# Maximizing Aggregated Information in Sensor Networks under Deadline Constraints

Srikanth Hariharan* and Ness B. Shroff

***Abstract*—We study the problem of maximizing the aggregated information in sensor networks with deadline constraints. Our model is that of a sensor network that is arranged in the form of a tree topology, where the root corresponds to the sink node, and the rest of the network detects an event and transmits data to the sink over one or more hops. We assume a time-slotted synchronized system and a node-exclusive (also called a primary) interference model. We formulate this problem as an integer optimization problem and show that for unit capacity links, the optimal solution involves solving a Bipartite Maximum Weighted Matching problem at each hop. We propose a polynomial time algorithm that uses only local information at each hop to obtain the optimal solution. Thus, we answer the question of when a node should stop waiting to aggregate data from its predecessors and start transmitting in order to maximize aggregated information within a deadline imposed by the sink. We extend our model to allow for practical considerations such as arbitrary link capacities, and also for multiple overlapping events. Further, we show that our framework is general enough that it can be extended to a number of interesting cases such as incorporating sleep-wake scheduling, minimizing aggregate sensing error, etc.**

## I. Introduction

A wireless sensor network is a wireless network consisting of a number of sensors that are distributed in a region in order to cooperatively monitor certain physical or environmental conditions. These networks are used in a number of civilian and military applications, such as environment and habitat monitoring, battlefield surveillance, and traffic control. Due to size and cost constraints, sensor nodes have limited energy, processing, memory, and bandwidth capabilities. Typically, these nodes sense a desired aspect of the region in which they are deployed and occasionally report the sensed data to those sinks that have subscribed for that data. The sensed data is prone to error due to resource constraints and environmental factors. Therefore, sinks cannot rely on the data sensed by a single sensor. Moreover, since there is usually a certain degree of redundancy in the data sensed by different sensors, in many applications, the sinks only desire an aggregated form of the data sensed by different sensor nodes. Examples include finding the average temperature in a region, determining whether pressure in a region is below a certain value, and determining the average location and velocity of a target. It is known that when sinks require an aggregated form of the sensed data, performing in-network computation greatly reduces the communication overhead [2].

One of the key issues in data aggregation in sensor networks is the trade-off between energy, delay, and the quality of data obtained by the sink. This can be viewed as follows. Consider a data aggregation tree in which each parent aggregates data from all of its children before forwarding it to the next hop. Assuming error-free links and no collisions, each parent then needs to make at most one transmission. However, each parent will have to wait until it has received data from all its children. On the other hand, if each parent decides to transmit every time it receives a packet from one of its children, then this may result in excessive transmissions, defeating the very purpose of data aggregation. Therefore, in order to maximize the data quality at the sink under deadline or energy constraints, one needs to carefully control the number of transmissions a node can make and the time that a node can wait to gather the data.

Thus, there is a delay-energy trade-off that needs to be carefully considered depending on the level of delay an application can tolerate. Moreover, the quality of data reported at the sink is also important. In particular, we are interested in finding the maximum amount of aggregated information that can be obtained by a sink under deadline and energy constraints. Our goal in this paper is to maximize the aggregated information in a given data aggregation tree with the sink as the root. "Aggregated information" can be thought of in a number of ways. For example, if each packet has a priority associated with it, we can maximize the sum of the priorities of packets accounted for at the sink. If each node senses data with a certain accuracy, we can think of maximizing the accuracy of the aggregated data at the sink. For simplicity of presentation, we will first define *aggregated information* as the number of nodes whose packets have been accounted for at the sink within the imposed deadline. It is important to note that this definition is also motivated by practical considerations. For instance, suppose that a set of sensor nodes desire to obtain the average temperature in a region. Assuming that the nodes make independent observations, and that each of their observations has the same error variance ($\sigma^2$), the estimate of the average temperature will then have an error variance of $\frac{\sigma^2}{n}$, where $n$ is the number of source nodes whose data has been used to estimate the average. Thus, the greater the number of source nodes, the better the estimate.

This work was supported in part by ARO MURI Awards W911NF-07-10376 (SA08-03) and W911NF-08-1-0238, and NSF Awards 0626703-CNS, 0635202-CCF, and 0721236-CNS.

S. Hariharan is with the Department of Electrical and Computer Engineering, The Ohio State University, 2015 Neil Ave., Columbus, OH 43210, USA harihars@ece.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@ece.osu.edu

* Corresponding author.

A preliminary version of this paper by S. Hariharan and N. B. Shroff titled "Maximizing Aggregate Revenue in Sensor Networks Under Deadline Constraints" appeared in the proceedings of the IEEE Conference on Decision and Control, 2009 [1].

We now briefly examine the related work in this area. Currently, there exist a number of techniques addressing the formation of data aggregation trees [3]–[5]. In all these works, constructing an optimal data aggregation tree has been shown to be NP-Hard for a number of cases such as lifetime maximization, minimizing the total number of transmissions etc. These works then develop heuristic algorithms to construct trees that are provably efficient. A second category of works study trade-offs between energy, data accuracy, and delay for data aggregation in wireless sensor networks. Boulis et al., [6] study trade-offs between energy and data accuracy in data aggregation trees. In [7], Yu et al., study trade-offs between energy and latency in data aggregation trees assuming a time-slotted synchronized system. As mentioned in [7], enforcing the latency constraint requires time-synchronization schemes such as [8]. Bechchetti et al., [9] study the problem of minimizing energy in the presence of latency constraints for data aggregation. In [10], Ye et al., study the fundamental energy-delay trade-off for distributed data aggregation in wireless sensor networks. Their goal is to maximize a certain parameter called the "discounted reward" at each node, where the reward is due to data aggregation and the discount is due to the time for which the node waits in order to aggregate data from its predecessors. They propose two learning-based distributed approximation algorithms that empirically perform close to the optimal solution. The common drawback of these works is that interference is not a part of the optimization framework even though it is a critical component of the wireless environment. Lai et al. [11] study an interesting problem of minimizing a weighted sum of the expected energy consumed and an exponent of the latency probability. Compared to hard deadlines studied in this work, they consider the probability that the latency exceeds a threshold.

In this work, we propose an optimization framework that can be used to study trade-offs between aggregated information, energy and latency, in a time-slotted system, under a one-hop interference model.

The main contributions of this work are as follows.

- We develop an optimization framework for maximizing the aggregated information that is accounted for at the sink from all source nodes in a given data gathering tree, within a deadline. This optimization framework explicitly account for interference.
- We provide a distributed optimal solution that consists of two components - an optimal data aggregation policy, and an optimal scheduling policy. For unit capacity links, we show that the scheduling problem can be reduced to a well-known Maximum Weighted Matching problem (MWM). *The key fact here is that this Maximum Weighted Matching is only within a single hop (because of the data aggregation policy)*, and not the entire network, and thus *it has a significantly lower complexity than the traditional Maximum Weighted Matching problem in scheduling*.
- We extend our solution to account for arbitrary link capacities, and also for multiple overlapping events.

The rest of this paper is organized as follows. In Section 2, we list our assumptions and describe the system model. In Section 3, we formulate the problem for data aggregation trees as an integer optimization problem. In Section 4, we describe the optimal solution for unit capacity links. In Section 5, we analyze the complexity of this solution. In Section 6, we discuss various interpretations and applications of our problem. In Section 7, we extend our solution to account for arbitrary link capacities, and multiple overlapping events. Finally, in Section 8, we conclude the paper.

## II. System Model and Assumptions

We model the system as a graph $G(V, E)$ where $V$ is the set of nodes and $E$ is the set of links. The system has $N$ nodes and a sink. When an event occurs, nodes sense some desired quantity and send an aggregated form of the data to the sink. $G$ is a tree rooted at the sink. A node may or may not be a source for a particular event. We assume that the system is time-slotted and synchronized. During each time slot, a node can perform only one of the following: sending a packet, receiving a packet, or remaining idle. The sink imposes an event-dependent deadline within which it should receive data from the sensor nodes. We assume that events are static events in which each source node associated with that event knows that some parameter needs to be sensed at a particular time. For example, sensor nodes could periodically sense temperature in a region and report some aggregated form of the data to the sink.

