Submodular Utility Maximization for Deadline Constrained Data Collection in Sensor Networks

Zizhan Zheng, *Member, IEEE* and Ness B. Shroff, *Fellow, IEEE*

***Abstract*—We study the utility maximization problem for data collection in a wireless sensor network subject to a deadline constraint, where the data on a selected subset of nodes are collected through a routing tree subject to the 1-hop interference model. Our problem is closely related to the traditional utility maximization problems in networking and communications. However, instead of a separable concave form of utility functions commonly seen in this area, we consider the class of monotone submodular utility functions defined on subsets of nodes, which is more appropriate for the applications we consider. While submodular maximization subject to a cardinality constraint has been well understood, our problem is more challenging due to the multi-hop data forwarding nature even under the simple interference model. We have derived efficient approximation solutions to this problem both for raw data collection and when in-network data aggregation is applied.**

I. Introduction

A wireless sensor network (WSN) consisting of resource constrained sensor nodes connected via wireless communication channel provides an efficient infrastructure for monitoring the physical environment as well as the human world on an unprecedented scale [7], through cooperation among networked sensors. Due to the limited sensing and processing capability of a single node, local information sensed by each node needs to be gathered at one or more processing centers, commonly referred to as the "sinks". Moreover, sensor nodes are low power devices with short communication ranges; hence, data collection in a WSN is often implemented in a "multi-hop" paradigm, where data are forwarded to a sink node through multiple transmissions between nearby sensor nodes.

Multi-hop communication is especially important in harsh environments when there is no wired infrastructure available. In the last decade, multi-hop sensor networks of small (~10 nodes) to medium (~100 nodes) size have been deployed for various environmental monitoring projects [17], [24], [32], [33], and very large deployments (~1000 nodes) have also been considered in field trials [3] and test-bed environments [1], [2]. It is projected that large-scale long-lasting sensor networks will play a key role in enabling unattended environmental monitoring at scale. An important challenge is that collecting data from all the nodes in a WSN may incur high communication overhead such as large delays or high energy consumption especially when the network becomes large. To improve the efficiency of data collection while ensuring the quality of data, a key observation is that the spatial and temporal correlations between the sensor nodes lead to redundancy inherent in their data. For instance, the observations made by two nearby sensor nodes towards the same target are likely to be correlated. Thus, in many cases, information from a subset of nodes may be sufficient from the applications' perspective.

In this paper, we study the problem of utility maximization for data collection in sensor networks subject to a deadline constraint. We consider a setting where each node in a sensor network holds some sensed data with respect to an event. To collect the data, a routing tree rooted at a sink has been built. Two data collection schemes are considered: (a) data forwarding, where raw data are forwarded towards the sink without manipulation, which is appropriate when complicated post-processing on sensed data is needed, and (b) in-network data aggregation, where data packets can be aggregated in the internal nodes, which could significantly reduce the communication overhead when only an aggregated form of the sensed data is needed [10]. In both cases, data packets are forwarded towards the sink in a multi-hop way subject to the 1-hop interference model, where no two links sharing a node can be activated at the same time. This model has been used to characterize the interference in Bluetooth and FH-CDMA based systems, and also applies to sensors with directional antennae [9]. The problem is to decide which nodes should transmit data so that the aggregated information, which is described by a utility function, received at the sink within a deadline, is maximized. Such a deadline constraint is especially important for applications that require real-time data, such as intruder detection and tracking.

Our problem is closely related to the traditional utility maximization problems in networking and communications. In the classic formulation of utility maximization problems for wireless networks [8], [26]–[28], there are a set of users (flows) in the network. Each user s is associated with a source node f_s and a destination node d_s, a real data rate x_s with which data is sent from f_s to d_s in a multi-hop way subject to some interference model, and a utility $U_s(x_s)$, where U_s is typically a non-decreasing and strictly concave function. The problem is to choose a vector of data rates to maximize the system utility, i.e., $\sum_s U_s(x_s)$, such that the system is stable, or even better, the average or the worst-case delay is

Z. Zheng is with the Department of Electrical and Computer Engineering, The Ohio State University, 2015 Neil Ave., Columbus, OH 43210, USA. zheng.497@osu.edu.

N. B. Shroff is with the Department of Electrical and Computer Engineering and the Department of Computer Science and Engineering, The Ohio State University, 2015 Neil Ave., Columbus, OH 43210, USA. shroff.11@osu.edu.

The work is partially supported by the Army Research Office MURI Awards W911NF-08-1-0238 and W911NF-07-1-0376, NSF grants CNS-0831919 and CNS-0721434.

A preliminary version of this paper entitled "Maximizing a Submodular Utility for Deadline Constrained Data Collection in Sensor Networks" appeared in the proceedings of the 10th Intl. Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), 2012 [36].

bounded. In our problem, each sensor node can be viewed as a user with itself as the source and the sink as the destination. The decision for each node is binary, i.e., whether to send data or not. Hence, our problem can be viewed as a utility maximization problem with binary decisions.

The key difference between our problem and the traditional utility maximization problems is with the choice of utility functions. In particular, when x_s is binary for any s, the commonly considered system utility $\sum_s U_s(x_s)$ reduces to a simple additive form, where each node is associated with a non-negative weight, and the total utility for a set of nodes is simply the sum of their weights. The weight of a node models the quality of information that the node can provide and can represent, for instance, error variance or distortion in the sensed data. *However, such an additive utility largely ignores the spatial correlation of sensed data.* Therefore, we propose to consider a more general class of utility functions $f : 2^V \rightarrow \mathbb{R}^+$, where f is defined over all the subsets of a set of nodes V, and $f(S)$ denotes the quality of data associated with the nodes in set $S \subseteq V$. For additive utility, $f(S) = \sum_{a \in S} w_a$, where w_a is the weight of node a. In this paper, we consider the general class of set functions that are *monotone submodular* (a formal definition is provided in Section III), which captures the spatial correlation and includes additive utility as a special case.

Submodularity, a discrete counterpart of concavity, captures a diminishing return property commonly seen in reality: the marginal utility improvement when adding a node to a small subset of nodes is at least as much as adding the node to a larger subset. Many interesting sensor selection criteria have been shown to satisfy submodularity, such as the total area covered or total number of targets detected by a set of nodes in a disk sensing model, the mutual information [23] and variance reduction criterion [22] for modeling the uncertainty of unsensed locations in an information based sensing model, and the maximum a posteriori (MAP) estimate and a variant of the maximum likelihood (ML) estimate for parameter estimation [19], [31].

Most previous works on sensor selection with submodular utility, however, consider a simple cardinality constraint. The problem is to select a subset of nodes S that maximizes a submodular utility $f(S)$ subject to the constraint that the number of nodes selected is bounded by a parameter k, that is $|S| \leq k$. Even in this case, the problem is, in general, strongly NP-hard. In contrast, our problem is to maximize $f(S)$ such that $L(S) \leq D$, where $L(S)$ denotes the minimum delay for collecting all the data in S and D is given parameter. Note that the deadline constraint reduces to the cardinality constraint in a single hop network where all the nodes are connected to the sink directly. In a general multiple-hop network, however, two subsets of same cardinality can differ greatly in terms of delay. Moreover, even for a given set S, $L(S)$ varies under different data collection schemes. Hence, our problem for submodular maximization subject to a deadline constraint is much more challenging even under the simple 1-hop interference model. For data aggregation with *additive* utility, this problem can be solved efficiently using dynamic programming [13], which, however, does not apply to data forwarding or general submodular utility.

In this paper, we propose a general utility maximization framework for efficient data collection in large-scale sensor networks subject to communication resource constraints. We study submodular utility maximization problem for both data forwarding and data aggregation and establish provable bounds for our solutions. In addition to deadline constraints, the techniques developed can be applied to other types of resource constraints, such as a per-node energy constraint. The main contributions of this paper are as follows.

- For deadline constrained data forwarding, we propose an efficient greedy solution that achieves at least a fraction of 1/3 of the optimal utility for a general submodular utility function, and a better fraction of 1/2 for additive utility. To prove this result, we have corrected a key formula in [9] that characterizes the minimum delay for collecting raw data from any subset of nodes in a tree network subject to 1-hop interference. We further identify some interesting cases where the greedy algorithm is optimal.
- For data aggregation, we propose a bi-criteria approximation, which, for a given deadline D, achieves at least a fraction of $\frac{1}{\min(h_T, D)+1}$ of the optimum utility, and has a delay of at most $\rho_T D$. For a routing tree T, h_T denotes its height, and ρ_T is upper bounded (loosely) by the maximum node degree. We expect that ρ_T is typically small (< 2) in practice, which is confirmed in simulations. This algorithm is further utilized to derive a feasible solution with guaranteed utility.
- We evaluate the performance of our algorithms using simulations for two application scenarios: target point coverage and parameter estimation. For small networks where optimal solutions can be found through exhaustive search, we show that our algorithms perform very close to the optimal in both scenarios. We further consider large networks and observe significant performance improvement when comparing our algorithms with two heuristics that ignore the spatial correlation among nodes.

The current paper improves our previous conference paper [36] in various ways. First, for data forwarding, we have provided new analytic results for the greedy algorithm (Algorithm 1). In particular, we have shown that the algorithm is optimal when the utility function is additive and node weights satisfy a monotonicity condition (see Section IV). Second, for data aggregation, we have proposed a new searching-based algorithm (Algorithm 3). Our original algorithm (Algorithm 2) may output an infeasible solution, albeit the infeasibility is bounded; hence, it may not be directly applicable in practice. In contrast, the new algorithm always generates a feasible solution while still providing a performance guarantee (see Section V). Third, we have considered the new application scenario of parameter estimation in simulations. Moreover, in addition to the random sensor selection heuristic, we have also compared our algorithms with the optimal solutions in small deployments, and a heuristic based on the dynamic programming algorithm in [13] for additive utility (see Section VI).

The rest of the paper is organized as follows. We review the related works in Section II, and present the system model

and the problem definition in Section III. Our solutions to the deadline constrained data forwarding and data aggregation problems are presented in Sections IV and V, respectively. Simulation results are presented in Section VI. We conclude the paper in Section VII.

