-d\left(v_0,v_{j_0}\right)$, and $d(v_i,v_{i+1})\le R_{\max}$ for all i, $1\le i\le k-1$. Meanwhile, the total sojourn time of the sink derived from path $P_{v_{i_0},v_{j_0}}$ is the maximum one in G'_D. The local improvement on the feasible solution obtained also meets the distance constraint. Thus, the sink tour obtained at stage 2 is a feasible solution of the problem. That is, the total travel distance of the sink tour is bounded by L, the distance between two neighboring nodes v_i and v_{i+1} is no more than $R_{\max}$, $t_i\ge T_{\min}$, and $t_{i+1}\ge T_{\min}$ if $i\ne 0$. Within stage 3, the sojourn time t'_i of the sink at location v_i is maintained no less than t_i, that is, $t'_i=t_i+\Delta t_i\ge t_i\ge T_{\min}$. Therefore, the solution is a feasible solution. It can also be seen that $\langle t'_1,t'_2,\ldots,t'_k\rangle$ is a feasible solution of the problem, because its extension $\langle t'_1,t'_2,\ldots,t'_k,0,0,\ldots,0\rangle$ is a solution to inequality (2).

We then analyze the time complexity of the proposed algorithm. Following algorithm CSPLI, step 2 takes $O(mn^2\log n)$ time by Theorem 1, whereas step 3 takes $O(n^2)$ time. Step 4 takes $O(n^3)$ time to solve a linear programming. The running time of step 5 is analyzed as follows. The construction of G'_D takes $O(n^2)$ time. It is required to find $O(n^2)$ pairs of distance-constrained shortest paths in G'_D, while finding each of the paths takes $O((m+n\log n)(L/\epsilon))$ time [26]. In total, it takes $O((n^2m+n^3\log n)(L/\epsilon))$ time. The local improvement on the found sink tour takes $O(n^2)$ time. Step 6 takes $O(n^3)$ time. Thus, the proposed algorithm takes $O(mn^2\log n+(n^2m+n^3\log n)(L/\epsilon))$ time. □

4.6. An improved algorithm

In this subsection, we propose an improved algorithm for the problem of concern by using the basic algorithm CSPLI as its subroutine. If the sojourn time profile t_v at location v is less than the minimum sojourn time threshold $T_{\min}$, then v will not be included in any sojourn tour by algorithm CSPLI, because the distance-constrained shortest path in G'_D is constructed based on the sojourn time profile of each potential sojourn location. However, if v is chosen as a sojourn location of the mobile sink, the actual sojourn time t'_v of the sink at v may be above $T_{\min}$, because the calculation of t_v is based on the assumption that the

```
begin
        /* Stage one: calculate the sojourn time at each sensor location */
   1.   for   each $v \in V$ do
   2.           construct a load balanced spanning tree $T_v$ rooted at $v$
                    by calling algorithm Balanced_Load_Tree;
   3.           compute the energy consumption $ec_v(u)$ of $u$ in $T_v$ for each $u \in V$;
        endfor;
   4.   compute the sojourn time $t_v$ of the sink located at each sensor $v \in V$
                by solving linear programming, based on inequalities (2) and (6);
        /* Stage two: find a sojourn tour for the mobile sink */
   5.   **call** routine Find_Sink_Tour($G$);
        /* Let $P_{v_{i_0},v_{j_0}}$ be the feasible solution delivered in Stage two */
        /* Let $v_1, v_2, \ldots, v_k$ be the sensors in $P_{v_{i_0},v_{j_0}}$ where $v_{i_0} = v_1$ and $v_{j_0} = v_k$ */
        /* Stage three: calculate the sojourn time of the sink at each location of $v_i$ */
   6.   determine the actual sojourn times $t'_i$ of the sink at location $v_i$ with $1 \le i \le k$
                by solving inequalities (5) and (6);
end
```

Algorithm 1. Constr_Short_Path_Local_Impro($G(R_{\max},T_{\min},L)$).

sink will sojourn at all the potential sojourn locations. In fact, the sink only sojourns at some of all potential locations because of its travel distance constraint. To find a better solution, we further refine the solution of the proposed heuristic by including the locations like v in the sojourn tour finding. The strategy adopted here is to reduce the minimum sojourn time threshold from $T_{\min}$ to $T'_{\min}$. A new candidate solution is then delivered based on this new threshold. The candidate solution will then be examined to see whether the actual sojourn time at each sojourn location is no less than $T_{\min}$. If so, it is a feasible solution; otherwise, the next candidate solution will be examined by doubling the current threshold. In the end, a feasible solution with the maximum network lifetime will be chosen as the solution to the problem. To this end, Routine 3 `Is_Candidate` will examine whether a candidate solution is a feasible solution by returning a 'true' or 'false' boolean value.