The aggregation function we consider, can be any divisible function [12]. Divisible functions are those that can be computed in a divide and conquer fashion. For example, assume that the sink desires the function $f(x_1, x_2, ..., x_N)$, where $x_1$, $x_2$, ..., $x_N$ are the raw data measurements of the $N$ sensor nodes. Let $S$ denote the set $\{x_1, x_2, ..., x_N\}$ and let $f(S)$ denote $f(x_1, x_2, ..., x_N)$. The function $f$ is divisible if, given any partition, $P(S) = \{S_1, ..., S_j\}$, of S, there exists a function $g^{P(S)}$, such that $f(S) = g^{P(S)}(f(S_1), f(S_2), ..., f(S_j))$ for any $x_1, ..., x_n$. The complete definition of divisible functions can be found in [12]. Examples of divisible functions include MIN, MAX, Sum, Median, Mode, etc. Consider the MAX function for instance. Suppose the sink desires $MAX(1, 2, 3, 4, 5)$. Then, given a partition, (say) $\{\{1, 2\}, \{3, 4\}, \{5\}\}$ of $\{1, 2, 3, 4, 5\}$, $MAX(1, 2, 3, 4, 5) = MAX(MAX(1, 2), MAX(3, 4), MAX(5))$. We illustrate this example in Figure 1. In Figure 1, the source nodes (denoted by filled circles) hold the measurements 1,2,3,4, and 5. The intermediate nodes send $MAX(1, 2)$, and $MAX(3, 4)$ rather than sending 1,2,3,4 separately. As commonly used in the literature, we refer to this computation performed by intermediate nodes as *in-network computation*. The sink can finally still compute $MAX(1, 2, 3, 4, 5)$ because MAX is a divisible function.

While Average is not a divisible function, it can also be computed if the total number of source nodes is known (or transmitted) at the sink.

We consider the one-hop or node exclusive interference model. In the one-hop interference model, any two links that share a node cannot be active at the same time, which captures one of the key attributes of practical wireless transmission.

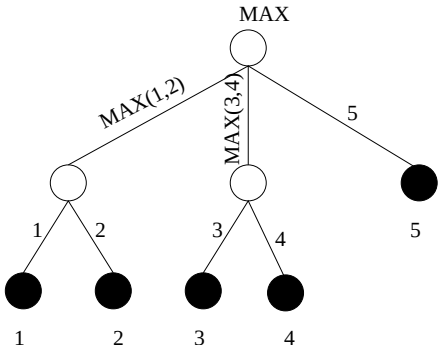


Fig. 1: In-network Computation of MAX

**Relaxable Assumptions**

In order to illustrate our problem and solution, we make the following additional assumptions. We later explain how our solution can be modified when these assumptions are relaxed.

We assume that the aggregation delay (the time required to aggregate data from different sensors) is negligible. We assume that the capacity of each link is fixed and equal to 1. We assume that packet sizes are equal and that it takes one time slot to transmit one packet. For each event, we assume that all the source nodes that are associated with the event, sense this event and are ready to transmit their observation at time zero. We assume that the next event occurs only after the deadline for the current event expires.

## III. Problem Formulation

We now consider a network, modeled as a tree with the sink being the root, and formulate the problem of maximizing the aggregated information at the sink when the sink imposes a deadline. As mentioned earlier, we define aggregated information at the sink as the number of source nodes whose data has been accounted for at the sink.

We next provide some notations and definitions.

- $V$ - Set of $N$ sensor nodes and a sink, $S$.
- $E$ - Set of edges.
- $P(i)$ - Parent of node $i$.
- $PATH(i)$ - The set consisting of $i$ and all its ancestors in its path to the sink, but not including the sink, i.e., $PATH(i) = \{i, P(i), P(P(i)), ...\}$, and $S \notin PATH(i)$.
- $T_i$ - Denotes whether node $i$ is a source for a particular event, i.e., it denotes whether node $i$ has its own packet to send for a particular event. Hence,

$$T_i = \begin{cases} 1, & \text{node } i \text{ has its own packet to send} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

- $n_i$ - An indicator variable representing whether child $i$ is allowed to transmit to its parent, i.e., $n_i = 1$ if child $i$ is allowed to transmit to its parent $j$, and 0 otherwise..
- $W_i$ - The time that node $i$ waits to aggregate packets from its predecessors for a particular event. After $W_i$ time units, node $i$ will no longer accept packets from its predecessors. Also, until $W_i$ expires, node $i$ will not transmit any aggregated packet to its parent. As mentioned before, we assume that the event is sensed by each source node at time zero.
- $V_L$ - Set of all leaf nodes in the tree.
- $D$ - The deadline by which packets must reach the sink.

Now the optimization problem can be framed as follows. We call this problem $Y$.

**Problem *Y*:**

$$\max_{\vec{n},\vec{W}} \quad \sum_{i \in V} T_i \prod_{j: j \in PATH(i)} n_j \quad (2)$$

$$\text{s.t.} \quad \text{For each } i \in V \backslash V_L\text{: } \forall C \subseteq \{(j,i) : (j,i) \in E\},$$

$$\sum_{j:(j,i) \in C} n_j \leq W_i - \min_{j:(j,i) \in C} W_j \quad (3)$$

$$n_i \in \{0,1\} \ \forall (i,j) \in E \quad (4)$$

$$W_i \in \{0,1,...,D-1\} \ \forall i \in V \backslash \{S\}, W_S = D \quad (5)$$

The goal of problem $Y$ is to determine the control variables, $n_i$, for each link $(i,j) \in E$, and $W_i$, for each node $i \in V \backslash \{S\}$, such that the aggregated information is maximized at the sink.

Now, before we present the optimal solution to $Y$, it is illustrative to understand the constraints in $Y$. The constraints in (4) and (5) are straightforward to interpret. The constraint in (3) explains the relationship between interference and delay, in data aggregation trees. Under the one-hop interference model, a parent node can only receive packets from one of its children nodes during a particular slot. However, when a child node transmits to its parent, the other children (of the same parent) can receive data from their children (by the definition of the one-hop interference model). For example, consider Figure 2(a) with node $P$ receiving data from its children $C_1$, $C_2$ and $C_3$. This figure represents a single hop in a large data aggregation tree. During a slot in which $C_1$ transmits to $P$, $C_2$ and $C_3$ can receive data from their children. However, no two children of $P$ can transmit to $P$ in the same slot. Let node $P$ have a waiting time $W$, and $C_1$, $C_2$ and $C_3$ have waiting times $W_1$, $W_2$ and $W_3$, respectively. Let $W_1 < W_2 < W_3 < W$. Then, the total number of transmissions that can be made from all the children nodes to the parent $P$ is limited by the difference between $W$ and $W_1$ (since the first transmission can occur only after $W_1$ and the last transmission can occur only before $W$, by the definition of waiting time). Also, the total number of transmissions that can be made from $C_2$ and $C_3$ to $P$ is limited by $W - W_2$. So, (3) says that for any subset of children nodes, the total number of transmissions made by this subset of nodes is bounded above by the difference between the waiting time of the parent and the waiting time of the child that has the least waiting time in the chosen subset. We will use these observations when we develop an optimal algorithm to solve $Y$.

Note that we have made two important assumptions while formulating the problem (which will be shown to not affect the optimal solution).

1) The data aggregation policy does not allow any node to transmit more than once.
2) In our data aggregation policy, a node cannot accept packets once its waiting time expires. For example, consider a node $i$ with waiting time $W_i$. Consider the sub-tree rooted at node $i$. Then $W_i$ serves as a deadline

by which source nodes in the sub-tree rooted at node $i$ should send their packets to $i$.

**Note:** We have made these assumptions only in order to assist in formulating the optimization problem. It can be readily shown that these assumptions do not affect the optimal solution of the general problem, $Z$, that is defined below.

**Problem *Z*:**

Consider an optimization problem, $Z$, whose objective is to maximize the number of source nodes accounted for at the sink, within a deadline imposed by the sink, under the one-hop interference model. Let $Z$ allow for multiple transmissions and also allow a node to accept packets irrespective of the current time.

**Theorem III.1.** *Any optimal solution to problem $Y$ is also an optimal solution to problem $Z$.*

*Proof:* Suppose that in the optimal solution to problem $Z$, a node $Q$ makes $k$ "useful" transmissions. By "useful", we mean that the transmitted packet reaches the sink within the deadline $D$. Specifically, let node $Q$ transmit packet $p_i$ at slot $W_i$, $1 \leq i \leq k$ $(W_1 < W_2 < ... < W_k)$.

Since the packets $p_1, ..., p_k$ can be aggregated into a single packet, the same optimal solution could have been achieved if $Q$ had simply aggregated these packets and transmitted the aggregated packet at slot $W_k$ instead of making $k$ separate transmissions. However, this is equivalent to $Q$ having a deadline $W_k$ and transmitting it exactly once.

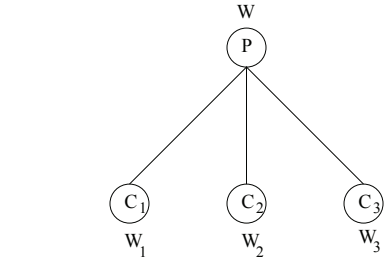
Thus, it is clear that any optimal solution to $Z$ can be transformed into a solution satisfying the constraints of $Y$, and with both solutions resulting in the same value of the objective function. Therefore, any optimal solution to problem $Y$ is also an optimal solution to problem $Z$. ■

From Theorem III.1, it is enough to solve Problem $Y$ in order to obtain an optimal solution to problem $Z$.