II. Related Work

Delay constrained data collection in sensor networks has been studied to some extent. Most pervious works consider the problem of finding a minimum-delay schedule for collecting data over a network. For data forwarding, a minimum-delay schedule is identified in [9] for a given routing tree under the 1-hop interference model, which is further extended to the omni-directional antenna model in [18]. For data aggregation, a minimum-delay schedule for a routing tree under 1-hop interference model can be easily determined as we remark at the end of Section V-C. On the other hand, the problem is known to be NP-hard for a general graph even under the 1-hop interference model [30] and for a unit disk graph under the protocol interference model [6]. Approximation algorithms have been proposed in [6], [30], [35].

In contrast, we consider the dual problem of utility maximization under a deadline constraint, which has received relatively less attention until recently. To the best of our knowledge, this problem has only been considered for data aggregation with additive utility. In particular, for a routing tree subject to the 1-hop interference model, an optimal dynamic programming based solution is provided in [13]. The algorithm is further extended to consider unreliable links [14], per-node energy constraints [15], and joint deadline and energy constraints [16]. These techniques, however, do not apply to data forwarding or a general submodular utility. Our work provides efficient algorithms for general submodular utility functions and both types of data collection schemes.

Submodular functions have been used in modeling utility in various network design problems, with applications in sensor networks [22], [23], [25] and social networks [20], [25]. For instance, the influence of a given set of seeds in a social network can be modeled by a submodular function [20]. Most works on submodular utility maximization have a simple cardinality constraint, which allows a $(1-1/e)$ approximation ratio by a simple greedy technique when the utility is also nondecreasing [11]. The same bound can be achieved for a general matroid constraint [4]. The problem becomes more challenging when a connectivity requirement is added [12], [21], i.e., a feasible solution is a connected subgraph of certain type. For instance, in the group Steiner problem [12], the objective is to maximize a particular submodular function defined on the subsets of nodes in an edge-weighted graph, subject to a constraint on the weight of a Steiner tree that spans the set of nodes selected. This problem only allows a logarithmic factor approximation even when the underlying graph is a tree [12]. Our problem is related to this problem with two key differences. First, we are looking at a general monotone submodular utility function. Second, the delay cost with/without data aggregation is very different from the cost of a weighted Steiner tree.

TABLE I
Notation List

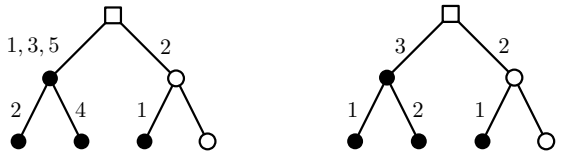
Symbol	Meaning
n	Number of sensor nodes
s_0	Sink node
T	Data collection tree (rooted at s_0)
V_T	Set of sensor nodes in T
E_T	Set of links in T
h_T	Height of T
Δ_T	Maximum number of children of nodes in T
$T(S)$	A subtree of T of minimum size that spans nodes $S \cup \{s_0\}$
$L_F(S)$	Minimum number of time slots required for forwarding data in nodes $S \subseteq V_T$ to s_0 subject to 1-hop interference
$L_A(S)$	Minimum number of time slots required for aggregating data in nodes $S \subseteq V_T$ to s_0 subject to 1-hop interference
$L_A(T)$	$\triangleq L_A(V_T)$
f	A monotone submodular utility function defined on subsets of V_T
D	Deadline constraint

III. System Model and Problem Formulation

Consider a sensor network of n sensor nodes and a sink s_0. We assume that a routing tree rooted at s_0 that spans all the nodes has been built for data collection. We let $T = (V_T, E_T)$ denote this tree, where V_T is the set of nodes and E_T is the set of links. Let h_T denote the height of T, and Δ_T the maximum number of children of any node in T. Let $|T|$ be the size of T, i.e., the number of links in T. We say that a node is at level k if the path in T that connects the node to the sink has k links. The level of s_0 is 0. Table I summarizes the notations used in the paper.

We assume that all the nodes are sensing a single event, and each node has *at most* one data packet ready to be delivered, which contains the information sensed by the node in the last period of time. The set of nodes with packets can be either learned through direct communication or inferred from the geographic span of the event, and is assumed to be known. Two data collection patterns are considered: *data forwarding* and *data aggregation*. For data forwarding, raw data packets are forwarded along the routing tree to the sink without manipulation within the network. All the packets are assumed to be of the same size. In the case of data aggregation, each internal node applies an aggregation function, e.g., MAX, MIN, SUM, etc., to the data received from its children and the local data to generate a new packet of the same size, which is then forwarded to its parent.

We consider a time-slotted and synchronized system. In each time slot, a node either forwards one packet to its parent or receives a packet from one of its children, or remains idle, subject to the 1-hop interference constraint. That is, when a node is transmitting, it cannot receive packets from its children, and two children cannot transmit at the same time. Links are assumed to be reliable. For a subset $S \subseteq V_T$, let $L_F(S)$ (resp. $L_A(S)$) denote the minimum number of time slots needed for forwarding (resp. aggregating) the packets at nodes S to the sink s_0. Fig. 1 shows an example of minimum delay schedules for data forwarding and data aggregation,



(a) data forwarding (b) data aggregation

Fig. 1. Minimum delay schedules for a tree with 6 internal nodes (dot) and a sink (square). A black node has a single packet and a white node has no packet. The numbers besides a link denote the time slots when the link is scheduled to forward a packet. Links with the same number are activated together.

respectively, in a small tree, where data forwarding takes 5 time slots to collect all the 4 packets, while data aggregation takes 3 time slots only. It is expected that the saving using aggregation is much bigger for large networks.

For a subset $S \subseteq V_T$, let $f(S)$ denote the utility associated with the data sensed by the nodes in S. The value of f can be learned from historical data or domain knowledge. A common example of f is an additive utility, where each node a has a weight $w_a \in \mathbb{R}^+$ and $f(S) = \sum_{a \in S} w_a$. We consider a more general class of utility functions that satisfy the following conditions where f is (a) normalized, i.e., $f(\emptyset) = 0$; (b) nondecreasing, i.e., $f(S) \leq f(R)$ if $S \subseteq R \subseteq V_T$; and (c) submodular, i.e., $f(S \cup \{a\}) - f(S) \geq f(R \cup \{a\}) - f(R)$ for any $S \subseteq R \subseteq V_T$ and $a \in V_T \backslash R$. Note that an additive f satisfies all the three conditions with equality holds in the last condition. In this paper, we consider a general monotone submodular f and study the following optimization problem:

Problem 1: $\max_{S \subseteq V_T} f(S)$ s.t. $L_F(S) \leq D$ (resp. $L_A(S) \leq D$).

Hardness of the problem: Problem 1 is strongly NP-hard for a general submodular utility. To see this, consider a tree of height 1. Then for both data forwarding and data aggregation, the problem becomes maximizing a monotone submodular function subject to a cardinality constraint, which includes the maximum set covering problem as a special case and no $(1 - 1/e + \epsilon)$-approximation is possible for any $\epsilon > 0$, unless P = NP [34]. On the other hand, in the special case when f is additive, a polynomial time solution has been developed for data aggregation [13].

IV. Deadline Constrained Data Forwarding

In this section, we study Problem 1 for data forwarding. For a general monotone submodular utility, we approach the problem using a standard greedy technique (Section IV-A). The *main challenge* is to show that the algorithm can be implemented efficiently (Section IV-B), and achieves a small approximation ratio (Section IV-C). We further show that the greedy algorithm is actually optimal in some special but important cases (Section IV-C).

A. Greedy sensor selection

The problem of maximizing a submodular set function subject to a cardinality constraint and more general constraints has received a lot of attention in combinatorial optimization [4], [11], [29]. Among the solutions proposed, the most intuitive one is based on a simple greedy heuristic that always chooses the nodes with maximum *marginal utility*. For a given set of nodes S and a node $a \notin S$, the marginal utility of a with respect to S is defined as $f(S \cup \{a\}) - f(S)$. In many problem settings, the simple greedy heuristic can be implemented in an efficient way. Moreover, it can often provide a good performance guarantee depending on the structure of the feasible set. Therefore, we choose to adapt the greedy algorithm to our problem and study its performance accordingly.

When applied to our problem, the standard greedy algorithm can be illustrated as follows (see Algorithm 1). The algorithm starts with an empty set S. In each step, a node with maximum marginal utility subject to the deadline constraint is added to S (lines 3-6). To achieve this, we first identify the set of nodes that satisfy the following two conditions (line 3): (1) the node has not been selected before; and (2) when the node is added to S, the new set is still feasible, that is, the deadline constraint is still satisfied. The algorithm finishes if no such nodes exist (line 4). Otherwise, among all these nodes, the one has the maximum marginal utility is chosen (line 5-6).

Algorithm 1 A greedy algorithm for utility maximization under a deadline constraint

Input: T, f, D; Output: $S \subseteq V_T$

1: $S \leftarrow \emptyset$;
2: **while** true **do**
3: $\quad A \leftarrow \{a : a \in V_T \backslash S$ and $L_F(S \cup \{a\}) \leq D\}$;
4: $\quad$ **if** $A = \emptyset$ **then** break;
5: $\quad a \leftarrow \text{argmax}_{a \in A} f(S \cup \{a\}) - f(S)$;
6: $\quad S \leftarrow S \cup \{a\}$;
7: **return** S

We note that the greedy algorithm requires access to two oracles, a value oracle that returns $f(S)$ for a given set of nodes S, and a membership oracle that decides whether $L_F(S) \leq D$, i.e., checking the feasibility of a given set S. We assume an exact value oracle is readily available when stating our results, which also extends to the case when only an α-approximate value oracle is available as we remark in Section IV-C. On the other hand, we show, in the next section, that $L_F(S)$ can be determined for any S in an efficient way; hence, a membership oracle can be easily implemented for our problem. With these two oracles, the greedy algorithm serves as an efficient solution to our problem.