When the found path $P\left(T'_{\min}\right)$ is a feasible solution, the actual sojourn time at each location $v \in P\left(T'_{\min}\right)$ can be calculated by stage 3 of algorithm `CSPLI`. Let $\Delta T_{\min}$ be the new minimum sojourn time threshold initially. It takes $\lfloor \log(T_{\min}/\Delta T_{\min}) \rfloor$ calls of routine `Find_Sink_Tour` and stage 3 of algorithm `CSPLI` to find a candidate sojourn tours and the actual sojourn time at each location in the tours. One of the feasible tours with the maximum sum of sojourn times will be chosen as the solution to the distance-constrained mobile sink problem. The improved algorithm, referred to as algorithm `RCSPLI`, is described in Algorithm 2.

We, thus, have the following theorem.

Theorem 3. *Given a wireless sensor network $G(V, E)$ and a mobile sink with the maximum travel distance L, the maximum moving distance at each movement $R_{\max}$, and the minimum sojourn time $T_{\min}$, there is a heuristic algorithm `RCSPLI` for the distance-constrained mobile sink problem, which takes $O(mn^2 \log n + [(n^2 m + n^3 \log n)(L/\epsilon)] \cdot \log(T_{\min}/\Delta T_{\min}))$ time, where n is the number of sensors and m is the number of edges in G, ϵ is a constant with $0 < \epsilon \leq 1$, and $\Delta T_{\min}$ $(\leq T_{\min})$ is a positive constant.*

Proof. The proof is similar to the one for Theorem 2, omitted. □

5. PERFORMANCE EVALUATION

In this section, we evaluate the performance of proposed algorithms. We also investigate the impacts of constraint parameters $R_{\max}$, L, and $T_{\min}$ on the network lifetime through experimental simulations.

```
begin
    /* assume that Routine Find_Sink_Tour($G(R_{max}, T'_{min}, L)$) has been called, */
    /* let $P(T'_{min})$ be the sojourn tour and $T'$ the network lifetime delivered by the routine. */
    /* Let $t_v$ be the sojourn time profile of location $v$ in $P(T'_{min})$ */
    1.  $is\_solution \leftarrow$ '$true$'; /* test whether the solution is a feasible solution */
    2.  for each $v \in P(T'_{min})$ do
    3.      for $j \leftarrow 1$ to $n$ do
    4.          RE$(v_j) \leftarrow$ IE $- t_v \cdot ec_v(v_j)$;
            endfor;
        endfor;
    5.  for each $v \in P(T'_{min})$ do
    6.      if $t_v < T_{min}$ then /* increment the sojourn time at location $v$ to $T_{min}$ */
    7.          for $j \leftarrow 1$ to $n$ do
    8.              RE$(v_j) \leftarrow$ RE$(v_j) - (T_{min} - t_v) \cdot ec_v(v_j)$;
    9.              if RE$(v_j) < 0$ then
                        $is\_solution \leftarrow$ '$false$' /* the solution is infeasible */
                    else $t_v \leftarrow T_{min}$;
                    endif;
                endfor;
            endif;
        endfor;
    10. return ($is\_solution$);
end
```