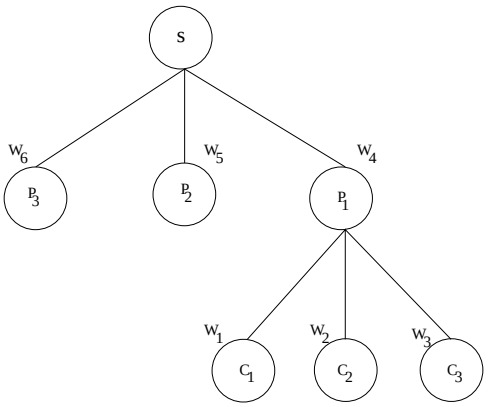
We now provide a few examples to explain how $Y$ can be solved in simple cases. Consider a tree network with six nodes, $C_1$, $C_2$, $C_3$, $P_1$, $P_2$, $P_3$, with corresponding waiting times $W_1$, $W_2$, $W_3$, $W_4$, $W_5$, $W_6$, and a sink $S$, as shown in Figure 2(b). Let all nodes other than the sink $S$ be sources for a particular event. If the deadline imposed by $S$ is 1, i.e., $D = 1$, then the maximum aggregated information that reaches the sink is one since in the given single time slot, only one of $P_1$, $P_2$ and $P_3$ can transmit to $S$ under the one-hop interference model). If $D = 2$, then the maximum aggregated information that reaches the sink is 3. In the first slot, one of $P_2$ and $P_3$ will transmit to $S$. In the same slot, one of $C_1$, $C_2$ and $C_3$ will transmit to $P_1$. In the second slot, $P_1$ will aggregate its own data with the data received from its child in the first slot and send the aggregated data to $S$. Thus, $P_1$ will account for two source nodes. Similarly it can be shown that if $D = 3$, then the maximum aggregated information that reaches the sink is 5.

## IV. Solution

In this section, we provide an algorithm (Algorithm $A$) that solves problem $Z$ and prove that this algorithm provides an optimal solution to $Z$. Note that when $D \geq N$, the problem is trivial, since irrespective of the order of transmission, all source nodes will be accounted for at the sink, by the deadline. So we will only consider the case when $D < N$.




(a) Parent $P$ waits until time slot $W$ to receive packets from children $C_1$, $C_2$ and $C_3$




(b) Problem $Y$ - Example

Fig. 2: Examples

**Algorithm *A*:**

1) Let $X[i, W]$ denote the maximum number of nodes that node $i$ can account for if its waiting time is $W$, $0 \leq W \leq D-1$. For every leaf node $l$ and for each $W$, $X[l, W] = T_l$.
2) Consider any node $j$ other than the sink and leaf nodes. Suppose it has $k$ children, $C_1, C_2, ..., C_k$. For every $W$, $0 \leq W \leq D-1$, calculate $X[j, W]$ as follows.
   a) If $W \geq k$, assign waiting times $W_1, W_2, ..., W_k$ to $C_1, C_2, ..., C_k$, respectively, such that each $W_i$, $1 \leq i \leq k$, takes a value in the set $\{W-1, W-2, ..., W-k\}$, where no two nodes can have the same waiting time, and such that the sum $\sum_{i=1}^{k} X[C_i, W_i]$ is maximized. This is a Maximum Weighted Matching (MWM) problem in the bipartite graph $(A, B, P, Q)$, where $A = \{C_1, C_2, ..., C_k\}$, $B = \{W-1, W-2, ..., W-k\}$, $P$ is the set of edges $\{(a, b) : a \in A, b \in B\}$, and $Q$ is the set of weights of each edge. An edge $(a, b) \in P$ has a weight $X[a, b]$.
   b) If $W < k$, assign waiting times from the set $\{W-1, W-2, ..., 0\}$ to $W$ out of the $k$ children such that no two children that have been assigned a waiting time from this set have the same waiting time,

and such that the sum $\sum_{i:C_i \text{ is assigned}} X[C_i, W_i]$ is maximized. This is an MWM problem in the bipartite graph $(A, B, P, Q)$, where $A = \{C_1, C_2, ..., C_k\}$, $B = \{W-1, W-2, ..., 0\}$, $P$ is the set of edges $\{(a,b) : a \in A, b \in B\}$, and $Q$ is the set of weights of each edge. An edge $(a,b) \in P$ has a weight $X[a,b]$.

3) Finally, at the sink, calculate $X[S,D]$ as illustrated in Step 2. Thus, we calculate $X[\cdot,\cdot]$ from the leaves to the root.
4) Knowing $D$ at the sink, assign waiting times to the sink's children based on how $X[S,D]$ is obtained. Proceed from root to leaves and assign waiting times.

Algorithm $A$ calculates $X[\cdot,\cdot]$ from the leaves to the root and assigns waiting times from the root to the leaves. Note that a general brute force approach for Step 2 would result in a complexity of $O(D^k)$. Identifying the Maximum Weighted Matching reduction, and also the set of possible waiting times, is key in reducing the complexity of the solution.

We now prove that this algorithm provides an optimal solution to problem $Y$, and hence, to problem $Z$. Before we do this, we first rewrite the objective function of Problem $Y$ in the following recursive manner.

It is clear that for a leaf node $l$, $X[l,W] = T_l$ for any waiting time $W$.

**Lemma IV.1.** *$X[S,D]$ provides the optimal solution to Problem $Y$, where for any non-leaf node $i$ with $k$ children ($C_1$,...,$C_k$), and for any waiting time $W$, $X[i,W]$ is calculated recursively by the equation*

$$X[i,W] = T_i + \max_{Q} \sum_{j=C_1}^{C_k} X[C_j, W_j] n_j, \qquad (6)$$

*where $Q$ is the set of constraints including constraint (3) for node $i$, $n_j \in \{0,1\}$, and $W_j \in \{0, ..., D-1\}$, for $j \in \{C_1, ..., C_k\}$.*

*Proof:* We show this result by induction.

Consider a node $i$ such that all its children are leaf nodes. Then the maximum number of sources that $i$ can account for if it waits for a time $W$ is simply its own data, and the maximum number of children of $i$ (that are sources) that can transmit their data to $i$ by time $W$. This is because $i$'s children are all leaf nodes, and therefore $X[C_j, W_j] = T_{C_j}$ for each $C_j$ and for each $W_j$. Therefore, (6) holds for nodes that have only leaf nodes as children.

Assume that the result is true for nodes that are at height $h$ from nodes that have only leaf nodes as children.

Consider a node $i$ at height $h+1$. We need to show (6) for this node. Clearly, node $i$ will account for its own data $T_i$ irrespective of the waiting time. Further, since the children of $i$ are at height $h$ from leaf nodes, $X[C_j, W_j]$ represents the maximum number of sources that $C_j$ can account for if its waiting time is $W_j$. Further, observe that for a given $W$, the constraints in Problem $Y$ that affect the calculation of $X[i,W]$ are only those mentioned in (6). Therefore, by maximizing over these constraints, we obtain the maximum of the sum of the aggregated information from the children of $i$. Hence, $X[i,W]$ is given by (6). ■

This lemma illustrates that one can obtain a distributed solution by solving for $X[\cdot,\cdot]$ at each hop.

**Lemma IV.2.** *For any node $i \in V$, $X[i,W]$ cannot decrease as $W$ increases.*

*Proof:* Clearly, if a node waits for a longer time to accumulate packets from its children, then it should be able to accumulate at least as many packets as it had accumulated when it had waited for a shorter period of time. ■

We now build the optimal solution from the leaves of the tree to the root. We will use induction to prove the optimality.

**Lemma IV.3.** *Consider a node $P$ whose children are all leaf nodes. Let there be $k$ children, $C_1$,...,$C_k$. For a given $W$, $X[P,W]$ is obtained by allocating waiting times, $W - \min(W,k)$,...,$W-1$, in the order of decreasing $T_{C_j}$ for $j = 1, 2, ..., k$ (i.e., by first allocating slots to nodes with $T_{C_j}$ = 1, and then allocating remaining slots to nodes with $T_{C_j}$ = 0).*

*Proof:* Clearly, if $P$ has a waiting time $W$, then at most $\min(W,k)$ children can transmit among the $k$ children within the deadline $W$ at $P$. The proof is now obvious since leaf nodes will account for one packet if they are sources and no packets otherwise. Therefore, $X[P,W]$ is obtained by first allocating slots to nodes that have a packet, and then allocating any remaining slots to nodes that do not have a packet. ■

Thus, we can find the optimal solution for any deadline $W$ at each parent having only leaf nodes as children. Now, assume that we can find the optimal solution for any deadline $W$ at each parent node that is $h$ hops away from the sink.