In addition to its efficiency, the greedy algorithm can often ensure a guaranteed performance when the set of feasible solutions has a nice structure. For instance, a classic result in combinatorial optimization [11] is that the greedy algorithm achieves a fraction 1/2 of the optimal solution when f is monotone submodular and when the feasible set is a matroid (formally defined in Section IV-C), which includes the cardinality constraint as a special case. In our problem, the feasible set is defined as $\{S \subseteq V_T : L_F(S) \leq D\}$. Therefore, we study the structure of $L_F(\cdot)$ in the next section, which enables us to prove the performance guarantee of Algorithm 1 in Section IV-C.

B. Minimum delay data forwarding

In this section, we show that the minimum delay $L_F(S)$ for a set $S \subseteq V_T$ can be efficiently determined; hence the feasibility of S can be easily checked. The problem of minimum delay data forwarding in a tree network subject to 1-hop interference has been studied in [9]. However, there is an error in a key construction. In the following, we provide a brief overview of the results in [9] by focusing on correcting the construction, which is needed for both determining $L_F(S)$ and the analysis of Algorithm 1 in the next section.

To find the minimum-delay schedule in a tree network, the equivalent problem of data dissemination is considered in [9], where packets flow from s_0 to the internal nodes subject to the 1-hop interference constraint, and the objective is to find a schedule of minimum length such that all the destination nodes receive the corresponding packets. The schedule can then be converted back to a schedule of equal length for data forwarding. The main observation in [9] is that an optimal schedule for a tree network can be derived by considering a multi-line network, which is obtained from the original tree graph by mapping each subtree rooted at s_0 to a line, where all the packets at level l of the subtree stay at level l in the line. Note that by our assumption that each node has at most one packet in the original tree network, each level-1 node in the equivalent line network has at most one packet, but the rest of the nodes in the line network may have more than one packet.

First consider the easy case when the sink has a single child, which can be mapped to a single line network, denoted as T. The following schedule is shown to be optimal in [9]: s_0 first sends packets towards the furthest node (that should have the packets), and then the second furthest node and so on. A node between s_0 and the destination of a packet forwards the packet in the next time slot after it receives the packet, and s_0 waits if a level-1 node has a packet to forward. Let v_i denote the number of level-i packets, and let $\mathbf{v} = (v_1, v_2, ..., v_m)$, where m is the maximum level of any packets in the network. The minimum delay for delivering $\mathbf{v}$, denoted as $L_F(\mathbf{v})$, can be determined as follows:

$$L_F(\mathbf{v}) = \begin{cases} \max\limits_{1\le i\le m-1}(i-1+v_i+2\sum\limits_{j=i+1}^{m} v_j) & \text{if } m>1, \\ v_1 & \text{if } m\le 1. \end{cases} \quad \text{(III.1)}$$

Next consider the more general case when T is composed of K lines rooted at s_0. To derive an optimal schedule, the idea is to consider a new tree T' constructed from T, where T' also has K lines and the packets on the k-th branch of T are redistributed on the k-th branch of T'. However, as we will show, the construction of T' in [9] is problematic and may lead to an incorrect formula of $L_F(\cdot)$. Below we first present a corrected construction as follows. First, let v_i^k denote the number of level-i packets on the k-th line of T, and let $\mathbf{v} = \{v_i^k\}_{i\in\{1,...,h_T\},k\in\{1,...,K\}}$ denote the distribution of packets on T. Let $\{p_1, ..., p_{n_k}\}$ denote the set of packets that are at level 2 or more on branch k of T, ordered in a nondecreasing order of their levels (with ties broken arbitrarily), with n_k being the number of these packets. Let t_i^k denote the minimum delay required for forwarding packets $p_1, ..., p_i$ as well as the level-1 packets on branch k to s_0, which can be determined by resorting to the single line case. Packets in T are redistributed in T' such that the number of level-i packets on the k-th line of T', denoted as $v_i'^k$, is determined as follows: (1) $v_1'^k = v_1^k$, (2) $v_{t_i^k}'^k = 1, 1 \le i \le n_k$, and (3) $v_i'^k = 0$ for other i's. That is, the level-1 packets stay where there are. Each packet p_i that is at level 2 or more is moved to level t_i^k. See Figure 2(a) for an example. T' constructed this way has some nice properties, which can be formalized to show the following result.

Proposition IV.1. *(1) An optimal schedule for T can be converted to a schedule of same length for T' and vice-versa. (2) Each node at level 2 or more in T' has at most 1 packet; (3) Two nodes in T' having packets are separated by at least one node that has no packet.*

Proof: Properties (2) and (3) are clear from the construction. To establish the first property, consider a data distribution schedule $\mathcal{S}$ for T with the following form: In each time slot, s_0 decides which child to forward a packet to if none of the level-1 nodes have a packet. Otherwise, s_0 waits. All the packets on the same line are then forwarded using the optimal schedule for a single line network discussed above. We then construct a data distribution schedule $\mathcal{S}'$ for T' as follows. $\mathcal{S}'$ mimics the decisions at s_0 in $\mathcal{S}$, and also forwards packets on each line using the optimal schedule for a single line. Then the k-th lines in both networks are scheduled the same number of time slots for all k. Our construction then guarantees that $\mathcal{S}'$ can distribute all the packets in T' as needed. The same argument applies to the other direction as well. Hence the claim follows. ■

By making use of properties (2) and (3), an optimal schedule for T' can be established as in [9], which can then be converted back to an optimal schedule for T by property (1). Let $\mathbf{v}'_j = \sum_{k=1}^{K} v_j'^k$. The minimum delay for forwarding packets in $\mathbf{v}$ to s_0 can then be determined as follows, where m denotes the highest level of packets in T'.

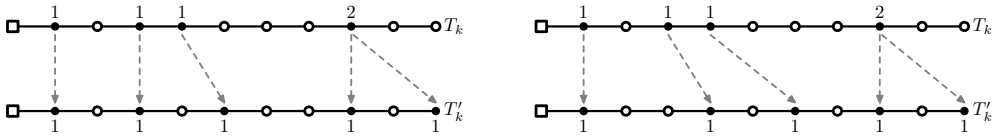
$$L_F(\mathbf{v}) = \max_{1\le i\le m}(i-1+\sum_{j=i}^{m}\mathbf{v}'_j). \quad \text{(III.2)}$$

Critique of the construction of T' in [9]: The original construction of T' in [9] redistributes packets in a different way. In particular, packets at level-1 stay where they are as we do, but each packet p_i at level 2 or more is moved to level $t_{n_k}^k - 2(n_k - i), i = 1, ..., n_k$[1]. See Figure 2(b) for an example. The T' built in this way still satisfies the properties (2) and (3) in Proposition IV.1. But, T' and T may require different number of time slots to forward their packets to s_0 as shown in Figure 3, which leads to an incorrect formula of $L_F(\cdot)$.

C. Analysis of Algorithm 1

In this section, we show that Algorithm 1 approximates the optimal solution to Problem 1 within a constant factor. Our analysis is based on a classic result of submodular maximization over p-systems. We first introduce some terms.

[1]The formula given in [9] has a different but equivalent form.



(a) Our construction of T'. (b) The construction of T' in [9].

Fig. 2. Construction of the k-th branch of T'. The number of packets at each node with at least one packet is labeled.

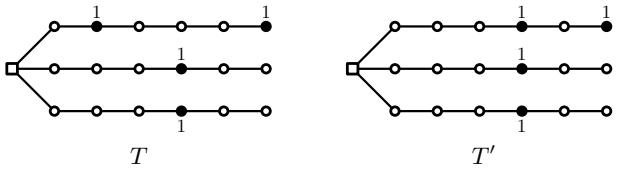


Fig. 3. An example that shows that T' constructed in [9] may be inequivalent to T. In T, it takes 6 time slots to forward the 4 packets to the sink, while it takes 7 time slots in T' due to the incorrect shifting of a packet in the topmost branch.

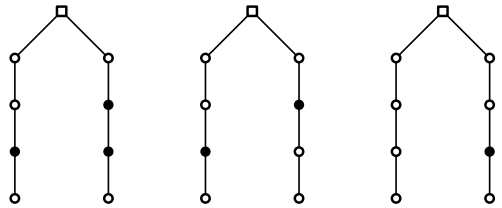


Fig. 4. (a) A tree with three packets (on the black nodes). The minimum delay is 4 time slots. Assuming D = 3, then (b) shows a maximal feasible subset with 2 packets, and (c) shows another maximal feasible subset with 1 packet.

Definition IV.1. *Given a ground set A and a collection of subsets $\mathcal{I} \subseteq 2^A$, $(A, \mathcal{I})$ is called an **independence system** if (1) $\emptyset \in \mathcal{I}$, and (2) for every $S \subseteq A$, if $S \in \mathcal{I}$, and $S' \subseteq S$, then $S' \in \mathcal{I}$.*

Given an independence system $(A, \mathcal{I})$ and a set $S \subseteq A$, let $\mathcal{B}(S)$ denote the set of maximal independent sets in S, i.e., $\mathcal{B}(S) = \{S' \subseteq S : S' \in \mathcal{I}$ and there is no $a \in S \backslash S'$ such that $S' \cup \{a\} \in \mathcal{I}\}$.

Definition IV.2. *An independence system $(A, \mathcal{I})$ is called a **p-system** if for all $S \subseteq A$, $\frac{\max_{S' \in \mathcal{B}(S)} |S'|}{\min_{S'' \in \mathcal{B}(S)} |S''|} \leq p$.*

We note that a 1-system is more commonly known as a *matroid*. As an example of p-systems, consider the set of matchings $\mathcal{M}_G$ in a graph G with edge set E. $(E, \mathcal{M}_G)$ is clearly an independence system since if a subset of edges $E' \subseteq E$ forms a matching, so does any subset of E'. Furthermore, for any $E' \subseteq E$, $\mathcal{B}(E')$ consists of all the maximal matchings contained in E'. It is well known that in any graph, the size of a maximum matching is at most twice the size of a maximal matching. Hence $(E, \mathcal{M}_G)$ is a 2-system. The following result for p-systems is classic [29]:

Lemma IV.1. *For a p-system $(A, \mathcal{I})$ and a monotone submodular function f defined on the subsets of A with $f(\emptyset) = 0$, the problem of maximizing $f(S)$ subject to $S \in \mathcal{I}$ can be approximated by a greedy algorithm (similar to Algorithm 1) within a factor of $1/(p+1)$. The factor improves to $1/p$ when f is additive.*

Remark: A slightly more general result is proved in [4], where it is shown that a ratio $\alpha/(p + \alpha)$ can be achieved by the greedy algorithm when only an α-approximate oracle for f is available for some $\alpha < 1$. In the following, we assume that an exact oracle for f is available for simplicity. Extensions of our results to the general case are straightforward.

