Maximizing a Submodular Utility for Deadline Constrained Data Collection in Sensor Networks

Zizhan Zheng, Ness B. Shroff

Maximizing a Submodular Utility for Deadline Constrained Data Collection in Sensor Networks

Zizhan Zheng and Ness B. Shroff

***Abstract*—We study the utility maximization problem for data collection in sensor networks subject to a deadline constraint, where the data on a selected subset of nodes are collected through a routing tree rooted at a sink subject to the 1-hop interference model. Our problem can be viewed as a Network Utility Maximization (NUM) problem with binary decisions. However, instead of a separable concave form of system utility commonly seen in NUM, 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 a 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 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 [3], 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'. However, collecting data from all the nodes may incur high communication overhead such as large delays or high energy consumption. On the other hand, due to the redundancy in the data, information from a subset of nodes may serve the applications' purposes.

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, 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 [5]. In both cases, data packets are forwarded towards the sink in a multi-hop way subject to the 1-hop interference model. 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 can be viewed as a discrete Network Utility Maximization (NUM) problem without exogenous packet arrivals. In the classic formulation of NUM for a wireless network [9], 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 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.

The key difference in our problem and the traditional NUM is with the choice of the utility functions. A NUM problem mainly focuses on throughput optimality, and by choosing different forms of U_s, various types of fairness can also be provided. These objectives, however, are not necessarily appropriate for the data gathering applications in sensor networks. Furthermore, for binary decisions, separable form of system utility commonly considered in NUM 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. *However, such an additive utility largely ignores the spatial correlation of sensed data.* In addition, for binary decisions, it is more natural to define the system utility as a set function, i.e., a function over the subsets of nodes. In this paper, we consider the general class of set functions that are monotone submodular, which includes the additive utility as a special case.

Submodularity, a discrete counterpart of concavity, captures the diminishing return property commonly seen in reality: the marginal improvement of the utility by 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 [8] and variance reduction criterion [7] for modeling the uncertainty of unsensed locations in an information based sensing model,

Z. Zheng is with the Department of ECE at The Ohio State University. N. B. Shroff is with the Department of ECE and CSE at The Ohio State University. Emails: {zhengz, shroff}@ece.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.

978-3-901882-45-6/2012 - Copyright is with IFIP

and the maximum a posteriori (MAP) estimate and a variant of the maximum likelihood (ML) estimate for parameter estimation [12].

In most previous works on sensor selection for maximizing a submodular utility, however, a simple cardinality constraint is considered, where the problem is already strongly NP-hard in general. Due to the multi-hop data forwarding nature, our problem for submodular maximization subject to a deadline constraint is much more challenging even under the simple 1-hop interference model. For additive utility, this problem can be solved efficiently for data aggregation using dynamic programming [11]. However, the technique does not apply to a general submodular utility. For more general interference models, the problem of aggregating all the data with minimum delay has been considered [14], where even the additive utility maximization problem remains open.

The main contributions of this paper are as follows.

- For data forwarding without aggregation, we show that a simple greedy technique that can be implemented efficiently achieves a factor 1/2-approximation when the sink has a single child, and a 1/3-approximation for a general routing tree.
- 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 bounded by a small constant (< 2) in practice, which is confirmed in simulations.

The rest of the paper is organized as follows. We present the system model and the problem definition in Section II. Our solutions to the deadline constrained data forwarding and data aggregation problems are presented in Sections III and IV, respectively. Simulation results are presented in Section V. We conclude the paper in Section VI.

II. 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 edges. 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 edges in T. We say that a node is at level k if the path connecting the node to s_0 has k edges. The level of s_0 is 0. A summary of the notations used throughout the paper is given in Table I.

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. Two data collection patterns are considered: *data forwarding* and *data aggregation*. For data forwarding, raw data packets are forwarded along the routing tree to s_0 without manipulation within the network. All the packets are assumed to be of the same size. In the case of data aggregation, an 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.

TABLE I: Notation List

T	Data collection tree
V_T	Set of sensor nodes in T
h_T	Height of T
Δ_T	Maximum number of children of nodes in T
$T(S)$	Minimum subtree of T that spans nodes $s_0 \cup S$
D	Deadline constraint
$L_F(S)$	Minimum delay of forwarding data on nodes $S \subseteq V_T$
$L_A(S)$	Minimum delay of aggregating data on nodes $S \subseteq V_T$
$L_A(T)$	$L_A(V_T)$
f	Monotone submodular utility over subsets of V_T

(a) data forwarding (b) data aggregation

Fig. 1: Minimum delay schedules for a tree with three internal nodes (dot) and a sink (square). Each node has a single packet. The number(s) besides a link denotes the time slot(s) when the link is scheduled to forward a packet.