**Lemma IV.4.** *Consider a node $P$ that is $h-1$ hops from the sink having $k$ children $C_1$,...,$C_k$. Let $P$ have a waiting time $W$. Then, at least one of the solutions for the optimal waiting times of $C_1$,...,$C_k$ will satisfy the following conditions.*

- *If $W \geq k$, then $n_i = 1 \; \forall \; i \in \{C_1, C_2, ..., C_k\}$, and each child $C_1$,...,$C_k$ transmits in one of the slots in the set $\{W-1, ..., W-k\}$ where no two nodes transmit in the same slot.*
- *If $W < k$, then $n_i = 1$ for exactly $W$ children among the $k$ children. These $W$ children transmit in one of the slots in the set $\{0, 1, ..., W-1\}$ and no two nodes transmit in the same slot.*

*Proof:* We prove both the cases in this result by contradiction.

*Case 1: $W \geq k$*

Suppose $n_i \neq 1$ for some node $C_i$. Then there is at least one slot during which there is no transmission. By scheduling node $C_i$ to transmit in this slot, we can obtain at least as much aggregated information as when $C_i$ was not provided a slot to transmit.

Suppose some node $C_i$ does not transmit in one of the slots in the set $\{W-1, ..., W-k\}$, i.e., $\{W-1, ..., W-k\}$ is not a set of optimal waiting times for $C_1$, $C_2$, ...., $C_k$. Suppose $C_i$ transmits in the slot $W-k-\alpha, \alpha > 0$. Then there exists at least one slot in the set $\{W-1, ..., W-k\}$ during which

no transmission takes place. If the node $C_i$ had transmitted during this slot, the aggregated information that we would have obtained would be at least as much as what would have been obtained if it had transmitted during the slot $W-k-\alpha$ (by Lemma IV.2), thus contradicting our assumption that $\{W-1, ..., W-k\}$ is not a set of optimal waiting times for $C_1$, $C_2$, ...., $C_k$.

*Case 2:* $0 \leq W < k$

If $n_i = 1$ for more than $W$ nodes, then the solution is infeasible since it is not possible to schedule more than $W$ nodes in $W$ slots. In particular, the constraint in (3) is violated.

On the other hand, if $n_i = 1$ for less than $W$ nodes, then by a similar argument as made in Case 1, we can obtain at least as much aggregated information by making $W$ nodes transmit.

Similarly, the rest of the proof follows from the same argument as made in Case 1. ■

**Theorem IV.5.** *Algorithm $A$ results in an optimal solution to problem $Z$.*

*Proof:* We have found the set of optimal waiting times for the children $C_1, ..., C_k$. We now only need to assign these slots to the $k$ children.

Suppose $X[i, W]$ represents the optimal number of nodes that node $i$ can account for if its waiting time is $W$, $0 \leq W \leq D-1$. Then the problem of finding $X[i, W]$ is an MWM problem in a bipartite graph with node-exclusive interference. Consider the bipartite graph in Figure 3. The nodes at the top of the graph represent the children nodes and the nodes at the bottom represent waiting times. If a child node $C_i$(say) has a waiting time of $W-j$, $1 \leq j \leq \min(W, k)$, the edge connecting $C_i$ and $W-j$ has a weight $X[C_i, W-j]$. No two nodes can have the same waiting time and a particular node can be allotted at most one waiting time. The goal is to assign waiting times to the children nodes such that the sum of the weights on the edges assigned is maximized. This is clearly an MWM problem and can be solved in polynomial time for a one-hop interference model.

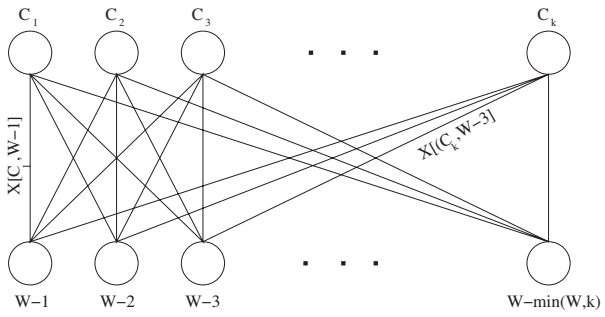



Fig. 3: Maximum Weighted Matching Solution

Hence we have now solved the optimization problem for nodes that are $h-1$ hops from the sink. By induction, from Lemmas IV.3 and IV.4, given the waiting time of the parent node at any hop, the optimal waiting times and the optimal number of transmissions of the children nodes in that hop can be determined using algorithm $A$. Hence, at the sink, $X[S, D]$ can be determined. Thus, algorithm $A$ results in an optimal solution to problem $Y$.

Now from Theorem III.1, the theorem is proved. ■

## V. Computational Complexity of Algorithm $A$

In this section, we analyze the computational complexity of algorithm $A$ described in Section 4. Let the farthest node (in terms of number of hops) be $h$ hops away from the sink. Let the in-degree of each node in the tree (apart from leaf nodes) be bounded by $k$. By this, we mean that each non-leaf node has at most $k$ children. Let $D$ represent the deadline imposed by the sink. Let the total number of sensor nodes (not including the sink) be $N$.

**Theorem V.1.** *The time complexity of algorithm $A$ is $O(hk^2(D+k)\log k)$.*

*Proof:* At every node $i$, we need to calculate $X[i, W]$, $0 \leq W \leq D-1$.

$$\begin{aligned} W = 0 \quad &\Rightarrow \text{ Time } = 0 \\ W = 1 \quad &\Rightarrow \text{ Time } \leq k \\ W = 2 \quad &\Rightarrow \text{ Time } \leq k(k-1) \\ 3 \leq W \leq k \quad &\Rightarrow \text{ Time } \leq (k+W)^2 \log(k+W) \\ k \leq W \leq D-1 \quad &\Rightarrow \text{ Time } \leq (2k)^2 \log(2k) \end{aligned} \tag{7}$$

We obtain (8) from the fact that the MWM problem in a bipartite graph can be solved in $O(V^2 \log V + VE)$ time [13], where $V$ is the number of vertices and $E$ is the number of edges. When $W \geq k$, the vertices and edges remain the same but the edge weights change.

Therefore the total time required at node $i$ is bounded by $k + k(k-1) + \sum_{W=3}^{k-1} (k+W)^2 \log(k+W) + (D-k)(2k)^2 \log(2k)$. Since for $3 \leq W \leq k$, $\log(k+W) \leq \log(2k)$, the time required is bounded by

$$\begin{aligned} & k^2 + \sum_{W=3}^{k-1} (k+W)^2 O(\log k) + 4k^2(D-k)O(\log k) \\ = \quad & k^2 + O(\log k)(4k^2(D-k) + \sum_{W=3}^{k-1} (k+W)^2) \\ = \quad & k^2 + O(k^2 D \log k) + O(k^3 \log k) \\ = \quad & O(k^2(D+k)\log k) \end{aligned}$$

Since nodes that are equal number of hops away from the sink can perform this computation in parallel and since we have $h$ hops, the complexity of algorithm $A$ is $O(hk^2(D+k)\log k)$. ■

## VI. Interpretations and Applications

In this section, we discuss a number of different interpretations of problem $Y$.

The optimization framework that we have described in Section 3 is general and can be interpreted in a number of ways. A few such interpretations are listed below.

1) **Priority**: Suppose that each source node $i$ has a priority, or more specifically an importance metric $\rho_i$ associated with the packet that it generates. Then, instead of maximizing the number of source nodes accounted for at the sink, we can maximize the total priority of packets accounted for at the sink within a deadline. This can be done by appropriately modifying $X[\cdot,\cdot]$ so that for a node $i$ with waiting time $W$, $X[i,W]$ represents the maximum sum priority of packets that can be accounted for by node $i$ if it waits for a time $W$.
2) **Observation Errors**: The data observed by sensor nodes may not be accurate. Suppose we associate a certain "confidence index" to each node's observation, we can then maximize the data accuracy at the sink by maximizing the total confidence index.
3) **Energy Constraint**: Suppose we have an additional constraint that the number of time slots during which a node $i$ can transmit/receive is limited to $r_i$ (say). Nodes go to sleep during other slots. Then, since each node transmits at most once, we can modify the constraint in (3) to $\sum_{j:(j,i)\in E} n_j \leq \min(r_i - 1, W_i - \min_{j:(j,i)\in E} W_j)$ $\forall i \in V \backslash V_L$. This problem can be solved by modifying Step 2 of algorithm $A$. Specifically, in order to calculate $X[i,W]$ for a node $i$ with $k$ children $C_1, C_2, ...C_k$, the waiting times of the children can now only take values in the set $\{W-1, W-2, ..., \max(0, W-k, W-(r_i-1))\}$ which is a subset of $\{W-1, W-2, ..., \max(0, W-k)\}$.
4) **The Dual Problem**: The dual problem of $Y$ is to determine the minimum deadline $D$ by which at least a certain number of sources ($K$) are accounted for at the sink. This problem can be solved by combining algorithm $A$ with a binary search algorithm as follows. It uses the fact that $X[S,D]$ is a non-decreasing function of $D$.
   a) Initialize $D = \lfloor \frac{N}{2} \rfloor$, $D_{left} = 0$, $D_{right} = N$. Calculate $X[S,D]$ and $X[S,D-1]$ using Algorithm $A$.
   b) If $X[S,D] < K$, set $D_{left} = D$.
   c) If $X[S,D-1] \geq K$ and $X[S,D] \geq K$, set $D_{right} = D$.
   d) If $X[S,D-1] < K$ and $X[S,D] \geq K$, return $D$ and stop.
   e) Set $D = \frac{1}{2}(D_{left} + D_{right})$, and go to Step b).