The main idea for establishing the approximation factor of Algorithm 1 is to show that the feasible set subject to the deadline constraint forms a p-system for a small p. For instance, Figure 4 gives an example where a tree with 3 packets forms a 2-system. We then proceed to prove the main result of this section.

Lemma IV.2. *Let $\mathcal{I}_F$ denote the set of feasible solutions to Problem 1 for data forwarding, i.e., $\mathcal{I}_F = \{S \subseteq V_T : L_F(S) \leq D\}$. Then $(V_T, \mathcal{I}_F)$ is a 1-system when the sink has a single child, and is a 2-system in general.*

Proof sketch: It is clear that $(V_T, \mathcal{I}_F)$ is an independence system. By converting each subtree rooted at s_0 to an equivalent line as mentioned above, it suffices to consider a single line network for the first part and a multi-line network for the second part. The main idea of the proof is that, for any two feasible subsets X and Y, if $|X| < |Y|$ (or $2|X| < |Y|$ in the multi-line case), then there is a packet in Y that can be added to X such that X is still feasible. The proof for the more difficult multi-line case relies on our construction of T' and Proposition IV.1. The detailed proof is provided in the appendix.

The following proposition then follows directly from Lemma IV.1 and Lemma IV.2.

Proposition IV.2. *Algorithm 1 is a 1/2-approximation to Problem 1 for data forwarding if the sink has a single child in T, and a 1/3-approximation for a general T. When f is additive, Algorithm 1 is optimal in the former case and a 1/2-approximation in general.*

Remark: For a given deadline constraint D, it is clear that the maximum achievable utility depends on the tree topology. The above result reveals that the performance of the greedy algorithm is also related to the tree topology. An interesting open problem is then to study the joint optimization of routing tree construction and data collection, which is left as part of our future work.

According to Proposition IV.2, the greedy algorithm is optimal for additive utility when the sink has a single child, which, however, is rarely satisfied in practice. In the following, we show that the greedy algorithm remains optimal for a

general tree structure if f is additive and the distribution of node weights satisfies a monotonicity condition. This condition includes the special case where each node has the same weight and the problem is to maximize the total number of packets collected by a deadline (recall that each node has at most one packet). The result is based on the following observation (the proof is similar to Lemma IV.2 and is provided in the appendix).

Lemma IV.3. *Consider a set of packets on a routing tree T and let T' denote the equivalent multi-line structure, as discussed in Section IV-B. Then $(V_T, \mathcal{I}_F)$ is a matroid (i.e., a 1-system) if T' and T are the same, i.e., no packet redistribution is needed in T'.*

Note that the above condition is satisfied if and only if in the multi-line representation of T, each node has at most 1 packet and two nodes on the same line that have packets are separated by at least one node without packets. Using this lemma, we then derive the following optimality condition.

Proposition IV.3. *Algorithm 1 is optimal if f is additive and satisfies the following monotonicity condition: For the set of nodes in T that have packets, the weight of a lower level node is no less than the weight of a higher level node that belongs to the same subtree of the sink.*

Proof: First note that the condition on node weight distribution implies a hereditary property in optimal solutions to Problem 1. That is, there is an optimal solution where if a node of level i is included, so are all the lower level nodes with packets in the same branch. Otherwise, the level i node can be replaced by a lower level node with packet, which does not increase delay and does not hurt utility by the assumption. Based on this property, we then construct a multi-line network $\bar{T}$ and redistribute packets in T to $\bar{T}$ such that for a given deadline constraint, the same optimal utility is achieved in both trees. The construction of $\bar{T}$ is similar to the construction of T' in Section IV-B with one difference. When redistributing the set of packets on the same node in the multi-line representation of T, or equivalently the set of packets at the same level of a subtree of the sink in T, instead of breaking the tie arbitrarily as we did before, the one with higher weight is kept closer to the sink. The equivalence of the two trees in terms of optimal utility under the same deadline constraint then follows from the hereditary property and Proposition IV.1. Furthermore, $\bar{T}$ satisfies the condition of Lemma IV.3. Hence, the set of feasible solutions in $\bar{T}$ forms a matroid, and Algorithm 1 is optimal by Lemma IV.1. ■

Note that the monotonicity condition is trivially satisfied when all the nodes have the same weight. Hence the greedy algorithm is optimal for maximizing the total number of packets collected by a deadline.

V. Deadline Constrained Data Aggregation

In this section, we study the deadline constrained utility maximization problem for data aggregation, which turns out to be much harder than the case of data forwarding. Some evidences on the hardness of this problem are discussed at the end of this section. While a greedy algorithm similar to Algorithm 1 can be applied to this case as well, proving a non-trivial bound for it eludes us so far. In the following, we provide a bi-criteria approximation to this problem. We first observe that due to the monotonicity of the utility function f, there is an optimal solution to Problem 1 such that if a node a is selected, all the nodes along the unique path from a to the sink s_0 are also included, since the minimum delay does not increase by doing so. Hence a feasible solution to Problem 1 for data aggregation is a subtree of T rooted at the sink. Without loss of generality, we assume in this section that each node in T has *exactly one* packet. Let $L_A(T)$ denote the minimum delay for aggregating all the packets in T to the sink, that is, $L_A(T) = L_A(V_T)$. We begin with proving some bounds on $L_A(T)$ (Section V-A), which will be needed in analyzing our algorithm. We then consider the data aggregation problem under the "clique" interference model and propose a greedy solution to it (Section V-B), which is then used as a subroutine in deriving the bi-criteria approximation to Problem 1 (Section V-C). Since a solution found using the bi-criteria approximation may violate the deadline constraint, it may not be directly applicable in practice. We therefore propose a searching procedure that utilizes the bi-criteria approximation to obtain a feasible solution with performance guarantee in Section V-D.

A. Upper and lower bounds on minimum delay

We first note that due to the aggregation nature, among the schedules that achieve the minimum delay $L_A(T)$, there is one that satisfies the following condition: Each node should wait until it receives the packets from all its children and then forwards one aggregated packet to its parent. We then establish the following bounds on $L_A(T)$.

Proposition V.1. $h_T \leq L_A(T) \leq h_T \Delta_T$.

Proof: The lower bound is obvious since there is at least one level-h packet, which takes at least h time slots to reach the sink. To see the second inequality, first expand T to a complete Δ_T-ary tree T_1 such that $h_{T_1} = h_T$ and there is one packet at each node of T_1. We then show that $L_A(T_1) \leq h_T \Delta_T$ by considering the following schedule. For each node $a \in T_1$ except the sink, let $t(a)$ denote the time slot when a is scheduled to forward a packet to its parent (each node only needs to be scheduled once as argued above). The value of $t(a)$ is determined in a top-down way. First consider the level-1 nodes. Index the nodes in an arbitrary order. Then the i-th child of the sink is scheduled at time $(h_T - 1)\Delta_T + i$, $i = 1, ..., \Delta_T$. Suppose $t(b)$ has been determined for a node b. Then the i-th child of b is scheduled at time $t(b) - \Delta_T + i - 1, i = 1, ..., \Delta_T$. It is easy to see that this schedule has a delay of $h_T \Delta_T$. It follows that $L_A(T) \leq L_A(T_1) \leq h_T \Delta_T$. ■

Given the above bounds, we can derive the following simple $(\Delta_T, 1)$-approximation to Problem 1. Given the deadline constraint D, only the nodes at level $h = \min\{D, h_T\}$ or less can possibly be reached. Hence by selecting all the nodes of level h or less, the algorithm achieves at least the optimal utility. These nodes form a subtree of T, whose minimum

delay is bounded by $h\Delta_T$ by the above proposition. Note that this algorithm does not access the utility function at all.

B. Data aggregation under "clique" interference model

In this section, we study a problem that is similar to Problem 1, but replaces the 1-hop interference model with the clique model, where at any time, at most one node can transmit a packet to its parent. Note that, under this model, the minimum delay for aggregating all the packets in a routing tree to s_0 is equal to the size of the tree. The solution to this problem will be used as a subroutine for solving Problem 1. Formally, let $T(S)$ denote the minimum subtree of T that is rooted at s_0 and spans the nodes in S, the new optimization problem is:

$$\textbf{Problem 2:} \quad \max_{S\subseteq V_T} f(S) \text{ s.t. and } |T(S)| \leq D.$$

Let $\mathcal{I}^\infty$ denote the set of feasible solutions to Problem 2, i.e., $\mathcal{I}^\infty = \{S \subseteq V_T : |T(S)| \leq D\}$. It is clear that $(V_T, \mathcal{I}^\infty)$ is an independence system. We further have the following result:

Proposition V.2. *$(V_T, \mathcal{I}^\infty)$ is a h_T-system.*

Proof: Consider a subset $S \subseteq V_T$, and two maximal independent sets X and Y in S. Without loss of generality, we assume both X and Y are non-empty. Then S contains at least one node at level D or less. We will show that $|Y|/|X| \leq h_{T(S)}$, which implies the proposition. First it is clear that $|Y| \leq \min(D, |S|)$ since a subtree of size D can span at most D nodes in S. Write $D = kh_{T(S)} + l, k \geq 0, 0 \leq l < h_{T(S)}, k, l \in \mathbb{N}$. We will argue for all the possible values of k and l. (1) $k = 0$. Then $D = l < h_{T(S)}$. Hence $|Y|/|X| \leq D < h_{T(S)}$. (2) $k \geq |S|$. Then $D \geq |S|h_{T(S)}$. Hence $|X| = |Y| = |S|$, $|Y|/|X| = 1 \leq h_{T(S)}$. (3) $0 < k < |S|, l = 0$. Then $|X| \geq \min(|S|, k) = k = D/h_{T(S)}$. Hence $|Y|/|X| \leq h_{T(S)}$. (4) $0 < k < |S|, l > 0$. Then $|X| \geq k$. We distinguish the following two cases. (4.a) $|X| \geq k + 1$. Then $|Y|/|X| \leq D/(k+1) = (kh_{T(S)} + l)/(k+1) < (k+1)h_{T(S)}/(k+1) = h_{T(S)}$. (4.b) $|X| = k$. Then since X is maximal independent and $k < |S|$, every node in X is at least $l + 1$ hops away from s_0. Furthermore, every node in Y is at least $l + 1$ hops away from s_0, since otherwise a node at level l or less in Y can be added to X without violating the cost constraint, which contradicts the assumption that X is maximal. It follows that $|Y| \leq D - l$. Hence $|Y|/|X| \leq (D-l)/k = kh_{T(S)}/k = h_{T(S)}$. ∎

Figure 5 shows an example where the factor is tight. By the above proposition, a greedy algorithm similar to Algorithm 1 achieves a factor $\frac{1}{h_T+1}$-approximation to Problem 2 by Lemma IV.1. Furthermore, the feasibility of $S \cup \{a\}$ when adding a node a to a partial solution S can be easily checked by following the unique path that connects a to a node in $T(S)$.

C. A bi-criteria approximation

By utilizing the greedy algorithm for Problem 2, we propose the following algorithm to Problem 1 for data aggregation.

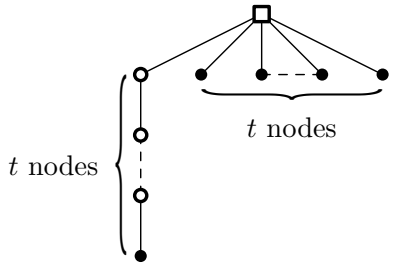


Fig. 5. Consider a subset S consisting of the $t+1$ black nodes in the tree, and assume $D = h_{T(S)} = t$. Then the bottom left node forms a maximal independent set in S, so do the t level-1 nodes.

All the subtrees in the algorithm are rooted at s_0. First, T is truncated to include only nodes at level $h = \min(h_T, D)$ or less (line 1), as these are the nodes that can possibly be reached by the deadline D. A subtree T_1 of maximum size subject to the deadline constraint is then found (line 2), which can be implemented using the dynamic programming algorithm in [13] for additive utility by associating with each node a unit weight. Line 3 invokes the greedy algorithm to Problem 2 to find a subtree T_2 that achieves (approximately) the maximum utility with its size bounded by $|T_1|$. Note that the minimum delay with respect to T_2 could be larger than D. T_2 is then expanded without further increasing the delay (line 4). This can be done in an arbitrarily way and does not effect the approximation ratio that we will derive. For instance, we can again use the greedy approach. To check the feasibility of a partial solution, the minimum delay of a subtree needs to be computed. An efficient algorithm is provided at the end of this subsection.

Algorithm 2 A bi-criteria approximation for data aggregation under a deadline constraint

Input: T, f, D; Output: T_2: a subtree of T

1: $h \leftarrow \min(h_T, D)$ and remove all the nodes in T at level $h+1$ or more;
2: Find a subtree $T_1 \subseteq T$ (rooted at s_0) of maximize size such that the minimum delay $L_A(T_1) \leq D$;
3: Find a maximum utility subtree $T_2 \subseteq T$ (rooted at s_0) with its size bounded by $|T_1|$ using the greedy algorithm;
4: Expand T_2 greedily without further increasing the delay;
5: **return** T_2

Algorithm 2 achieves a bi-criteria approximation to Problem 1 as stated in the following proposition.

Proposition V.3. *Let OPT denote the optimal utility for a given deadline D. Let $\mathbb{T}$ denote the set of subtrees of T rooted at s_0. Then for the subtree T_2 found by Algorithm 2, we have $f(T_2) \geq \frac{1}{\min(h_T, D)+1}$OPT, and $L_A(T_2) \leq \rho_T D$, where $\rho_T \triangleq \max_{T', T'' \in \mathbb{T}} \left\{ \frac{L_A(T')}{L_A(T'')} : h_{T'} \leq h_{T''}, |T'| \leq |T''| \right\}$, which is bounded by Δ_T.*

Proof: We first study the utility of T_2 compared with OPT. Let T^{opt} denote the optimal subtree as a solution to Problem 1, and $T^{opt(2)}$ the optimal solution to the Problem 2 with budget $|T_1|$. Since $|T^{opt}| \leq |T_1|$, $f(T^{opt}) \leq f(T^{opt(2)})$

Algorithm 3 A searching algorithm for utility maximization

Input: T, f, D; Output: $\tilde{T}$: a subtree of T

1: **for** $d \leftarrow D, 1$ **do**
2: $T_2 \leftarrow$ call Algorithm 2 with parameters T, f, and deadline d;
3: **if** $L_A(T_2) \leq D$ **then**
4: $\tilde{T} \leftarrow T_2$; break
5: **return** $\tilde{T}$

by the optimality of $T^{opt(2)}$. Hence $f(T_2) \geq \frac{f(T^{opt(2)})}{h_{T_2}+1} \geq \frac{f(T^{opt})}{h+1} = \frac{OPT}{\min(h_T,D)+1}$.

We then bound $L_A(T_2)$. It is not hard to see that T_1 has at least one level-h node. Hence $h_{T_1} = h$. Since $h_{T_2} \leq h$ and $|T_2| \leq |T_1|$, we have $L_A(T_2) \leq \max(\rho(T)L_A(T_1), D) \leq \rho(T)D$. Furthermore, for any $T', T'' \in \mathbb{T}$ where $h_{T'} \leq h_{T''}$, we have $L_A(T') \leq h_{T'}\Delta_{T'} \leq h_{T''}\Delta_T$ and $L_A(T'') \geq h_{T''}$ by Proposition V.1. Hence $\rho_T \leq \Delta_T$. . ∎

We expect that a routing tree built for a sensor network usually has a relatively small height to bound the worst case delay, especially in a relatively dense deployment. We have evaluated the typical values of ρ_T in a randomly generated T of a given maximum degree (≤ 10) and height (≤ 6), by randomly selecting pairs of subtrees T', T'' in T, and observe that the values of ρ_T are upper bounded by 1.5 on average and by 3 in the worst case. The detailed simulation results are given in Section VI.

Minimum delay data aggregation: We now provide an efficient algorithm to find $L_A(T)$, which is needed to implement the last step of Algorithm 2 in a greedy way, and is interesting by itself. We note that the algorithm in [13] for maximizing additive utility under a deadline constraint combined with a binary search can be used to solve this problem. However, our solution is more efficient. Let T_v denote the subtree rooted at node v. Let $L'_A(T_v)$ denote the minimum delay for aggregating data in T_v to v. Then $L_A(T) = L'_A(T_{s_0})$. Our algorithm computes $L'_A(T_v)$ in a bottom-up way as follows. Let $v_1, v_2, ..., v_m$ denote the set of children of node v, and suppose $L'_A(T_{v_1})$ through $L'_A(T_{v_m})$ have been found. We then create a m-line network rooted at v, with a single packet at level $L'_A(T_{v_k}) + 1$ on the k-th line, for $k = 1, ..., m$. $L'_A(T_v)$ can then be determined by Equation (III.2). This algorithm takes $O(n\Delta_T L_A(T))$ time.

D. A feasible approximation

For a given deadline constraint, the solution found by Algorithm 2 is not directly applicable when it violates the deadline constraint. To obtain a feasible solution while retaining a performance guarantee, we propose a searching approach by utilizing Algorithm 2 as a subroutine. We first state a simplified version in Algorithm 3 and characterize its performance in Proposition V.4. We then discuss a refinement of the searching procedure that achieves better performance.

The procedure starts with the deadline constraint D, and searches over $d = D, D-1, ..., 1$. For each value of d, Algorithm 2 is invoked with deadline d to obtain a subtree T_2. The procedure continues until the T_2 found is feasible subject to D. It is clear that Algorithm 3 always finds a feasible solution to the utility maximization problem. The following proposition characterizes its performance.

Proposition V.4. *Let $OPT(D)$ denote the optimal achievable utility subject to a deadline D, and $ALG(D)$ the utility obtained by Algorithm 3. Then $ALG(D) \geq \frac{1}{\min(h_T,d_0)+1} OPT(d_0)$, where $\frac{D+1}{\rho_T} - 1 \leq d_0 \leq D$.*

Proof: First if $D \geq L_A(T)$, then all the packets in T can be aggregated by D and Algorithm 3 is clearly optimal. Assume $D < L_A(T)$. Let d_0 denote the value of d when the searching algorithm stops. Let $T_2(d)$ and $T_1(d)$ denote the subtrees found in Algorithm 2 given a deadline d. We then have $ALG(D) = f(T_2(d_0)) \geq \frac{1}{\min(h_T,d_0)+1} OPT(d_0)$ by Proposition V.3. It is clear that $d_0 \leq D$. To show the lower bound, note that $L_A(T_2(d_0+1)) \geq D+1$ by the definition of d_0, and $L_A(T_2(d_0+1)) \leq \rho_T L_A(T_1(d_0+1)) = \rho_T(d_0+1)$ by the definition of ρ_T. Hence $\rho(d_0+1) \geq D+1$ and $d_0 \geq \frac{D+1}{\rho_T} - 1$. ∎

To further improve the performance of the above algorithm, one approach is to refine the searching procedure. Let $c(d)$ denote the size of T_1 found in Algorithm 2 (line 2) given deadline d, that is, $c(d)$ is the maximum cardinality of a subtree (rooted at the sink) of T subject to d. Instead of searching over $D, D-1, ..., 1$, a better way is to search over $c = c(D), c(D)-1, ..., 1$. For a given value of c, the maximum d such that $c(d) \geq c$ and $c(d-1) < c$ can then be determined, and is used to invoke Algorithm 2 to find T_2. The searching procedure again stops when T_2 is feasible respecting to D. Since multiple consecutive values of c can correspond to the same d, this approach provides a fine-grained searching compared with the original algorithm. On the other hand, it is not hard to see that the performance bound in Proposition V.4 still holds. This is the version we will implement and evaluate in simulations.