Routine 3. `Is_Candidate`$\left(P\left(T'_{\min}\right), T'_{\min}\right)$.

```
begin
  1.  call Algorithm Constr_Short_Path_Local_Impro($G(R_{max}, T_{min}, L)$);
      /* Let $P$ be the delivered sojourn tour and $T$ the network lifetime */
  2.  $T \leftarrow \sum_{v \in P} t'_v$;
  3.  $T'_{min} \leftarrow \Delta T_{min}$;
      /* $\Delta T_{min} < T_{min}$ the new minimum sojourn time threshold */
  4.  for   $i \leftarrow 1$ to $\lfloor \log \frac{T_{min}}{\Delta T_{min}} \rfloor$ do;
  5.        call Routine Find_Sink_Tour($G(R_{max}, T'_{min}, L)$);
            /* let $P(T'_{min})$ be the tour delivered by routine Find_Sink_Tour */
  6.        call Routine Is_Candidate($P(T'_{min}), T'_{min}$));
  7.        if   $is\_solution$ then /* a feasible solution */
  8.             compute the actual sojourn time $t'_v$ for each $v \in P(T'_{min})$
                      by calling Stage 3 of Algorithm CSPLI;
  9.             $T' \leftarrow \sum_{v \in P(T'_{min})} t'_v$;
  10.            if   $(T < T')$ then /* a longer network lifetime is found */
                      $P \leftarrow P(T'_{min})$; $T \leftarrow T'$;
                 endif;
            endif;
  11.       $T'_{min} \leftarrow 2T'_{min}$;
      endfor;
end
```

Algorithm 2. Refined_Constr_Short_Path_Local_Impro(G).

5.1. Simulation environment

We consider a wireless sensor network consisting of homogeneous sensors with network size from 20 to 400 randomly distributed in a $200\,\text{m} \times 150\,\text{m}$ rectangle region. The sensor locations are known *a priori*. The transmission range R of each sensor is 25 m, and the initial energy capacity of each sensor IE is 50 J. The data generation rate is $ra = 4$ bits/s. We assume that location $(0, 0)$ is the center coordinate of the monitoring region. A mobile sink located at v_0 with (150 100) is initially located at the outside of the monitored region. Assume that all potential sojourn locations of the mobile sink are co-located with the sensors.

In our experiments, we vary the maximum travel distance of the mobile sink L from 350 to 525 m and the maximum distance at each movement $R_{\max}$ from 25 to 75 m. Notice that the travel distances of the sink from v_0 to its first sojourn location and from its last sojourn location to return to v_0 have not been constrained by $R_{\max}$. The minimum sojourn time $T_{\min}$ at each sojourn location varies from 600 and 4800 s. In all our experiments, we adopt the actual energy consumption parameters of a real sensor, MICA2 mote [27], where the amounts of energy by transmitting and receiving 1-bit data are $e_t = 14.4 \times 10^{-6}$ and $e_r = 5.76 \times 10^{-6}$ J/bit, respectively. The value in each figure is the mean of the results by applying each mentioned algorithm to five different network topologies of same size. It must be mentioned that the running time is obtained on a Pentium 4 3.2-GHz machine with 1-GB RAM.

To investigate the performance of algorithms CSPLI and RCSPLI, we propose one of their variants, referred to as a greedy heuristic, as a benchmark for the comparison purpose. This heuristic consists of three stages, too. The only difference between them lies in stage 2. Instead of constructing auxiliary graphs by reducing the problem to a distance-constrained shortest path problem, the greedy heuristic lists the locations of all sensors in decreasing order of sojourn time profiles of the sink at them. In other words, let $v_{i_1}, v_{i_2}, \ldots, v_{i_n}$ be the sorted sensor sequence by their sojourn time profile in increasing order, that is, $t_{i_j} \geq t_{i_{j+1}}$ for all j with $1 \leq j < n$. The greedy algorithm proceeds as follows. Assume that the sink tour is initially empty. The sink tour will be expanded greedily by adding one sensor at one time through checking the sensors in the sorted sequence. Specifically, let $P = \langle s_1, s_2, \ldots, s_l \rangle$ be the sink tour constructed so far, and assume that these sensors whose ranks are up to $j - 1$ have been examined. We now explore the next sensor v_{i_j} by checking whether the following constraints are met.

$$D\left(P_{v_0,s_l}\right) + d\left(s_l, v_{i_j}\right) + d\left(v_{i_j}, v_0\right) \leq L, d\left(s_l, v_{i_j}\right) \leq R_{\max}, \text{ and } t_{i_j} \geq T_{\min}.$$

If yes, sensor v_{i_j} will be appended to the end of P to be part of the sink tour, and the sink tour $P = \langle s_1, s_2, \ldots, s_l, s_{l+1} \rangle$ is updated with $s_{l+1} = v_{i_j}$. Otherwise, it explores the next sensor $v_{i_{j+1}}$, and so on. The sink tour P will be constructed after examining all sensors in the sequence. We refer to this simple greedy heuristic as algorithm `Sorting-based Sink_Tour`, or `SST` for short.