   The above algorithm will terminate and return a $D$ for which $X[S,D-1] < K$ and $X[S,D] \geq K$. Thus, it determines the minimum deadline by which at least $K$ packets reach the sink. The complexity of this algorithm is $O(\log N)$ times the complexity of algorithm $A$.

## VII. Extensions

In this section, we explain the structure of the solution when some of the assumptions that we made in Section II are relaxed. Some of these extensions are relatively straightforward while others require a more careful investigation.

### A. Different Observation Instants

In our previous discussion, we had assumed that source nodes simultaneously observe the event at time zero. Suppose that each node observes the event at a certain *known* time and that the deadline is measured from the instant the first source node observes the event. For example, each node could be periodically observing an event. However, a source node close to the sink could observe the event as late as possible (and still send its data to the sink) so that its data is fresher than the data observed by nodes farther away from the sink. Algorithm $A$ can clearly be applied in a straightforward manner to solve this problem. For a source node $i$, $X[i,W]$ will account for the data from node $i$ only if $W$ is larger than the time instant at which $i$ observes the data. The rest of the algorithm remains the same.

### B. Arbitrary Link Capacities

In Section II, we assumed unit link capacities for each link in the network. We now study the case where each link has an arbitrary fixed capacity. We first observe that Theorem III.1 does not depend on the link capacity. This implies that the optimal aggregation policy remains the same even when each link has an arbitrary fixed capacity. Therefore, to solve problem $Z$ with arbitrary link capacities, we only need to optimally determine $X[\cdot,\cdot]$ and assign waiting times to nodes.

Let us begin with a simple case. Suppose that within a hop in the tree, each link (connecting the given parent to its children) has the same capacity (not necessarily 1). Link capacities may be different across hops. For example, consider a parent $P$ having four children with each of the link capacities being equal to $\frac{1}{2}$. It takes two slots for each child to transmit a packet. Then, for a waiting time $W$ for the parent, the optimal waiting times of the children falls in the set $\{W-2, W-4, W-6, W-8\}$, where no two children can be assigned the same waiting time and no two waiting times can be assigned to the same child. For instance, if a child is assigned a waiting time $W-2$, it uses slots $W-2$ and $W-1$ for transmitting its packet. Then, $X[P,W]$ can be calculated using the same Maximum Weighted Matching algorithm described in Section IV. Note that the transmission slots assigned to a child here are consecutive. This turns out to be one of the optimal solutions (Lemma VII.1).

Now, let us consider a more general case in which each node requires an arbitrary fixed integral number of time slots to transmit its packet. In this case, we cannot calculate $X[\cdot,\cdot]$ using a Maximum Weighted Matching algorithm. Calculating $X[i,\cdot]$ for any leaf node $i$ is still straightforward, since $X[i,W] = T_i$ for all $W \in \{0,1,...,D-1\}$. We now provide an algorithm for determining $X[i,W]$ for any node $i$ that is not a leaf node, and for any given $W$. Assume that $i$ has $k$ children, $C_1$,...,$C_k$, and that node $C_j$ requires $\alpha_j$ time slots ($j = 1,2,3,...,k$) to transmit a packet to its parent.

**Algorithm** $A_1$:

1) Construct an interference graph, $G'$, as follows. For each child $C_j$, we construct $\sum_{l=1, l\neq j}^{k} \alpha_l$ nodes labeled

$(C_j, W - \sum_{l=1}^{k} \alpha_l)$, $(C_j, W+1-\sum_{l=1}^{k} \alpha_l)$, ..., $(C_j, W-\alpha_j)$, respectively. Here the second term of the label of each node denotes the time at which $C_j$ starts transmitting. In other words, it denotes the waiting time of $C_j$. A node labeled $(C_j, W_j$ in this interference graph is assigned a weight $X[C_j, W_j]$. Consider two nodes, $(C_m, W_m)$ and $(C_n, W_n)$. There exists an edge between these two nodes if and only if *(a)* $\{W_m, W_m + 1, ..., W_m + \alpha_m - 1\} \cap \{W_n, W_n + 1, ..., W_n + \alpha_n - 1\} \neq \emptyset$, or *(b)* $C_m = C_n$.
[1]

2) Find a Maximum Weighted Independent Set in $G'$. $X[i, W]$ is given by the weight of this Maximum Weighted Independent Set.

We will now show that the above algorithm yields the maximum amount of aggregated information that can be accounted for by node $i$ if it waits for $W$ slots. We will assume that $W - \sum_{l=1}^{k} \alpha_l \geq 0$. Note that if this is not true, we can replace $W - \sum_{l=1}^{k} \alpha_l$ by $\max(0, W - \sum_{l=1}^{k} \alpha_l)$, and our results will still hold.

**Lemma VII.1.** *Assume that a node $i$ needs to make $k$ transmissions to send a packet to its parent $P$ (i.e., the link capacity is $\frac{1}{k}$). Let $W_1^* < W_2^* < ... < W_k^*$ be the $k$ time slots during which $i$ makes these transmissions. If $W_1^*, W_2^*, ..., W_k^*$ are optimal transmission slots for node $i$ for problem $Z$, then $W_1^*, W_1^* + 1, ..., W_1^* + k - 1$ are also optimal transmission slots for node $i$ for problem $Z$.*

*Proof:* Before we describe the details of the proof, it is important to note that since node $i$ is sending an aggregated packet, it cannot modify the aggregated information once it has started transmitting the packet. Therefore, the amount of aggregated information accounted for by node $i$ is given by $X[i, \min(W_1^*, ..., W_k^*)] = X[i, W_1^*]$. We now prove the result by induction on $k$.

$k = 2$: We show the result for $k = 2$ by contradiction. Suppose that $W_1^*, W_1^* + 1$ are not optimal transmission slots for node $i$ for problem $Z$. Consider the slot $W_1^*+1$ in which a node $j$ that interferes with $i$ is scheduled. Suppose that node $j$ is now scheduled at $W_2^*$ instead of $W_1^*+1$, and that node $i$ is now scheduled at $W_1^*+1$. Note that this is a feasible schedule because of our aggregation policy, and because of the fact that we have a tree network with one-hop interference constraints. Suppose that $j$ made its first transmission in the slot $W'^*$. We consider two cases.

Case 1: $W'^* = W_1^* + 1$

In this case, by interchanging the schedules of $i$ and $j$, the total aggregated information accounted for by nodes $i$ and $j$ is given by $X[i, W_1^*] + X[j, W_2^*] \geq X[i, W_1^*] + X[j, W_1^* + 1]$ (by Lemma IV.2 since $W_2^* \geq W_1^* + 1$). This contradicts our assumption that $W_1^*, W_1^*+1$ are not optimal transmission slots for node $i$.

Case 2: $W'^* < W_1^* + 1$

In this case, by interchanging the schedules of $i$ and $j$, the total aggregated information accounted for by nodes $i$ and $j$ is given by $X[i, W_1^*] + X[j, W'^*]$, which is the same amount of total aggregated information that we get even without interchanging schedules.

Thus, combining cases 1 and 2, by interchanging the schedules of $i$ and $j$, we get at least as much total aggregated information from $i$ and $j$ as that of the original schedule. This contradicts the fact that $W_1^*, W_1^* + 1$ are not optimal transmission slots for node $i$ for problem $Z$.

$k = m$: Assuming that the result is true for $k = m$, i.e., if node $i$ needs to make $m$ transmissions to send a packet to its parent $P$, and if $W_1^*, W_2^*, ..., W_m^*$ are optimal transmission slots for node $i$ for problem $Z$, then $W_1^*, W_1^*+1, ..., W_1^*+m-1$ are also optimal transmission slots for node $i$ for problem $Z$.

$k = m + 1$: Since the result is true for $k = m$, $W_1^*, W_1^* + 1, ..., W_1^* + m - 1$ are optimal transmission slots for node $i$ for problem $Z$. We now need to show that if the node (that interferes with $i$) that transmits at $W_1^* + m$ is now made to transmit at $W_m^*$, and $i$ transmits at $W_1^* + m$, the resulting schedule is still optimal. Note that the proof for this case is identical to that for the case $k = 2$.