E. Discussion

We note that the utility maximization problem under the full interference model is related to the group Steiner problem. In the group Steiner problem, an edge-weighed graph is considered, where every node covers a subset of targets from a ground set. The problem is to find a subset of nodes so that the total number of targets covered by them is maximized subject to a constraint on the weight of the Steiner tree spanning the nodes. For general edge weights, it is known that even when the graph is a tree, the problem does not have $\frac{1}{log^{1-\epsilon}m}$-approximation for any fixed $\epsilon > 0$, where m is the total number of targets in the ground set, unless $NP \subseteq ZTIME(n^{\text{polylog}(n)})$ [12]. Furthermore, no combinatorial algorithm is known to achieve a ratio close to this bound. Our problem under full interference model can be viewed as a variant of the group Steiner problem with a general submodular function defined on a tree graph with unit edge weight.

Although we did not establish the tightness of our results, we conjecture that our problem under 1-hop interference is

harder than the problem under full interference, and achieving polylogarithmic approximation is hard. We remark that the recursive greedy algorithm in [5] can be adapted to provide a $\frac{1}{\min(h_T, D)}$-approximation, which however has a complexity of $O((\Delta_T!)^{h_T})$, and hence is only applicable when Δ_T is very small, e.g., 2 or 3, and $h_T = O(\log n)$.

VI. Simulations

In this section, we study the performance of our solutions using simulations. Two application scenarios are considered: (1) target point coverage under a disk sensing model, and (2) parameter estimate using maximum-likelihood ratio test. In each case, we consider a properly defined monotone submodular utility function over sensors deployed in a 2-d area. We first consider small networks with 20 nodes, and observe that our algorithms perform very closely to optimal solutions found by exhaustive search. We then consider larger deployments with 100 nodes, and show that our algorithms achieve clearly better utility compared with two heuristics that ignore spatial correlations. We further demonstrate that the parameter ρ_T in our bi-criteria approximation algorithm is typically a small constant by doing a random sampling of subtrees.

A. Sensor selection for 2-d target point coverage

We first consider a small deployment, where 20 sensor nodes are uniformly distributed in a 500×500 2-d region. Each node has a sensing range of 100, and a communication range of 200. There is a link between two nodes if they are within the communication range of each other. The 2-d region is partitioned into a 5×5 grid. Each grid cell is assigned a weight randomly selected in $\{1, ..., 20\}$. 1000 target points are distributed in the region such that the probability that a point is within certain cell is proportional to the weight of the cell, and within a cell, points are uniformly distributed. A target point is covered by a node if the point is within the sensing range of the node.

Under the coverage model defined above, an application may be either interested in an aggregated value over some parameters associated with the target points, or would like to collect these parameters without modification. We do not specify the concrete format of sensed data, which varies over applications. However, we expect that, for a given subset of nodes, the number of points covered by these nodes is a good indicator of their sensing quality in most settings. Hence, we define the utility function to be the number of points covered by a set of nodes, and use this utility to evaluate both data forwarding and data aggregation algorithms. We note that this utility function is additive only in the very special case when all the sensing disks are disjoint. In general, the function is non-additive due to the overlapping of sensing disks, but is easily seen to be monotone submodular.

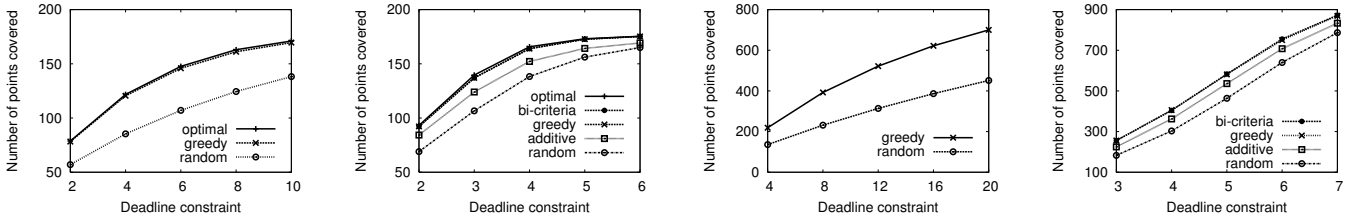
We generate 100 random distributions of points and nodes. For each of them, one node is selected randomly as the sink, and a tree T rooted at the sink is then built by a breadth-first search (BFS), which is repeated 10 times. Algorithm 1 (greedy for short) is then applied to T for data forwarding under various deadline constraints, and Algorithm 3 (bi-criteria for short) is applied for data aggregation. We further compare our algorithms with the following algorithms.

- **Optimal solutions** (optimal for short): For small networks, a set of nodes that achieve the optimal utility subject to the deadline constraint can be found by exhaustive search for both data forwarding and data aggregation.
- **Random sensor selection** (random for short): The algorithm starts with an empty set S. In each step, a node is selected from the set of nodes not in S in a uniformly random way and is added to S subject to the deadline constraint. The procedure repeats until no extra node can be added to S. In our simulations, the random algorithm is repeated 100 times for each T generated.
- **Additive sensor selection** (additive for short): Each node is associated with a weight that is equal to the number of target points covered by the node. The dynamic programming algorithm in [13] is then applied to find a set of nodes of maximum total weight subject to the deadline constraint. Effectively, the algorithm ignores the overlapping of sensing regions and considers an additive utility. Note that this algorithm only applies to data aggregation.

In the case of data aggregation, we further implement a greedy algorithm similar to Algorithm 1 with $L_F(\cdot)$ replaced by $L_A(\cdot)$ and compare it with Algorithm 3.

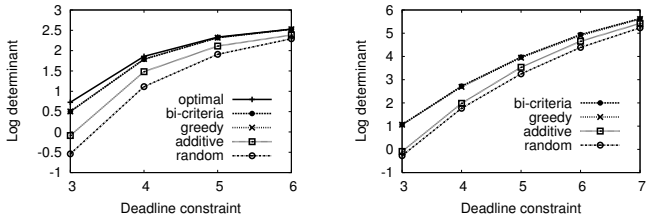
Figures 6(a) and 6(b) show the results under small deployments for data forwarding and data aggregation, respectively. The results are averaged over the random distributions of target points, sensor locations, and the sink node, and the 100 repetitions for the random algorithm. We observe that in both cases, our algorithm perform nearly optimal. Moreover, Figure 6(a) shows that our greedy algorithm achieves $30\% - 40\%$ higher utility than random for data forwarding, and Figure 6(b) shows that our bi-criteria algorithm achieves 20% higher utility on average compared to random, and $5\% - 10\%$ higher utility compared to additive, for data aggregation. We further observe that the performance of bi-criteria and that of greedy are very close in the evaluated scenarios. This may imply that the approximation ratio we obtained through bi-criteria approximation actually provides a good characterization for the greedy algorithm even if proving a bound for it directly is difficult. This raises an interesting question that deserves further investigation.

We next consider a large deployment with 100 nodes uniformly distributed in a 1000×1000 2-d region. 1000 target points are distributed in the 2-d region in a similar way as in the small deployments. The nodes have the same configuration as above. Under the large deployments, Figure 6(c) shows that greedy achieves about 70% higher utility than random in data forwarding. For data aggregation, Figure 6(d) shows that bi-criteria achieves about 40% higher utility for small deadlines and close to 30% higher utility on average compared to random, and 10% improvement on average compared to additive. Moreover, we again observe that the performance of bi-criteria and greedy are very close in data aggregation.



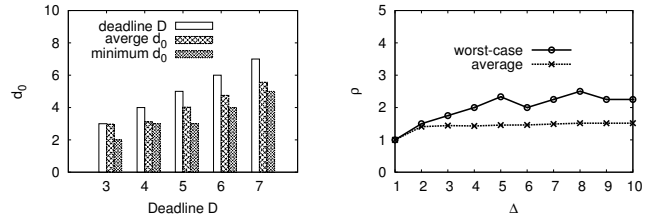
(a) forwarding (20 nodes) (b) aggregation (20 nodes) (c) forwarding (100 nodes) (b) aggregation (100 nodes)

Fig. 6. Number of points covered under various deadline constraints. For data aggregation, the performance of bi-criteria approximation and the greedy algorithm are almost the same in the evaluated scenarios.



(a) 20 nodes (b) 100 nodes

Fig. 7. Sensing quality for parameter estimation under various deadline constraints.



(a) (b)

Fig. 8. (a) The average and minimum values of d_0 when Algorithm 3 stops for trees built on the 100-node sensor network. (b) The average and worst-case values of ρ of a randomly sampled tree with degree bounded by $\Delta = 1, ..., 10$ and height bounded by 6.

B. Sensor selection for parameter estimation

In our second application scenario, we consider the problem of parameter estimation using sensors as studied in [19], [31]. The problem is to estimate an m-dimensional vector $x \in R^m$ from the measurements of sensor nodes. Let $y_i = a_i^T x + n_i$ denote the measurement of node i, where a_i is the measurement vector and n_i is the zero-mean Gaussian noise with variance σ^2. Using the maximum-likelihood (ML) criterion, the estimate of x given the measurements from a set of nodes S has the form $\hat{x} = (\sigma^{-2} \sum_{i \in S} a_i a_i^T)^{-1} \sum_{i \in S} y_i a_i$. As in [19], [31], we assume that both the measurement vectors a_i and σ^2 are known by the fusion center. Hence the sensor network only needs to compute $\sum_{i \in S} y_i a_i$, which can be implemented by data aggregation. We therefore only consider data aggregation in this section. The objective is to select a set of nodes S subject to certain constraints to minimize the estimation error, which is equivalent to maximize $\log\det(\sum_{i \in S} a_i a_i^T)$. This problem is first studied in [19] under the cardinality constraint and a heuristic is proposed based on convex relaxation. The same problem is then studied in [31], where it is shown that a slightly modified objective function $\log\det(\sum_{i \in S} a_i a_i^T + \epsilon I)$ for any $\epsilon > 0$ is monotone submodular; hence, a standard greedy algorithm achieves a constant factor approximation under the cardinality constraint. We apply our algorithms to ML based parameter estimation, which generalize the standard greedy algorithm by considering a deadline constraint instead.