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 gives examples of minimum delay schedules for data forwarding and data aggregation in a small tree.

For a subset $S \subseteq V_T$, let $f(S)$ denote the utility associated with the data sensed by the nodes in S. We assume that 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 $a \in V_T \backslash R$. Our solutions apply to any f that satisfies these conditions. The optimization problem that we study in this paper is:

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

Remark: 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 [13].

III. Deadline Constrained Data Forwarding

In this section, we study Problem 1 for raw data collection. We approach the problem using a standard greedy technique

(Section III-A). The *main challenge* is to show that the algorithm can be implemented efficiently (Section III-B), and achieves a small approximation ratio (Section III-C).

A. Greedy sensor selection

The simple greedy algorithm for Problem 1 is sketched in Algorithm 1. The algorithm starts with an empty set. In each step, a new node that maximizes the marginal improvement subject to the deadline constraint is added. This process repeats until no such node can be found.

Algorithm 1 Utility maximization for data forwarding under a deadline constraint

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

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

We note that the greedy algorithm requires access to two oracles, a value oracle that returns $f(S)$ given $S \subseteq V_T$ as input, and a membership oracle that decides whether $L_F(S) \leq D$, i.e., checking the feasibility of $S \subseteq V_T$. 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 III-C. In the following, we first present an efficient approach for checking feasibility, and then prove that the greedy solution closely approximates the optimal solution.

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, and 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 [4]. However, there is an error in a key construction in [4]. In the following, we provide a brief overview of the results in [4] 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 converse problem of data dissemination is considered in [4], where packets flow from s_0 to the internal nodes subject to the 1-hop interference constraint, which can then be converted back to a schedule of equal length for data forwarding. The main idea of [4] is the observation that an optimal schedule for a tree network can be derived by considering an equivalent multi-line network obtained by mapping each subtree rooted at s_0 to a line, where all the packets at level l of the subtree stay at the level l of the line. Hence we only need to consider the cases of a single line network and a multi-line network. A key construction in the multi-line case needs to be fixed for deriving the correct minimum delay schedules. 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 case when T is a single line network. The following schedule is shown to be optimal in [4]: s_0 first sends packets belonging to the furthest node, 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, i.e., the packets belonging to the level i node. Let $\mathbf{v} = (v_1, v_2, ..., v_m)$, where m is the maximum packet level. The minimum delay for delivering $\mathbf{v}$, denoted as $L_F(\mathbf{v})$, satisfies the following equation:

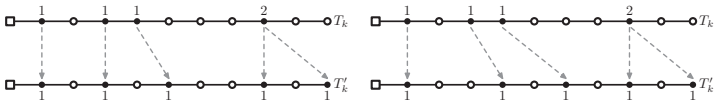
$$L_F(\mathbf{v}) = \begin{cases} \max\limits_{1 \leq i \leq m-1} (i - 1 + v_i + 2 \sum\limits_{j=i+1}^{m} v_j) & \text{if } m > 1, \\ v_1 & \text{if } m \leq 1. \end{cases} \quad \text{(III.1)}$$

Next assume that T is composed of K lines rooted at s_0. We will construct an equivalent tree T' also having K lines, where the packets on the k-th line of T are redistributed on the k-th line of T'. Our construction is different from that in [4] and is stated as follows. 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 [4]. Let v_i^k and $v_i'^k$ denote the number of level-i packets on the k-th lines of T and T', respectively. T' is constructed such that (1) $v_1'^k = v_1^k$, (2) $v_{t_j^k}'^k = 1, 1 \leq j \leq n_k$, and (3) $v_i'^k = 0$ for other i's. That is, the packets at level-1 stay where they 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 is equivalent to T and has some nice properties, which can be formalized to show the following result.