Hence, the result follows by induction. ■

**Lemma VII.2.** *Consider any node $i$ other than the leaf nodes. Suppose that $i$ has $k$ children, $C_1$,...,$C_k$. Let node $C_j$ require $\alpha_j$ time slots to transmit its packet. If $W^*$ is the optimal waiting time of node $i$, one of the optimal set of slots during which the children transmit is given by* $\{W^* - \sum_{j=1}^{k} \alpha_j, W^* + 1 - \sum_{j=1}^{k} \alpha_j, ..., W^* - 1\}$.

*Proof:* From Lemma VII.1, we know that once a node starts transmitting, it transmits in consecutive slots until it finishes transmitting the entire packet. The proof now follows by contradiction.

Suppose that the set of time slots $\{W^* - \sum_{j=1}^{k} \alpha_j, W^* + 1 - \sum_{j=1}^{k} \alpha_j, ..., W^* - 1\}$ is not optimal. This implies that at least one of the children transmits before $W^* - \sum_{j=1}^{k} \alpha_j$. This means that there exists at least one time slot in $\{W^* - \sum_{j=1}^{k} \alpha_j, W^* + 1 - \sum_{j=1}^{k} \alpha_j, ..., W^* - 1\}$ during which none of $C_1, C_2, ..., C_k$ make a transmission. If the child that transmitted before $W^* - \sum_{j=1}^{k} \alpha_j$ had waited until this free slot, the total aggregated information accounted for by the $k$ children in this case would have been at least as much as that in the original case. This contradicts the assumption that the set of time slots

[1]Note that a node scheduled to transmit at a slot $W$ transmits during the interval $[W, W + 1)$. Thus, the set of transmission slots $\{W_m, W_m + 1, ..., W_m + \alpha_m - 1\}$ represents the interval $[W_m, W_m + \alpha_m)$.

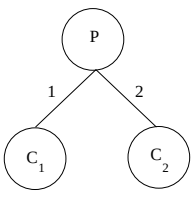
$\{W^* - \sum_{j=1}^{k} \alpha_j, W^* + 1 - \sum_{j=1}^{k} \alpha_j, ..., W^* - 1\}$ is not optimal. ■

**Theorem VII.3.** *Algorithm $A_1$ yields the maximum amount of aggregated information accounted for by node $i$ if it waits for $W$ slots.*

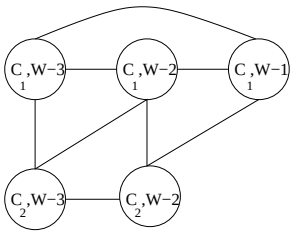
*Proof:* First, we explain the intuition behind the construction of the interference graph in $A_1$. Within the set $\{W - \sum_{j=1}^{k} \alpha_j, W + 1 - \sum_{j=1}^{k} \alpha_j, ..., W - 1\}$, we first determine the possible transmission slots for each child. For example, consider $C_1$. $C_1$ can start transmitting at any of $\{W - \sum_{j=1}^{k} \alpha_j, W + 1 - \sum_{j=1}^{k} \alpha_j, ..., W - \alpha_1\}$ and make $\alpha_1$ consecutive transmissions until it finishes transmitting its packet. Note that due to interference, no other child can transmit when $C_1$ is making these consecutive transmissions. For example, if $C_1$ starts transmitting at $W - \alpha_1$, then no other child can transmit during the slots $W - \alpha_1$, $W + 1 - \alpha_1$, ..., and $W - 1$.

We construct the interference graph $G'$ as illustrated in Step 1 of algorithm $A_1$. From the construction, it is clear that two nodes in $G'$ will be in an independent set (which is a set of vertices in a graph, no two of which are adjacent) only if (a) the nodes represent different children, and (b) the transmission slots of the nodes do not intersect. Since a Maximum Weighted Independent Set is an independent set of maximum total weight, $X[i, W]$ is obtained by finding such a set in $G'$. ■

**Example**: Figures 4(a) and 4(b) give an example of the construction of $G'$. In Figure 4(a), parent node $P$ has two children $C_1$ and $C_2$. It takes one time slot to send a packet from $C_1$ to $P$, and it takes two time slots to send a packet from $C_2$ to $P$. Figure 4(b) shows the interference graph, which will be used to calculate $X[P, W]$ at parent $P$. Note that $C_2$ cannot begin transmitting at $W - 1$, since it cannot make two transmissions before the deadline $W$ expires. So we do not have a node labeled $(C_2, W-1)$. Each node in the interference graph has a weight as explained before. For instance, the node labeled $(C_1, W-3)$ has a weight $X[C_1, W-3]$. Note that there are three maximal independent sets for this graph, namely, $\{(C_1, W-2)\}$, $\{(C_1, W-3), (C_2, W-2)\}$, and $\{(C_1, W-1), (C_2, W-3)\}$. The maximum independent set corresponds to that maximal independent set that has the maximum total weight. This corresponds to $\max(X[C_1, W-2], X[C_1, W-3] + X[C_2, W-2], X[C_1, W-1] + X[C_2, W-3])$.




(a) Arbitrary Link Capacities




(b) Interference Graph Construction

**Computational complexity of $A_1$**

Finding a maximum weighted independent set is, in general, an NP-Hard problem. However, note that the number of nodes in $G'$ is $O(k \sum_{j=1}^{k} \alpha_j)$, where $k$ is the number of children. The number of children is typically small. Therefore, in this case, finding a maximum weighted independent set may not be computationally complex. However, a maximum weight independent set algorithm for a general graph has a complexity $O(2^{cn})$ where $c$ is a constant and $n$ is the number of nodes. Here, $n = k \sum_{j=1}^{k} \alpha_j = O(k^2)$. Even when $k$ is a small constant, it is computationally complex to find a maximum weight independent set using a general algorithm. Therefore, we need to find an algorithm with significantly lower complexity. It is well known that the complexity of the maximum weight independent set algorithm depends on the structure of the graph. We now show that the problem of finding a maximum weight independent set in $G'$ is equivalent to finding a profit-maximizing schedule for a Job Interval Scheduling Problem (JISP). JISP is a well-studied problem in integer optimization theory. It is known to be MAX-SNP Hard, which implies that unless P=NP it is not possible to find a Polynomial-Time Approximate Solution (PTAS) to JISP.

**Theorem VII.4.** *Determining $X[i, W]$ for an arbitrary node $i$ having $k$ children, and for an arbitrary waiting time $W$ is MAX-SNP Hard when the input to the problem is $k$.*

*Proof:* We briefly describe the Job Interval Scheduling Problem (JISP). In a single-server JISP, $n$ jobs need to be served by a single machine. Each job has $k$ instances, where each instance is associated with an explicit time interval during which it must be scheduled, and a certain profit. The machine can only serve one instance of any job during each time slot. The goal is to find a schedule such that at most one instance of a job is present in the schedule, the instances in the schedule do not conflict in time, and the sum of the profits of the job instances is maximum.

Determining $X[i, W]$ for an arbitrary node $i$ and an arbitrary waiting time $W$ is a JISP. This can be shown as follows. In our problem, the jobs correspond to the children nodes that need to be served by the parent. Each instance corresponds to a node in the interference graph $G'$. Recall that a node in $G'$ is associated with a child node, the interval during which it transmits, and a weight. The interval corresponds to the interval of the job instance, and the node weight corresponds to the profit of the job instance. Thus, our problem is identical to JISP.

Hence, when the input to the problem is the number of children $k$ (which corresponds to the number of jobs $n$ in JISP), determining $X[i, W]$ is MAX-SNP Hard. ■

A polynomial time algorithm that provides a constant factor approximation of $\frac{1}{2}$ to JISP can be found in [14]. This means that this algorithm guarantees that the total profit of the schedule that it finds is at least $\frac{1}{2}$ of the total profit of the optimal schedule. However, while determining $X[i, W]$ for a given node $i$ and a given waiting time $W$ is a JISP, determining

$X[S, D]$ requires calculating $X[\cdot, \cdot]$ for all other nodes in the network. If the maximum number of hops in the network from a leaf node to the sink is $h$, then the approximation factor by directly applying the JISP approximation algorithm could be as poor as $\frac{1}{2^h}$. Hence, it is necessary to look at other approaches than just using an existing JISP approximation scheme.

It is important to observe that even though our solution has exponential complexity, the input is the number of children in each hop (or, in other words, the degree of each node). In many cases, the number of children is a small constant. Therefore, we also wish to find the complexity of an optimal algorithm even if it has exponential complexity in the number of children. We now provide an $O(k!)$ algorithm for finding a Maximum Weighted Independent Set in $G'$, where $k$ corresponds to the number of children. We observe that given the order in which children transmit, we can find an independent set in $G'$ in constant time. For instance, if it is known that the order of transmission is given by $C_1 \rightarrow C_2 \rightarrow ... \rightarrow C_k$, then the corresponding independent set in $G'$ is given by $\{(C_1, W - \sum_{l=1}^{k} \alpha_l), (C_2, W + \alpha_1 - \sum_{l=1}^{k} \alpha_l), (C_3, W + \alpha_1 + \alpha_2 - \sum_{l=1}^{k} \alpha_l), ..., (C_k, W - \alpha_k)\}$. This follows from Lemmas VII.1 and VII.2. Since there are $k$ children, there are exactly $k!$ ways in which the children can be ordered to transmit. The Maximum Weight Independent Set corresponds to the order that has the maximum total weight.