In our simulation, we again consider both a small setting and a large setting as in the first application scenario. In each case, we take $m = 5$ and $\epsilon = 0.001$. The measurement vectors a_i are generated randomly and independently from a Gaussian distribution $\mathcal{N}(0, I/\sqrt{m})$ as in [19], [31]. The random algorithm is again repeated 100 times for each configuration of sensor locations, measurement vectors, and the tree structure. On the other hand, there is no simple way to summarize the contribution of a single node in this setting. Hence, in this scenario, the additive algorithm assigns each node a unit weight. Figures 7(a) and 7(b) show the simulation results for small and large deployments, respectively. We again observe that our algorithms (both bi-criteria and greedy) achieve close to optimal performance in small deployments. Moreover, the performance of greedy is very close to that of bi-criteria in both settings, and both algorithms achieve much better sensing quality than random and additive.

C. The distribution of ρ_T

We have shown that ρ_T is upper bounded by Δ_T in Section V-C, which we expect to be only a loose bound. To estimate its value in the simulation settings discussed above, we have plotted d_0, the value of d when Algorithm 3 stops under various deadline constraints for the point coverage scenario in large deployments. The result is shown in Figure 8(a). Both the minimum value and the average value of d_0 over different node distribution and tree structure are plotted. We observe that D/d_0 is upper bounded by 1.7 in the worst case and by 1.3 in the average case. A similar result is also observed for the parameter estimation scenario (not shown).

We then make some effort to study the typical values of ρ_T in a general routing tree. We start with a complete tree T_0 of degree 10 and height 6, rooted at s_0. For each $\Delta = 1, ..., 10$ and $h = 1, ..., 6$, we find a subtree T_1 of T_0 rooted at s_0 with $\Delta_{T_1} \leq \Delta, h_{T_1} \leq h$, by doing a pre-order traversal of T_0 and removing each edge with probability 0.5 independently. For each pair of Δ and h, 100 T_1 are sampled. For each T_1, 100 subtrees T_2 of T_1 rooted at s_0 are sampled in a similar way. Then for each T_2, 100 subtrees T_3 of T_1 rooted at s_0 are

sampled with the requirement that $h_{T_3} \leq h_{T_2}$ and $|T_3| = |T_2|$. We then compute the worst-case ratio $L_A(T_3)/L_A(T_2)$ over the 10000 pairs of T_2 and T_3, as an estimate of ρ_{T_1} for each T_1. For $\Delta = 1, ..., 10$, the worst-case and average values of ρ_{T_1} (averaged over all the samples of T_1 of various height) are plotted in Figure 8(b). We note that the worst-case ρ_{T_1} is upper bounded by 3 in all the settings we have evaluated and the averages are even smaller, upper bounded by 1.5.

VII. Conclusion

The resource-constrained nature of sensor networks calls for efficient data collection schemes that can better trade off the quality of data and communication overhead. In this paper, we study the problem of sensor selection for maximizing the utility of data collected through a routing tree subject to a deadline constraint. We consider the general class of utility functions that are monotone submodular, and two data collection schemes under the 1-hop interference model. We show that a simple greedy algorithm achieves a small constant factor approximation for raw data collection, and propose a bi-criteria approximation for the harder case when in-network aggregation is applied.

VIII. Appendix

A. Proof of Lemma IV.2

Single-line networks: To prove the first part of the proposition, let T be a single line network of height h, with v_i packets at level-i. Note that $v_1 \leq 1$ by our assumption made in Section III. Consider two subsets of packets X and Y. Let $x_i \leq v_i$ denote the number of level-i packets in X, and $\mathbf{x} = \{x_1, ..., x_h\}$. Define $\mathbf{y} = \{y_1, ..., y_h\}$ similarly. Let m_x and m_y denote the maximum level of any packets in X and that in Y, respectively. Suppose $|X| < |Y|, L_F(X) \leq D$, and $L_F(Y) \leq D$. We will show that there is a packet p in Y that can be added to X to get $\bar{X} = X \cup \{p\}$, subject to the deadline constraint, i.e., there is an index l with $y_l > 0$ such that the new sequence $\bar{\mathbf{x}} = \{x_1, ..., x_{l-1}, x_l + 1, x_{l+1}, ..., x_h\}$ is still feasible.

Let l be the smallest index such that $x_l < y_l$. Such a l must exist since $|X| < |Y|$. Define $g(X, i) = i - 1 + x_i + 2\sum_{j>i} x_j$. Define $g(Y, i)$ similarly. First assume $l \geq m_x$. For any i such that $1 \leq i < l$, let $d_i = x_i - y_i$. We have $d_i \geq 0$ and $\sum_{j>i} x_j + d_i < \sum_{j>i} y_j$ by the choice of l, and therefore $g(\bar{X}, i) = i - 1 + x_i + 2\sum_{j>i} x_j \leq i - 1 + y_i + d_i + 2(\sum_{j>i} y_j - d_i - 1) \leq g(Y, i) \leq D$ by the feasibility of Y. Hence $L_F(\bar{X}) = \max_{1 \leq i \leq l-1} g(\bar{X}, i) \leq D$ by Equation III.1, i.e., $\bar{X}$ is still feasible. Next assume $l < m_x$. For any i such that $1 \leq i < l$, the above argument still applies. For any i such that $l < i < m_x$, we have $\bar{x}_i = x_i$ and hence $g(\bar{X}, i) = g(X, i) \leq D$ by the feasibility of X. For $i = l$, we distinguish the following two cases:

Case 1: $y_i - x_i = 1$. We have $\sum_{j>i} x_j \leq \sum_{j>i} y_j$ by the choice of l. Hence $g(\bar{X}, i) = i - 1 + x_i + 1 + 2\sum_{j>i} x_j \leq i - 1 + y_i + 2\sum_{j>i} y_j = g(Y, i) \leq D$.

Case 2: $y_i - x_i > 1$. Then $i > 1$ by the assumption that $v_1 \leq 1$. We then have $D \geq g(Y, i-1) = i - 2 + y_{i-1} + 2\sum_{j \geq i} y_j \geq i - 2 + 2\sum_{j \geq i} y_j > i - 2 + 2(1 + \sum_{j \geq i} x_j) = i + 2x_i + 2\sum_{j>i} x_j \geq g(\bar{X}, i)$.

It follows that $L_F(\bar{X}) \leq D$ and we have proved the first part of the proposition.

Multi-line networks: To prove the second part of the proposition, let T denote a multi-line network with K lines, with v_i^k packets at the level-i node of branch k. Each level-1 node has at most 1 packet by assumption. Again consider two subsets of packets X and Y. Let $\mathbf{x} = \{x_i^k\}_{1 \leq k \leq K, i \geq 1}$, where $x_i^k \leq v_i^k$ denotes the number of level-i packets on branch k in X. Define $\mathbf{y} = \{y_i^k\}$ similarly. Suppose $2|X| < |Y|$, $L_F(X) \leq D$ and $L_F(Y) \leq D$. We will show that there are indices k and i, such that $y_i^k > 0$, and a level-i packet on branch k in Y can be added to X subject to the deadline constraint. This is trivial if X is empty. Assume $X \neq \emptyset$ in the following.

Let $\mathbf{x}'$ denote the equivalent redistribution of packets in $\mathbf{x}$ to T' constructed in Section IV-B. Define $\mathbf{y}'$ similarly. Let s denote the smallest index such that $\sum_{k, i \geq s} y_i'^k \leq \sum_{k,i} x_i'^k$. We have $s > 1$ and $\sum_{k, i<s} y_i'^k > \sum_{k,i} x_i'^k$ by the assumption that $2|X| < |Y|$, and $\sum_{k, i \geq s-1} y_i'^k > \sum_{k,i} x_i'^k$ by the choice of s. Hence there is k such that $\sum_{i<s} y_i'^k > \sum_i x_i'^k$, that is, on the k-th branch, the number of packets at level $s-1$ or less in $\mathbf{y}'$ is large than the total number of packets in $\mathbf{x}'$. Let t be the largest index less than s such that $y_t'^k = 1$ (recall that $y_t'^k \leq 1$ by the construction of T'). Let $Y_t^k \subseteq Y$ denote the set of packets on the k-th branch that is at level t or less in $\mathbf{y}'$. Define X_t^k similarly. By our construction of T', $L_F(Y_t^k) = t$. We claim that there is a packet in Y_t^k that can be added to X, subject to the deadline constraint. We first distinguish the following two cases:

Case 1: $x_{t+1}'^k = 0$. Since $|Y_t^k| > |X_t^k|, L_F(X_t^k) \leq t, L_F(Y_t^k) = t$, by applying the first part of the proposition, there is a packet p in Y_t^k that can be added to X_t^k to get $\bar{X}_t^k = X_t^k \cup \{p\}$ such that $L_F(\bar{X}_t^k) \leq t$. Now let $\bar{X} = X \cup \{p\}$, and $\bar{\mathbf{x}}$ the packet distribution of $\bar{X}$, and $\bar{\mathbf{x}}'$ the equivalent redistribution in T'. Then by the properties of our construction and $L_F(\bar{X}_t^k) \leq t$, it is clear that $\bar{x}_i'^k = x_i'^k$ for all $i > t$.