Proposition III.1. *(1) An optimal schedule for T can be converted to a schedule of the 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.*

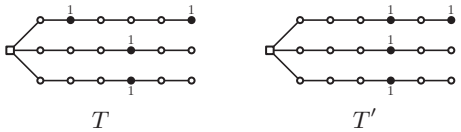
Proof: Properties (2) and (3) are clear from the construction. For the equivalence part, consider a data distribution schedule $\mathcal{S}$ for T of the following form: In each time slot, s_0 decides which level-1 node to forward a packet to among the 'legal' level-1 nodes. A level-1 node is 'legal' if it has no packet to forward in the current time slot. All the packets for the same branch 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 for the same line using the optimal single line schedule. Hence the k-th branches 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. ■

By making use of properties (2) and (3), an optimal schedule



(a) Our construction of T'. (b) Construction of T' in [4].

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



T T'

Fig. 3: T' constructed as in [4] is not equivalent 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'.

for T' can be easily established as in [4], which can then be converted back to an optimal schedule for T by property (1). Let $\mathbf{v}' = \{v_i'^k\}_{i\in\{1,...,m\},k\in\{1,...,K\}}$ denote the distribution of packets on T', where m denotes the maximum packet level in $\mathbf{v}'$. Let $\mathbf{v}'_j = \sum_{k=1}^{K} v_j'^k$. The minimum delay for forwarding packets in $\mathbf{v}$ to s_0 satisfies the following equation.

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

Remark: Packets are distributed in a different way in [4]: Packets at level-1 stay where they are, each packet p_i at level 2 or more is moved to level $t^k_{n_k} - 2(n_k - i), i = 1, ..., n_k$ [1]. See Figure 2(b) for an example. The T' built in this way still satisfies the properties (2) and (3). However, T' and T are not necessarily equivalent in this case. See Figure 3 for a counter-example.

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.

Definition III.1. *Given a 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 $S' \in \mathcal{I}$. For an independence system $(A, \mathcal{I})$ and a subset $S \subseteq A$, 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 $S' \cup \{a\} \in \mathcal{I}\}$. Then $(A, \mathcal{I})$ is a p-***system** *if for all $S \subseteq A$, $\frac{\max_{S'\in\mathcal{B}(S)} |S'|}{\min_{S''\in\mathcal{B}(S)} |S''|} \le p$.*

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 independent system. Furthermore, for any subset of edges $E' \subseteq E$, $\mathcal{B}(E')$ 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.

For an independent system $(A, \mathcal{I})$, consider a utility function f defined over the subsets of A. Then for the problem of maximizing $f(S)$ subject to $S \in \mathcal{I}$, a greedy algorithm similar to Algorithm 1 can be applied, where in each step, a new element with maximum marginal utility subject to the constraint is added. The following result for p-systems is classic [10]:

Lemma III.1. *For maximizing a monotone submodular utility f subject to a p-system constraint, and $f(\emptyset) = 0$, the greedy algorithm achieves a factor $1/(p + 1)$ approximation.*

[1]The formula given in [4] has a different but equivalent form.

A slightly more general result is proved in [1], where it is shown that a factor $\alpha/(p + \alpha)$ 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. Using the above lemma, we can establish the following performance bounds for Algorithm 1.

Proposition III.2. *Algorithm 1 is a 1/2-approximation to Problem 1 for data forwarding if the sink s_0 has only one child in T, and a 1/3-approximation for a general T.*

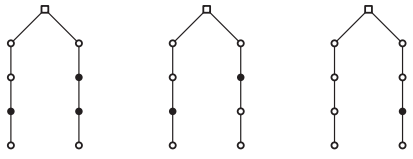
Proof sketch: Let $\mathcal{I}_F$ denote the set of feasible solutions to Problem 1 for data forwarding, i.e., $\mathcal{I}_F = \{S \subseteq V_T : L_F(S) \le D\}$. It is clear that $(V_T, \mathcal{I}_F)$ is an independent system. We show that $(V_T, \mathcal{I}_F)$ is a 1-system when T has a single level-1 node, and is a 2-system in general. The proposition then follows from Lemma III.1. By converting each subtree rooted at s_0 to an equivalent line as in [4], 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 of the more difficult multi-line case relies on our construction of T', Proposition III.1, and Equation III.2. The details are provided in the Appendix. ■

Figure 4 gives an example where a tree with 3 packets forms a 2-system.

Remark: For a given deadline constraint D, it is clear that the maximum achievable utility depends on the tree topology. The above proposition 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.