The above derivation holds even when all the link capacities (in any particular hop) are distinct from each other. In many cases, it may turn out that there are only a few fixed set of rates at which a node can transmit. When the link capacities (within a hop) were identical, we showed that we can calculate $X[\cdot, \cdot]$ using a Maximum Weighted Matching algorithm.

Now suppose that there are only two distinct link capacities (say $\gamma$ and $\delta$), i.e., $\alpha_1, \alpha_2, ..., \alpha_k \in \{\gamma, \delta\}$. Then we can find an algorithm with a complexity lower than $O(k!)$ to find $X[\cdot, \cdot]$. We do this as follows.

Suppose that $r$ out of $k$ children have link capacity $\gamma$, and the rest have link capacity $\delta$. The number of ways of arranging $k$ items of which $r$ are of one type, and the rest are of a second type is given by $\frac{k!}{r!(k-r)!}$. Let us consider one such arrangement of link capacities. Each arrangement of link capacities corresponds to an arrangement of waiting times. For example, consider three children requiring 10, 20, and 10 transmission slots respectively. Then, given an arrangement, 10-20-10 (say), the waiting times are given by $W-40$, $W-30$, and $W-10$. Since the second child requires 20 transmission slots, it transmits at W-30. However, we do not know whether the first child transmits at $W-10$ or at $W-40$. Thus, while we obtain an arrangement of waiting times, we do not immediately have an assignment of these waiting times to the children. This assignment can be done using a Maximum Weighted Matching algorithm. We know that only the first or the third child can transmit at $W-40$ and $W-10$. So we match these children to these waiting times so that the sum of the weights is maximized. Figure 4 illustrates this example. For two possible arrangements of link capacities, 10-20-10 and 10-10-20, the corresponding set of waiting times is given by $\{W-40, W-30, W-10\}$ and $\{W-40, W-30, W-20\}$, respectively. For the arrangement 10-20-10, the second child is automatically assigned $W-30$, but the first and the third children have to be matched to the set $\{W-40, W-10\}$. This is accomplished by performing a Maximum Weighted Matching between these children and the waiting times. The explanation is similar for the other arrangement (10-10-20). Note that there are a total of three possible arrangements here, namely, 10-10-20, 10-20-10, and 20-10-10.

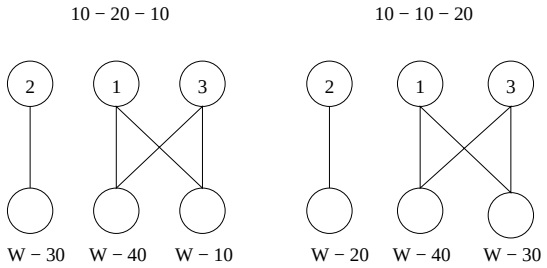



Fig. 4: Two ways of arranging link capacities and the corresponding Maximum Weighted Matchings

We provide a procedure below that determines $X[i, W]$ when the link capacities of children take $m$ ($m < k$) distinct values ($\gamma_1, \gamma_2, ..., \gamma_m$). For each possible arrangement of link capacities, do the following to calculate its weight.

1) Assign waiting times corresponding to this arrangement of link capacities.
2) Split the children into $m$ classes, where each child in the first class has link capacity $\gamma_1$, each child in the second class has link capacity $\gamma_2$, and so on.
3) Split the waiting times obtained in Step 2 into $m$ classes, where each waiting time in the first class corresponds to a link capacity $\gamma_1$, that in the second class corresponds to a link capacity $\gamma_2$, and so on.
4) Perform a Maximum Weighted Matching between the children in the $i^{th}$ class, and the waiting times in the $i^{th}$ class, $i = 1, 2, ..., m$.
5) Add the weights of all the $m$ classes to obtain the weight of the arrangement.

Finally, determine the arrangement that provides the maximum weight.

We now analyze the complexity of this procedure. Let $r_i$ be the number of nodes that have link capacity $\gamma_i$ to their parent, $i = 1, 2, ..., m$. Note that $\sum_{i=1}^{m} r_i = k$. The number of arrangements of link capacities is now given by $\frac{k!}{r_1!r_2!r_3!...r_m!}$. The complexity of the Maximum Weighted Matching for each of these arrangements is $O(\sum_{i=1}^{m} r_i^3)$. Therefore, the overall complexity of the procedure is $O(\frac{k!}{r_1!r_2!r_3!...r_m!}(\sum_{i=1}^{m} r_i^3))$.

To understand this complexity result, let us consider the case $m = 2$. Consider the worst case. This corresponds to finding $r$ that maximizes $\frac{k!}{r!(k-r)!}$. We know that this is given by $r = \frac{k}{2}$ when $k$ is even, and $r = \frac{k+1}{2}$ when $k$ is odd. Using Stirling's

approximation, this maximum value is approximately equal to $\frac{2^k}{\sqrt{k}}$. The Maximum Weighted Matching has a complexity of $O(k^3)$. Thus the overall complexity is given by $O(k^{2.5}2^k)$. Note that this is a significant reduction from the $O(k!)$ complexity when the link capacities were distinct.

Clearly, when $m = 1$, the procedure has the same complexity as that of Algorithm $A$, and when $m = k$, the procedure has the same complexity as that of Algorithm $A_1$.

### C. Multiple Events

In this section, we relax our earlier assumption that events must be non-overlapping, and extend our solution to multiple overlapping events. For the purpose of illustration, we explain the case with two overlapping events. This can be easily extended to a larger number of events.

We consider a tree with the sink as the root. The objective here is to maximize the total number of source nodes accounted for at the sink by both the events within their respective deadlines (say, $D_1$ and $D_2$). Note that packets of two different events cannot be aggregated together. We assume unit capacity links and a one hop interference model.

The optimal aggregation policy for this problem is a simple extension of the policy for a single event. For two events, a node needs to make at most two transmissions, one for each event. The problem can now be solved as follows.

Let $X[i, W_1, W_2]$ represent the maximum total number of source nodes, from both events, that node $i$ can account for if it transmits the packet corresponding to the first event at time slot $W_1$, and the packet corresponding to the second event at time slot $W_2$. We need to calculate $X[S, D_1, D_2]$ at the sink (with a minor modification that the sink does not transmit packets).

As described in the previous sections, we calculate $X[\cdot, \cdot, \cdot]$ from leaves to root, and assign waiting times from root to leaves. For any leaf node $l$, $X[l, W_1, W_2] = T_l^{(1)} + T_l^{(2)}$ for every $W_1 \in \{0, 1, ..., D_1 - 1\}$, $W_2 \in \{0, 1, ..., D_2 - 1\}$, and $W_1 \neq W_2$, where $T_l^{(i)} = 1$, if node $l$ is a source for event $i$. Otherwise, $T_l^{(i)} = 0$. Note that $W_1 \neq W_2$ because a node can make only one transmission during a single time slot.

Now consider any node $i$ having $k$ children, $C_1$, $C_2$, ..., $C_k$. We need to calculate $X[i, W_1, W_2]$ for every $W_1 \in \{0, 1, ..., D_1 - 1\}$, $W_2 \in \{0, 1, ..., D_2 - 1\}$, and $W_1 \neq W_2$. We explain how to perform this calculation when $W_1 < W_2$. The calculation can be done in a similar manner when $W_2 < W_1$. We now consider the following cases.

*Case 1:* $W_2 - W_1 \geq k + 1$

This means that the deadline for the second event at node $i$ is at least $k$ time slots more than the deadline for the first event at node $i$. Note that slot $W_1$ is allocated to node $i$ for transmitting the packet corresponding to the first event. This condition implies that the children of node $i$ need not start transmitting their packets corresponding to the second event before $W_1$. This is because, if a child transmits its packet corresponding to the second event before $W_1$, then at least one of the time slots in $\{W_1 + 1, W_1 + 2, ..., W_2 - 1\}$ is empty. If that child had waited until this empty slot, it could have potentially aggregated more packets corresponding to the second event and still made a successful transmission. Therefore, the set of possible transmission slots for the second event is given by $\{W_2 - k, W_2 - k + 1, ..., W_2 - 1\}$ (since $W_2 - k \geq W_1 + 1$).