5.2. Performance evaluation of the proposed heuristic

We first evaluate the performance of a heuristic algorithm $\mathcal{A}$, based on the sojourn time profile. Let T_{opt} be the optimal network time and T_{upper} be an upper bound on T_{opt}, where $T_{\text{upper}} = \sum_{i=1}^{n} t_i$. Clearly, $T_{\text{opt}} \leq T_{\text{upper}}$. Let $T_{\mathcal{A}} = \sum_{j=1}^{k} t'_i$ be the network lifetime delivered by algorithm $\mathcal{A}$, where the sojourn tour of the sink contains k sensor locations. Then, the performance ratio $\zeta_{\mathcal{A}}$ of algorithm $\mathcal{A}$ to the optimal one is defined as follows:

$$\zeta_{\mathcal{A}} = \frac{T_{\mathcal{A}}}{T_{\text{opt}}} \geq \frac{T_{\mathcal{A}}}{T_{\text{upper}}} = \frac{\sum_{i=1}^{k} t'_i}{\sum_{i=1}^{n} t_i}, \tag{7}$$

where algorithm $\mathcal{A}$ can be either of algorithms `RCSPLI`, `CSPLI`, or `SST`.

Figure 3 plots the ratio curves of different algorithms, where ζ_{RCSPLI}, ζ_{CSPLI}, and ζ_{SST} represent the performance ratios (approximation ratios) of algorithms `RCSPLI`, `CSPLI`, and `SST`, respectively, from which it can be seen that the performance of algorithm `RCSPLI` is marginally better than that of algorithm `CSPLI`. The approximation ratio of the network lifetime delivered by both of them is no less than 40% to 70%, while the approximation ratio of algorithm `SST` is no less than 30% to 55%, which is the worst among them. With the increase of the network size, the approximation ratio of all three algorithms decreases. Note that the estimate to the optimal network lifetime in Equation (7) is very conservative. The actual optimal network lifetime will be much less than T_{upper}, because of the maximum travel distance constraint on the mobile sink.

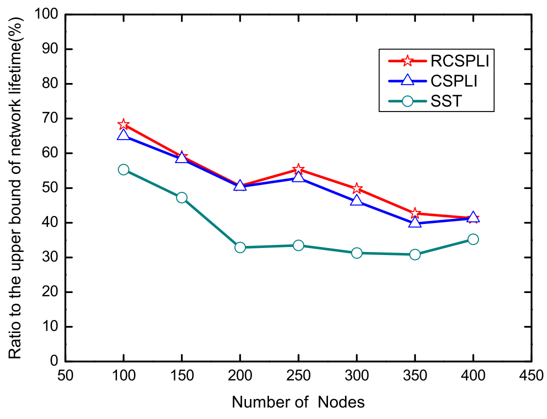


Figure 3. The performance ratios of various algorithms.

We then study the quality of the solutions delivered by algorithms `RCSPLI`, `CSPLI`, and `SST` against the upper bound T_{upper} on the optimal solution `MILP` by varying network sizes. Figure 4(a) plots the network lifetimes by algorithms `RCSPLI`, `CSPLI`, and `SST` when the network size varies from 20 to 35 under the constraints $R_{\max} = 35$ m, $L = 400$ m, and $T_{\min} = 600$ s. It can be seen that the performance of algorithms `RCSPLI` and `CSPLI` are nearly optimal when the network size is no greater than 35. Although algorithm `SST` is inferior to both algorithms `RCSPLI` and `CSPLI`, it still achieves around 94% of the approximation ratio of the optimal one. Overall, the average performance of `RCSPLI` and `CSPLI` are around 98.55% and 98% of the optimal one, respectively.