IV. 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. We note that a greedy algorithm similar to Algorithm 1 can be applied to this case as well. However, proving a performance guarantee for it directly eludes us. The main difficulty lies in the obscure structure of the feasible solutions induced from the 1-hop interference model subject to the deadline constraint. In the following, we provide a new algorithm, which has a bi-criteria approximation to the problem. The main idea is to reduce (approximately) the



(a) (b) (c)

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.

original problem to the easier problem of data aggregation under the 'clique' interference model.

We first observe that for data aggregation, by the monotonicity of the utility function f, if a node a is selected in an optimal solution, then including all the nodes along the unique path from a to the sink is still optimal. Hence it suffices to consider the set of subtrees of T rooted at the sink subject to the deadline constraint. 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 IV-A), which will be used in the analysis of our algorithm. We then consider the data aggregation problem under the 'clique' interference model and propose a greedy solution to it (Section IV-B), which is then used as a subroutine in deriving the bi-criteria approximation to Problem 1 (Section IV-C).

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 IV.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_T packet, which takes at least h_T time slots to reach the sink. To see the upper bound, 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 $h = \min\{h_T, D\}$ 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, which has a minimum delay bounded by $h\Delta_T$ by the 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 in any time slot, 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 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 spans s_0 and 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., $\mathcal{I}^\infty = \{S \subseteq V_T : |T(S)| \leq D\}$. Then $(V_T, \mathcal{I}^\infty)$ is an independence system. We further have the following result:

Proposition IV.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 further 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 $(V_T, \mathcal{I}^\infty)$ is a h_T-system but not a h_T-1 system. By the proposition, a greedy algorithm similar to Algorithm 1 achieves a factor $\frac{1}{h_T+1}$-approximation to Problem 2 by Lemma III.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 from a to $T(S)$.

C. A bi-criteria approximation to Problem 1

We now propose the following algorithm to Problem 1 for data aggregation. All the subtrees in the algorithm are rooted at s_0. First, T is truncated to $T^{(0)}$ to include only nodes at level $h = \min(h_T, D)$ or less (line 2), 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

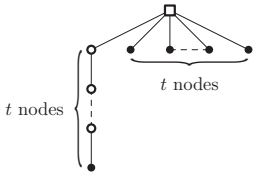


Fig. 5: Consider a subset S consisting of the $t+1$ black nodes in the tree, and assume $D = h_{T(S)} = t$. Then the bottom left node forms a maximal independent set in S, so do the t level-1 nodes.

then found (line 3), which can be implemented using the dynamic programming algorithm for additive utility in [11] by associating with each node a unit weight. Line 4 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 $L_A(T^{(2)})$ could be larger than D. $T^{(2)}$ is then expanded without further increasing the delay (line 5). This can be done in various ways without effecting 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 respecting a subtree needs to be determined. We show that this can be done efficiently at the end of this subsection.

Algorithm 2 Utility maximization for data aggregation under a deadline constraint

Input: T, f, L_A, D; Output: $T^{(3)}$: a subtree of T

1: $h \leftarrow \min(h_T, D)$.
2: $T^{(0)} \leftarrow$ the subtree of T including all the nodes at level h or less.
3: $T^{(1)} \leftarrow$ a maximum cardinality subtree of $T^{(0)}$ with $L_A(T^{(1)}) \leq D$.
4: $T^{(2)} \leftarrow$ a maximum utility subtree of $T^{(0)}$ with $|T^{(2)}| \leq |T^{(1)}|$.
5: $T^{(3)} \leftarrow$ a maximal subtree of T obtained by greedily expanding $T^{(2)}$ such that $L_A(T^{(3)}) \leq \max(L_A(T^{(2)}), D)$.
6: **return** $T^{(3)}$.

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

Proposition IV.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 $T^{(3)}$ found by Algorithm 2, we have $f(T^{(3)}) \geq \frac{1}{\min(h(T),D)+1} OPT$, and $L_A(T^{(3)}) \leq \rho_T D$, where $\rho_T := \max_{T',T'' \in \mathbb{T}} \left\{ \frac{L_A(T')}{L_A(T'')} : h_{T'} \leq h_{T''}, |T'| \leq |T''| \right\}$ and is bounded by Δ_T.*

Proof: We first study the utility of $T^{(3)}$ 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 $|T^{opt}| \leq |T^{(1)}|$, $f(T^{opt}) \leq f(T^{opt(2)})$ by the optimality of $T^{opt(2)}$. Hence $f(T^{(3)}) \geq f(T^{(2)}) \geq \frac{f(T^{opt(2)})}{h_{T^{(2)}}+1} \geq \frac{f(T^{opt})}{h+1} = \frac{OPT}{\min(h(T),D)+1}$.