Case 2: $x_{t+1}'^k = 1$. Since $|Y_t^k| > |X_{t+1}^k|, L_F(X_{t+1}^k) = t + 1, L_F(Y_t^k) = t < t + 1$, by applying the first part of the proposition, there is a packet p in Y_t^k that can be added to X_{t+1}^k to get $\bar{X}_{t+1}^k = X_{t+1}^k \cup \{p\}$ such that $L_F(\bar{X}_{t+1}^k) = t+1$. Again we have $\bar{x}_i'^k = x_i'^k$ for all $i > t$.

To prove the claim, let m_x denote the maximum index such that $x_{m_x}'^k \neq 0$. First assume $s \geq m_x$. For $i < s$, we have $i - 1 + \sum_{k, j>i} \bar{x}_j'^k \leq i - 1 + 1 + \sum_{k,j} x_j'^k \leq s - 2 + \sum_{k, j \geq s-1} y_j'^k \leq L_F(Y)$, where the first inequality holds since only one packet is added, the second inequality follows from $\sum_{k, i \geq s-1} y_i'^k > \sum_{k,i} x_i'^k$, and the last inequality follows from Equation (III.2). Hence $L_F(\bar{X}) \leq D$ by the feasibility of Y. Next assume $s < m_x$. Then for $i < s$, the above argument still applies. For all $s \leq i \leq m_x$, we have $i - 1 + \sum_{k, j \geq i} \bar{x}_j'^k = i - 1 + \sum_{k, j \geq i} x_j'^k \leq L_F(X)$. Again we have $L_F(\bar{X}) \leq D$ by Equation (III.2) and the feasibility of X. Therefore, we have proved the second part of the proposition. ■

B. Proof of Lemma IV.3

Consider the multi-line representation of a routing tree T, where each node has at most one packet, and in each branch, two nodes with packets are separated by at least one node without packet. Consider two subsets of packets X and Y in T, where $|X| < |Y|, L_F(X) \leq D$, and $L_F(Y) \leq D$. Let x_i denote the total number of level-i packets in X across all the branches, and define y_i similarly. Let l denote the smallest index such that $x_l < y_l$. Then following a similar argument as in the proof of Lemma IV.2, we can show that there is a level l packet p in $Y \backslash X$ that can be added to X such that $X \cup \{p\}$ is still feasible, which implies the statement of the lemma. ■

REFERENCES

[1] KanseiGenie: A Consortium of Sensor Testbeds for At-Scale Federated Experimentation. http://kansei.cse.ohio-state.edu/KanseiGenie/.

[2] Senslab: Very large scale open wireless sensor network testbed. http://www.senslab.info/.

[3] A. Arora, R. Ramnath, E. Ertin, P. Sinha, S. Bapat, et al. ExScal: Elements of an Extreme Scale Wireless Sensor Network. In *Proc. of IEEE RTCSA*, 2005.

[4] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák. Maximizing a Monotone Submodular Function subject to a Matroid Constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

[5] C. Chekuri and M. Pal. A Recursive Greedy Algorithm for Walks in Directed Graphs. In *Proc. of FOCS*, 2005.

[6] X. Chen, X. Hu, and J. Zhu. Minimum data aggregation time problem in wireless sensor networks. *Lecture Notes in Computer Science*, 3794:133–142, 2005.

[7] D. Culler, D. Estrin, and M. Srivastava. Overview of Sensor Networks. *IEEE Computer*, 37(8):41–49, 2004.

[8] A. Eryilmaz and R. Srikant. Fair Resource Allocation in Wireless Networks using Queue-length-based Scheduling and Congestion Control. In *Proc. of IEEE INFOCOM*, 2005.

[9] C. Florens, M. Franceschetti, and R. J. McEliece. Lower Bounds on Data Collection Time in Sensory Networks. *IEEE Journal on Selected Areas in Communications*, 22(6):1110–1120, 2004.

[10] A. Giridhar and P. R. Kumar. Computing and Communicating Functions Over Sensor Networks. *IEEE Journal on Selected Areas in Communications*, 23(4):755–764, 2005.

[11] G.L. Nemhauser, L.A. Wolsey, and M.L. Fisher. An Analysis of Approximations for Maximizing Submodular Set Functions - I. *Combinatorica*, 14(1):265–294, 1978.

[12] E. Halperin and R. Krauthgamer. Polylogarithmic Inapproximability. In *Proc. of STOC*, 2003.

[13] S. Hariharan and N. B. Shroff. Maximizing Aggregated Revenue in Sensor Networks under Deadline Constraints. In *Proc. of IEEE CDC*, Dec. 2009.

[14] S. Hariharan and N. B. Shroff. Deadline Constrained Scheduling for Data Aggregation in Unreliable Sensor Networks. In *Proc. of IEEE WiOpt*, May 2011.

[15] S. Hariharan and N. B. Shroff. On Optimal Energy Efficient Convergecasting in Unreliable Sensor Networks with Applications to Target Tracking. In *Proc. of ACM MobiHoc*, May 2011.

[16] S. Hariharan, Z. Zheng, and N. B. Shroff. Maximizing Information in Unreliable Sensor Networks under Deadline and Energy Constraints. *IEEE Transactions on Automatic Control (TAC)*, 58(6):1416–1429, 2013.

[17] R. Huang, W.-Z. Song, M. Xu, N. Peterson, B. A. Shirazi, and R. LaHusen. Real-World Sensor Network for Long-Term Volcano Monitoring: Design and Findings. *IEEE Transactions on Parallel and Distributed Systems*, 23(2):321–329, 2012.

[18] J-C. Bermond, L. Gargano, and A. A. Rescigno. Gathering with Minimum Completion Time in Sensor Tree Networks. *JOIN*, 11(1-2):1–33, 2010.

[19] S. Joshi and S. Boyd. Sensor Selection via Convex Optimization. *IEEE Transactions on Signal Processing*, 57(2):451–462, 2009.

[20] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the Spread of Influence through a Social Network. In *Proc. of ACM SIGKDD*, 2003.

[21] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg. Near-optimal Sensor Placements: Maximizing Information while Minimizing Communication Cost. In *Proc. of ACM/IEEE IPSN*, 2006.

[22] A. Krause, E. Horvitz, A. Kansal, and F. Zhao. Toward Community Sensing. In *Proc. of ACM/IEEE IPSN*, 2008.

[23] A. Krause, A. Singh, and C. Guestrin. Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. *Journal of Machine Learning Research*, 9:235–284, 2008.

[24] K. Langendoen, A. Baggio, and O. Visser. Murphy loves potatoes: Experiences from a pilot sensor network deployment in precision agriculture. In *Proc. of IEEE IPDPS*, 2006.

[25] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective Outbreak Detection in Networks. In *Proc. of ACM SIGKDD*, 2007.

[26] X. Lin and N. B. Shroff. Joint Rate Control and Scheduling in Multihop Wireless Networks. In *Proc. of IEEE CDC*, 2004.

[27] X. Lin and N. B. Shroff. The Impact of Imperfect Scheduling on Cross-Layer Rate Control in Multihop Wireless Networks. In *Proc. of IEEE INFOCOM*, 2005.

[28] M. J. Neely, E. Modiano and C. Li. Fairness and Optimal Stochastic Control for Heterogeneous Networks. In *Proc. of IEEE INFOCOM*, 2005.

[29] M.L. Fisher, G.L. Nemhauser, and L.A. Wolsey. An Analysis of Approximations for Maximizing Submodular Set Functions - II. *Mathematical Programming Study*, 8:73–87, 1978.

[30] R. Ravi. Rapid rumor ramification: approximating the minimum broadcast time. In *Proc. of FOCS*, 1994.

[31] M. Shamaiah, S. Banerjee, and H. Vikalo. Greedy Sensor Selection: Leveraging Submodularity. In *Proc. of IEEE CDC*, 2010.

[32] R. Szewczyk, A. Mainwaring, J. Polastre, and J. A. D. Culler. An Analysis of a Large Scale Habitat Monitoring Application. In *Proc. of IEEE Sensys*, 2004.

[33] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong. A Macroscope in the Redwoods. In *Proc. of IEEE Sensys*, 2005.

[34] U. Feige. A Threshold of $\ln n$ for Approximating Set Cover. *Journal of the ACM*, 45(4):634–652, 1998.

[35] P.-J. Wan, S. C.-H. Huang, L. Wang, Z. Wan, and X. Jia. Minimum-Latency Aggregation Scheduling in Multihop Wireless Networks. In *Proc. of ACM MobiHoc*, May 2009.

[36] Z. Zheng and N. B. Shroff. Maximizing a Submodular Utility for Deadline Constrained Data Collection in Sensor Networks. In *Proc. of IEEE WiOpt*, May 2012.

Zizhan Zheng (S'07-M'10) received his PhD in Computer Science and Engineering from The Ohio State University, Columbus, OH, USA, in 2010, MS in Computer Science from Peking University, China, in 2005, and BE in Polymer Engineering from Sichuan University, China, in 2002.

He is currently a postdoctoral researcher in the Department of Electrical and Computer Engineering at The Ohio State University. His research is in the broad areas of wireless networking and cyber-physical systems with a special emphasis on the design and control of networked systems.

Ness B. Shroff (S'91-M'93-SM'01-F'07) received the Ph.D. degree in electrical engineering from Columbia University, New York, NY, USA, in 1994.

He joined Purdue University, West Lafayette, IN, USA, immediately thereafter as an Assistant Professor with the School of Electrical and Computer Engineering (ECE). At Purdue, he became a Full Professor of ECE in 2003 and Director of the Center for Wireless Systems and Applications (CWSA) in 2004. In 2007, he joined The Ohio State University, Columbus, OH, USA, where he holds the Ohio Eminent Scholar endowed chair in Networking and Communications in the departments of ECE and Computer Science and Engineering. From 2009 to 2012, he served as a Guest Chaired Professor of wireless communications with Tsinghua University, Beijing, China, and currently holds an honorary Guest Professor with Shanghai Jiaotong University, Shanghai, China. His research interests span the areas of communication, social, and cyberphysical networks. He is especially interested in fundamental problems in the design, control, performance, pricing, and security of these networks.