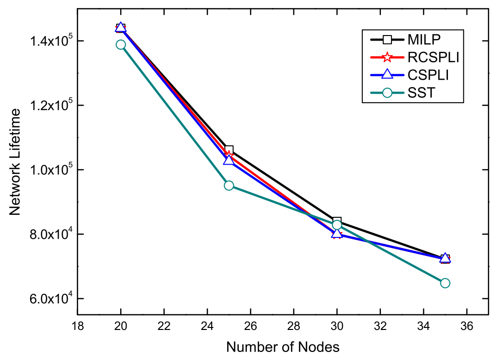
We finally evaluate the performance of heuristics `RCSPLI`, `CSPLI`, and `SST`, by varying the network size from 100 to 400, while keeping the other constraint parameters unchanged. It can be seen from Figure 4(b) that algorithms `RCSPLI` and `CSPLI` outperform algorithm `SST` in all cases. Specifically, the performance of algorithm `RCSPLI` is about 165% of that of algorithm `SST`, whereas the performance of algorithm `CSPLI` is around 158% of that of algorithm `SST` when $n = 250$ in the best case; otherwise, it is 117% of that of `SST` when $n = 100$.

5.3. Impacts of constraint parameters on the network lifetime

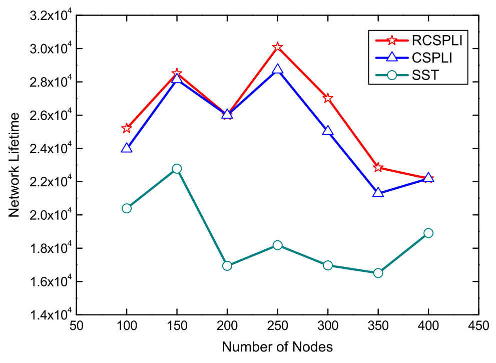
The rest is to analyze the impacts of constraint parameters $R_{\max}$, L, and $T_{\min}$ on the network lifetime. Notice that the value of $R_{\max}$ is closely related to the buffer size of sensors; the value of L in a certain degree will restrict the sink mobility. Intuitively, the larger the values of $R_{\max}$ and L are, the longer the network lifetime will be, because they give the mobile sink a high flexibility to choose its next sojourn location (from its current location). Similarly, the smaller the minimum sojourn time threshold $T_{\min}$ is, the longer the network lifetime will be, because it implies that the energy consumption among the sensors can be better balanced through the frequent movement of the mobile sink.

Figure 5(a) plots the network lifetimes delivered by different algorithms in a network consisting of 200 sensors, where $R_{\max}$ and $T_{\min}$ are fixed at 35 m and 600 s, respectively, while the value of L varies from 350 to 525 m. From this figure, it can be seen that the gap of network lifetime between algorithms `RCSPLI` (or `CSPLI`) and `SST` increases with the increase of the value of L. However, this improvement on the network lifetime is not unlimited, and any further improvement becomes insignificant when L reaches 500 m for algorithm `RCSPLI`, 475 m for algorithm `CSPLI`, and 450 m for algorithm `SST`.

Figure 5(b) shows the network lifetimes delivered by different algorithms in a network consisting of 200 sensors, where L and $T_{\min}$ are fixed at 400 m and 600 s, respectively, while $R_{\max}$ varies from 25 to 100 m. From this figure, it can be seen that although $R_{\max}$ heavily

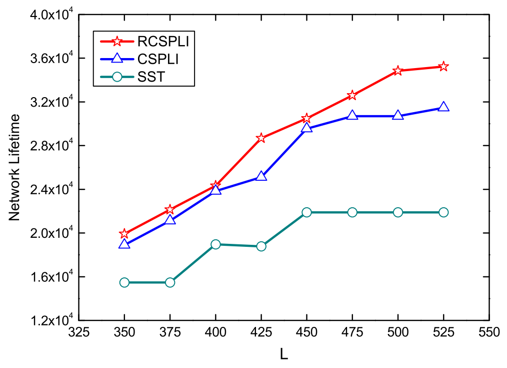


(a) The optimal solution vs feasible solutions.

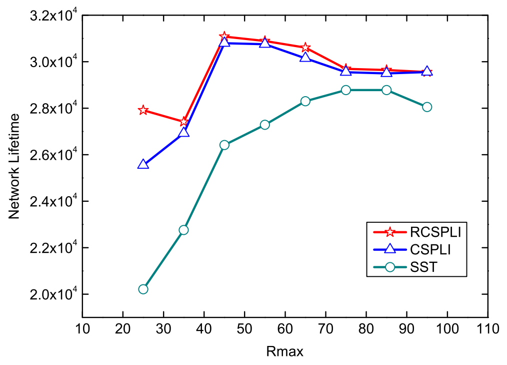


(b) Performance of different heuristics.