We then bound $L_A(T^{(3)})$. It is not hard to see that $T^{(1)}$ has one and only one level-h node. Hence $h_{T^{(1)}} = h$. Since $h_{T^{(2)}} \leq h$, and $|T^{(2)}| \leq |T^{(1)}|$, we have $L_A(T^{(3)}) \leq \max(L_A(T^{(2)}), D) \leq \max(\rho(T) L_A(T^{(1)}), D) \leq \rho(T) D$. Furthermore, for any $T', T'' \in \mathbb{T}$ such that $h_{T'} \leq h_{T''}$ and $|T'| \leq |T''|$, we have $L_A(T') \leq h_{T'} \Delta_{T'} \leq h_{T''} \Delta_T$ and $L_A(T'') \geq h_{T''}$ by Proposition IV.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 in average and by 3 in the worst case. The detailed simulation results are given in Section V.

Minimum delay data aggregation: We now provide an efficient algorithm to find $L_A(T)$, which can be used to implement the last step of Algorithm 2 in a greedy way, and is also interesting by itself. We note that the algorithm in [11] 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 $v_1, v_2, ..., v_K$ denote the children of v, and suppose $L'_A(T_{v_1})$ through $L'_A(T_{v_K})$ have been found. We then create a K-line network rooted at v, with a single packet at level $L'_A(T_{v_k}) + 1$ on the k-th line, for $k = 1, ..., K$. $L'_A(T_v)$ can then be determined by Equation (III.2). This algorithm takes $O(n \Delta_T L_A(T))$ time.

D. Discussion

The data aggregation problem under the 'clique' interference model can be viewed as a group Steiner problem with a general submodular coverage function and unit edge weight. Given an edge-weighted graph where every node covers a subset of groups from a ground set, the group Steiner problem is to find a subset of nodes that cover the maximum number of groups subject to a bound on the weight of the Steiner tree spanning the nodes. For general edge cost, it is known that even for a tree graph, there is no $\frac{1}{log^{1-\epsilon} m}$-approximation for any fixed $\epsilon > 0$ for the group Steiner problem, where m is the total number of groups, unless $NP \subseteq ZTIME(n^{\text{polylog}(n)})$ [6]. Furthermore, no combinatorial algorithm is known to achieve this bound.

We conjecture that our problem under 1-hop interference model is harder than the problem under the clique model, and achieving polylogarithmic approximation is hard. We remark that the recursive greedy algorithm in [2] 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 $h_T = O(\log n)$.

V. Simulations

In this section, we study the performance of our solutions using simulations. We first show the advantage of Algorithm 1 and Algorithm 2 compared with a random sensor selection algorithm, by considering a 2-d sensor network and a target detection utility function. We then show that ρ_T is typically a small constant by doing a random sampling of subtrees.

A. Performance of Algorithm 1 and Algorithm 2

We consider a 1000×1000 2-d region, which 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

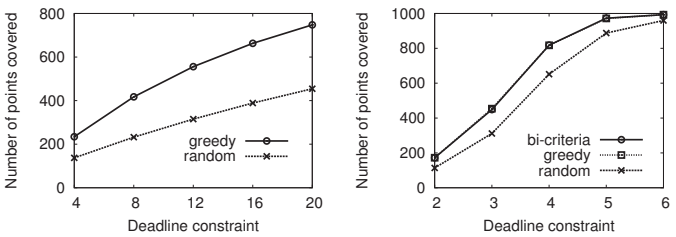
region such that the probability that a point is within certain cell is proportional to the weight of the cell, and within a cell, points are uniformly distributed. 200 sensor nodes are uniformly distributed in the entire region independently. Each node has a sensing range of 100, and a communication range of 200. A target point is detected by a node if the point is within the sensing range of the node. There is a link between two nodes if they are within the communication range of each other. The utility for a set of nodes is defined as the number of target points detected by nodes in the set, which is a (non-additive) monotone submodular function.