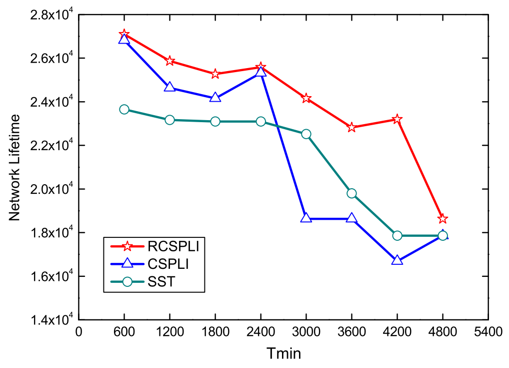
Figure 4. Network lifetime delivered by different algorithms.



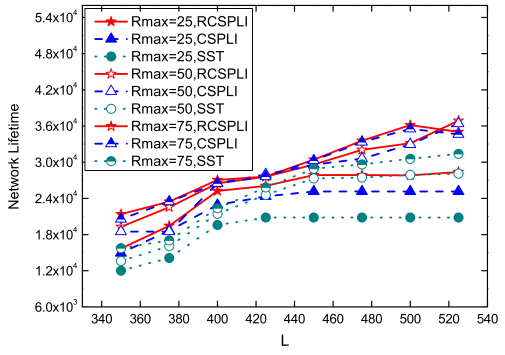
(a) Impact of L on network lifetime when $n = 200$, $R_{max} = 35m$, and $T_{min} = 600s$.



(b) Impact of R_{max} on network lifetime when $n = 200$, $L = 400m$, and $T_{min} = 600s$.



(c) Impact of T_{min} on network lifetime when $n = 200$, $L = 400m$, and $R_{max} = 35m$.



(b) Impact of R_{max} and L on network lifetime when $n = 200$ and $T_{min} = 600s$.

Figure 5. The impact of various constraint parameters on the network lifetime.

affects the network lifetime when it is no greater than 75 m for algorithms RCSPLI, CSPLI, and SST, this effect becomes diminishing with the further increase on the value of $R_{\max}$.

Figure 5(c) plots the network lifetime curves of 200-sensor networks when $R_{\max}$ and L are fixed at 35 and 400 m, while the value of $T_{\min}$ is ranged from 600 to 4800 s. Figure 5(c) implies, with the increase of the value of $T_{\min}$, that the network lifetime actually decreases. Once again, algorithm RCSPLI outperforms both algorithms CSPLI and SST. If the value of $T_{\min}$ is sufficiently large, it may lead to the sink tour consisting of one single sensor only, which means that if the overhead on the setting up of the next routing tree is excessive, keeping the mobile sink at the stationary rather than mobile status is a wise choice in the prolongation of network lifetime.

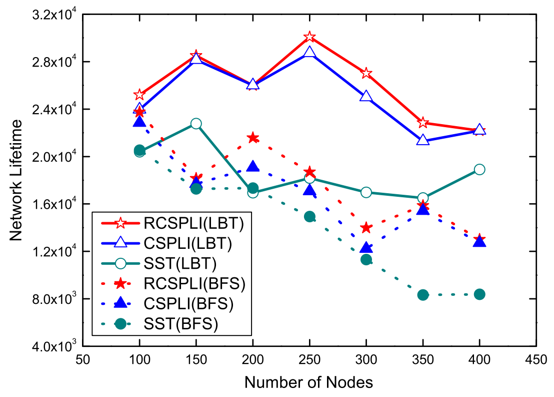


Figure 6. The performance ratios of various algorithms based on the load-balanced routing tree and the breadth-first search tree.

Figure 5(d) illustrates the impact of varying constraint parameters $R_{\max}$ and L on the network lifetime by algorithms RCSPLI, CSPLI, and SST when $T_{\min}$ =600 s is fixed. With the increase of L, the network lifetime becomes longer and longer, because the mobile sink can sojourn more locations. However, the performance of algorithms CSPLI and SST will not be improved when L reaches 450 and 400 m and $R_{\max}$ reaches 25 m. In contrast, by increasing $R_{\max}$ without changing L, the network lifetime may not necessarily increase as well. With the increase of both $R_{\max}$ and L, the network lifetime is further prolonged. It is noticed from Figure 5(d) that the impact of $R_{\max}$ on the network lifetime becomes diminishing, with the further increase of its value. For example, when $R_{\max} = 50$ m and $R_{\max} = 75$ m, any further increase of their values does not result in a much better network lifetime for neither algorithms RCSPLI and CSPLI nor algorithm SST.

5.4. Performance evaluation via existing solutions

In this subsection, we compare the performance of the proposed algorithm against the one in our previous work [19]. In paper [19], we studied the distance-constrained mobile sink problem and used a BFS tree instead of the LBT at each sojourn location for data gathering.

Figure 6 illustrates the network lifetime curves delivered by different algorithms, based on the LBT and the BFS tree. It clearly demonstrates that the algorithms based on LBT outperform the algorithms based on BFS trees. On average, the performance ratios by different algorithms based on LBT and BFS tree are 145.5% for RCSPLI, 149.8% for CSPLI, and 133.2% for SST.

6. CONCLUSION

In this paper, we have studied the network lifetime maximization problem for time-sensitive data gathering under a mobile sink environment, which is subject to the following constraints on the mobile sink: the maximum travel distance per tour, the maximum moving distance, and the minimum sojourn time at each sojourn location. We first formulated the problem as a multiple-constrained optimization problem. We then devised a novel three-stage heuristic. We finally conducted experiments by simulations to evaluate the performance of the proposed algorithms against existing ones as well as the optimal one. The experimental results demonstrate that the proposed heuristic is very promising, and the solution obtained is fractional of the optimal one.

REFERENCES

1. Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E. Wireless sensor networks: a survey. *Computer Networks* 2002; **38**: 393–422.
2. Akkaya K, Younis M, Bangad M. Sink repositioning for enhanced performance in wireless sensor networks. *Computer Networks* 2005; **49**: 512–534.
3. Gandham SR, Dawande M, Prakask R, Venkatesan S. Energy efficient schemes for wireless sensor networks with multiple mobile base stations. In *Proceedings of the Globecom'03*. IEEE: New Jersey, 2003; 377–381.
4. Wang ZM, Basagni S, Melachrinoudis E, Petrioli C. Exploiting sink mobility for maximizing sensor networks lifetime. In *Proceedings of the HICSS*. IEEE: New Jersey, 2005; 287a.
5. Basagni S, Carosi A, Melachrinoudis E, Petrioli C, Wang ZM. Controlled sink mobility for prolonging wireless sensor networks lifetime. *Wireless Networks* 2008; **14**: 831–858.
6. Luo J, Hubaux J-P. Joint mobility and routing for lifetime elongation in wireless sensor networks. In *Proceedings of the INFOCOM'05*. IEEE: New Jersey, 2005; 1735–1746.
7. Luo J, Panchard J, Piorkowski M, Grossblauser M, Hubaux J-P. MobiRoute: routing towards a mobile sink for improving lifetime in sensor networks. In *Proceedings of the DCOSS'06*, Vol. 4026. LNCS, Springer, 2006; 480–497.
8. Chatzigiannakis I, Kinalis A, Nikoletseas S. Sink mobility protocols for data collection in wireless sensor networks. In *Proceedings of the 4th International Symposium on Mobility Management and Wireless Access*. ACM: New York, 2006; 52–59.
9. Somasundara AA, Kansal A, Jea DD, Estrin D, Srivastava M B. Controllably mobile infrastructure for low energy embedded networks. *IEEE Transactions on Mobile Computing* 2006; **5**(8): 958–973.
10. Shi Y, Hou YT. Theoretical results on base station movement problem for sensor network. In *Proceedings*

of the IEEE INFOCOM'08. IEEE: New Jersey, 2008; 376–384.