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. For data forwarding, Algorithm 1 is then applied to T for various deadline constraints. We compare our algorithm with a random sensor selection algorithm, where nodes are added randomly subject to the deadline constraint on the set of nodes selected. The random algorithm is repeated 100 times for each T. A similar random selection algorithm is also applied to the case of data aggregation to compare with Algorithm 2. In this case, for fair comparison, the deadline constraint to the random algorithm equals to the minimum delay of the subtree found by our algorithm. We further implement a greedy algorithm for data aggregation, which is similar to Algorithm 1, and compare it with Algorithm 2.

Figure 6(a) shows the utility (number of target points detected) achieved by the two algorithms for data forwarding, where the results are averaged over the random distributions of target points, sensor nodes, and the sinks. Algorithm 1 achieves about 70% higher utility than the random selection algorithm for all the deadline constrains. Figure 6(b) shows the results in the case of data aggregation. Compared with the random selection algorithm, Algorithm 2 achieves about 50% higher utility for small deadlines and more than 25% higher utility in average. We note that the computational time of our algorithm and that of the random selection algorithm are comparable in this setting. This is because the random selection algorithm also needs to maintain a partial solution and to check the feasibility when adding a new node, which dominates its running time. For a more complicated utility function other than the simple set cover, our algorithm could be more time consuming due to the multiple evaluations of the utility function. But in that case, we expect our algorithm to achieve an even higher utility than the random selection algorithm that completely ignores the utility function. We further observe that the performance of Algorithm 2 and that of the greedy algorithm are very close in the evaluated scenarios, which might imply that the bi-criteria bound can be translated into an equivalent bound for the greedy algorithm. Also note that the two algorithms have comparable running time.

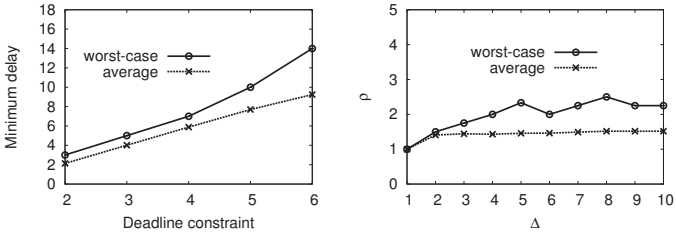
B. The distribution of ρ_T

We have shown that ρ_T is upper bounded by Δ_T in Section IV-C, which we expect to be only a loose bound. For instance, the minimum delay of the subtrees found by



(a) Data forwarding (b) Data aggregation

Fig. 6: Number of points covered under various deadline constraints. In (b), the real minimum delay of the solutions may be larger than the deadline constraints, but are the same for all the algorithms.



(a) (b)

Fig. 7: (a) The average and worse-case minimum delay for the subtrees found by Algorithm 2 under various deadline constraints for trees built on the 200-node sensor network. (b) The average and worst-case values of ρ_T of a randomly sampled tree with degree bounded by $\Delta = 1, ..., 10$ and height bounded by 6.

Algorithm 2 for various deadline constraints in the setting described above is plotted in Figure 7(a). We observe that for a given deadline D, the minimum delay of the trees is less than $2.5D$ in the worst case and is about $1.5D$ in average.

We then make some effort to study the typical values of ρ_T for 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 7(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 about 1.5.

VI. Conclusion

In this paper, we study the problem of sensor selection for maximizing the utility of data collected through a routing tree subject to the 1-hop interference model and a deadline constraint. We consider the general class of utility functions that are monotone submodular. 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 under the setting of in-network aggregation. As part of our future work, we plan to extend our solutions to a

more general setting by considering unreliable wireless links, imperfect data aggregation, and a more general interference model. We also plan to study efficient data collection schemes under other types of constraints, e.g., an energy constraint on each sensor node.

REFERENCES

[1] 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.

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

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

[4] 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.

[5] 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.

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

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

[8] 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.

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

[10] 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.

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

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

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

[14] 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 MOBIHOC*, 2009.

APPENDIX
PROOF OF PROPOSITION III.2

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 II. 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 III-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 will show that there is a packet in Y_t^k that can be added to X, subject to the deadline constraint. We 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 $i > t$.

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 $i - 1 + \sum_{k,j \geq i} \bar{x}_j'^k \leq i - 1 + 1 + \sum_{k,j} x_j'^k \leq s - 2 + \sum_{k,j \geq s-1} y_j'^k \leq L_F(Y)$, 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. ■