11. Xing G, Wang T, Jia W, Li M. Rendezvous design algorithms for wireless sensor networks with a mobile base station. In *Proceedings of the MobiHoc'08*. ACM: New York, 2008; 231–240.
12. Sugihara R, Gupta RK. Optimizing energy-latency trade-off in sensor networks with controlled mobility. In *Proceedings of the INFOCOM'09*. IEEE: New Jersey, 2009; 2566–2570.
13. Wang Q, Wang X, Lin X. Mobility increases the connectivity of K-hop clustered wireless networks. In *Proceedings of the MobiCom'09*. ACM: New York, 2009; 121–131.
14. Younis M, Akkaya K. Strategies and techniques of node placement in wireless sensor networks: a survey. *Ad Hoc Network Journal* 2008; **6**: 621–655. Elsevier.
15. Basagni S, Carosi A, Melachrinoudis E, Petrioli C, Wang ZM. A new MILP formulation and distributed protocols for wireless sensor networks lifetime maximization. In *Proceedings of ICC'06*. IEEE: NJ, 2006; 3517–3524.
16. Hou YT, Shi Y, Sherali HD. Optimal base station selection for anycast routing in wireless sensor networks. *IEEE Transactions on Vehicular Technology* 2006; **55**(3): 813–821.
17. Madden S, Franklin MJ, Hellerstein JM, Hong W. TAG: a Tiny AGgregation service for ad-hoc sensor networks. In *Proceedings of the OSDI'02*. ACM: NY, 2002; 131–146.
18. Liang W, Liu Y. Online data gathering for maximizing network lifetime in sensor networks. *IEEE Transactions on Mobile Computing* 2007; **6**(1): 2–11.
19. Liang W, Luo J, Xu X. Prolonging network lifetime via a controlled mobile sink in wireless sensor networks. In *Proceedings of the Globecom'10*. IEEE: New Jersey, 2010; 1–6.
20. Yun Y, Xia Y. Maximizing the lifetime of wireless sensor networks with mobile sink in delay-tolerant applications. *IEEE Transactions on Mobile Computing* 2010; **9**: 1308–1318.
21. Xu X, Luo J, Zhang Q. Delay tolerant event collection in sensor networks with mobile sink. In *Proceedings of the INFOCOM'10*. IEEE: New Jersey, 2010; 1–9.
22. Pottie GJ. Wireless sensor networks. In *Proceedings of Information Theory Workshop*. IEEE: New Jersey, 1998; 139–140.
23. Chang J-H, Tassiulas L. Energy conserving routing in wireless ad hoc networks. In *Proceedings of the INFOCOM'00*. IEEE: New Jersey, 2000; 22–31.
24. Luo J, Liang W. Load-balanced routing trees for data gathering in wireless sensor networks. *Technical Report*, 2009.
25. Cormen TH, Leiserson CE, Rivest RL, Stein C. *Introduction to Algorithms*, 3rd Ed. MIT Press: MA, 2009.
26. Chen S, Song M, Sahni S. Two techniques for fast computation of constrained shortest paths. *IEEE/ACM Transactions on Networking* 2008; **16**: 105–114.
27. Crossbow Inc. MPR—Mote Processor Radio Board User's Manual, 2003.

AUTHORS' BIOGRAPHIES

Weifa Liang (M'99–SM'01) received his PhD degree from the Australian National University in 1998, his ME degree from the University of Science and Technology of China in 1989, and his BSc degree from Wuhan University, China, in 1984, all in Computer Science. He is currently an associate professor in the School of Computer Science at the Australian National University. His research interests include design and analysis of energy-efficient routing protocols for wireless *ad hoc* and sensor networks, information processing in wireless sensor networks, routing protocol design for WDM optical networks, design and analysis of parallel and distributed algorithms, query optimization, and graph theory. He is a senior member of the IEEE.

Jun Luo received his MS degree in Software Engineering from the National University of Defense Technology, China, in 1989, and his BS degree in Computer Science from Wuhan University, China, in 1984. He is currently a full professor at the School of Computer in the National University of Defense Technology, China. His research interests are in *ad hoc* and sensor networks and design of energy-efficient protocols for wireless networks and operating systems.

Xu Xu received her ME degree from the Institute of Computing Technology, Chinese Academy of Sciences in 2009, and his BS degree from China Agricultural University in 2006, both in Computer Science. She is currently doing her PhD in the Research School of Computer Science at the Australian National University. Her research interests include sink mobility in wireless sensor networks, routing protocol design for wireless *ad hoc* and sensor networks, data gathering in wireless sensor networks, and optimization problems.