For determining the set of possible transmission slots for the first event, we also need to consider the transmissions made by the children of each child of $i$. This is because the second-hop children of $i$ could be sending packets corresponding to the second event while their parent is sending packets corresponding to the first event to node $i$. For simplicity of exposition, we will assume that each child of $i$ has $k$ children. Note that the sum of the number of transmissions made by $i$'s children before $W_1$ is at most $k$, and each child of $i$ receives at most $2k$ transmissions ($k$ transmissions for each event) from its children before $W_1$. Therefore, no child of $i$ needs to transmit its packet corresponding to the first event before $W_1 - 3k$. If a child of $i$ transmits before $W_1 - 3k$, then there exists a slot in $\{W_1 - 3k, W_1 - 3k + 1, ..., W_1 - 1\}$ during which none of the children of $i$ and none of the children of $i$'s children transmit. Therefore, that child could have waited until this free slot and still made a successful transmission, after having potentially aggregated more packets. Note that we have assumed here that $W_1 - 3k \geq 0$. When this is not true, we can always replace $W_1 - 3k$ by $\max(W_1 - 3k, 0)$.

Given an optimal set of possible transmission slots, it is now straightforward to compute $X[i, W_1, W_2]$. As in the previous sections, we construct an interference graph, $G'$. The nodes in $G'$ are labeled $(C_j, W_{j1}, W_{j2})$, where $C_j$ represents the child, $W_{j1}$ represents the time slot during which $C_j$ transmits its packet corresponding to the first event, and $W_{j2}$ represents the time slot during which $C_j$ transmits its packet corresponding to the second event. Note that there are a total of $3k^3$ nodes in $G'$, corresponding to all combinations of $\{C_1, ..., C_k\}$, $\{W_1 - 3k, ..., W_1 - 1\}$, and $\{W_2 - k, ..., W_2 - 1\}$. A node labeled $(C_j, W_{j1}, W_{j2})$ is assigned a weight $X[C_j, W_{j1}, W_{j2}]$. There exists an edge between two nodes in $G'$ if and only if at least one of the three terms in the names of the nodes are identical. This means that there exists an edge between two nodes if and only if both the nodes represent the same child, or, both the nodes have the same transmission slot for the first event, or, both the nodes have the same transmission slot for the second event.

We now obtain $X[i, W_1, W_2]$ by calculating the total weight of a Maximum Weighted Independent Set of $G'$. We can thus compute $X[S, D_1, D_2]$ at the sink and assign waiting times for the two events from the root down to the leaves.

*Case 2:* $W_2 - W_1 \leq k$

The main difference between the solution for this case and that of Case 1 is that the set of possible transmission slots for the two events are different. In this case, it may not be optimal to begin transmissions for the second event after the deadline for the first event expires, because there may not exist enough time slots for all children to make transmissions corresponding to the second event between $W_1$ and $W_2$. Based on the fact that the total number of transmissions made by $i$'s children is at most $2k$, and that the total number of transmissions made by the children of any of $i$'s children is also at most $2k$, one of the optimal set of transmission slots for both events lies in the set

$\{W_1-4k, W_1-4k+1, ..., W_1-1, W_1+1, W_1+2, ..., W_2-1\}$. This is because if a child of $i$ transmits before $W_1 - 4k$, then there exists a slot in the set above during which none of $i$'s children and none of the children of $i$'s children make a transmission. Therefore, if that child of $i$ had waited until this free slot, it could have potentially aggregated more packets from its children, and still made a successful transmission.

Now, given this set of possible transmission slots, $X[i, W_1, W_2]$ can be computed as shown in Case 1.

We have thus solved the problem of maximizing the total number of sources that can be accounted for at the sink by two events (potentially overlapping with each other) within their respective deadlines, under node-exclusive interference constraints. It is also possible to extend this to an arbitrary number of events. However, the complexity of finding a Maximum Weight Independent Set will exponentially increase in the number of events. Note that even when there are multiple events, if the deadlines of the events are far enough from each other that it is enough to start transmitting packets for the next event after the expiration of the deadline for the previous event, then this problem is significantly less complicated. One just needs to apply the algorithm in Section 4 for each event separately.

## VIII. Conclusion

In this paper, we have developed a general optimization framework for solving the problem of maximizing aggregated information in data aggregation trees when a deadline is imposed by the sink. We considered a one-hop interference model and proposed a polynomial time algorithm that uses only local information at each hop to obtain the optimal solution. Extensions to general interference models, and finding low complexity solutions for these models are challenging problems, and are a part of our future research. We discussed a number of interesting applications and interpretations to our solution, such as, incorporating sleep-wake scheduling, maximizing weighted aggregated information and maximizing accuracy. Finally, we extended our solution to account for more general problems concerning multiple overlapping events, and arbitrary link capacities.

## References


[1] S. Hariharan and N. B. Shroff, "Maximizing aggregate revenue in sensor networks under deadline constraints," in *Proceedings of the IEEE Conference on Decision and Control (CDC)*, 2009.

[2] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient Communication Protocols for Wireless Microsensor Networks," in *Proceedings of the International Conference on System Sciences*, 2000.

[3] Y. Wu, Z. Mao, S. Fahmy, and N. B. Shroff, "Constructing maximum-lifetime data-gathering forests in sensor networks," *IEEE/ACM Transactions on Networking*, vol. 18, no. 5, pp. 1571–1584, 2010.

[4] A. Goel and D. Estrin, "Simultaneous optimization for concave costs: Single sink aggregation or single source buy-at-bulk," in *Proceedings of the 14th Symposium on Discrete Algorithms (SODA)*, 2003.

[5] B. Krishnamachari, D. Estrin, and S. B. Wicker, "The impact of data aggregation in wireless sensor networks," in *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCSW)*, 2002.

[6] A. Boulis, S. Ganeriwal, and M. B. Srivastava, "Aggregation in sensor networks: An energy-accuracy trade-off," *Elsevier Ad Hoc Networks*, vol. 1, no. 2-3, pp. 317–331, 2003.

[7] Y. Yu, B. Krishnamachari, and V. K. Prasanna, "Energy-latency tradeoffs for data gathering in wireless sensor networks," in *Proceedings of IEEE INFOCOM*, 2004.

[8] L. G. J. Elson and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," in *Proceedings of the Symposium on Operating Systems Design and Implementation (OSDI)*, 2002.

[9] L. Becchetti, A. M. Spaccamela, A. Vitaletti, P. Korteweg, M. Skutella, and L. Stougie, "Latency-constrained aggregation in sensor networks," *ACM Transactions on Algorithms*, vol. 6, no. 1, pp. 13:1–13:20, 2009.

[10] Z. Ye, A. A. Abouzeid, and J. Ai, "Optimal stochastic policies for distributed data aggregation in wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 17, no. 5, pp. 1494–1507, 2008.

[11] W. Lai and I. C. Paschalidis, "Optimally balancing energy consumption versus latency in sensor network routing," *ACM Transactions on Sensor Networks*, vol. 4, no. 4, pp. 21:1–21:28, 2008.

[12] A. Giridhar and P. R. Kumar, "Computing and communicating functions over sensor networks," *IEEE Journal on Selected Areas in Communication*, vol. 23, pp. 755–764, 2005.

[13] D. Goldfarb, "Efficient dual simplex algorithms for the assignment problem," *Mathematical Programming*, vol. 33, no. 2, pp. 187–203, 1985.

[14] A. Bar-Noy, R. Bar-Yehuda, A. Freund, J. (Seffi) Naor, and B. Schieber, "A unified approach to approximating resource allocation and scheduling," *Journal of the ACM*, vol. 48, no. 5, pp. 735–744, 2001.



**Srikanth Hariharan** received his B.Tech. degree from the Indian Institute of Technology, Madras, in 2006, and his M.S. degree from Purdue University, West Lafayette, IN, in 2007. He is currently a Doctoral candidate in the Department of Electrical and Computer Engineering at the Ohio State University. His research interests include data aggregation in wireless sensor networks, target tracking, multi-hop wireless scheduling, and wireless security.



**Ness B. Shroff** (F'07) received his Ph.D. degree from Columbia University, NY, in 1994 and joined Purdue University as an Assistant Professor. At Purdue, he became Professor of the School of Electrical and Computer Engineering in 2003 and director of CWSA in 2004, a university-wide center on wireless systems and applications. In July 2007, he joined The Ohio State University as the Ohio Eminent Scholar of Networking and Communications, a chaired Professor of ECE and CSE. He is also a guest chaired professor of Wireless Communications and Networking in the department of Electronic Engineering at Tsinghua University. His research interests span the areas of wireless and wireline communication networks. He is especially interested in fundamental problems in the design, performance, pricing, and security of these networks.

Dr. Shroff is a past editor for IEEE/ACM Trans. on Networking and the IEEE Communications Letters and current editor of the Computer Networks Journal. He has served as the technical program co-chair and general co-chair of several major conferences and workshops, such as the IEEE INFOCOM 2003, ACM Mobihoc 2008, IEEE CCW 1999, and WICON 2008. He was also a co-organizer of the NSF workshop on Fundamental Research in Networking (2003) and the NSF Workshop on the Future of Wireless Networks (2009). Dr. Shroff is a fellow of the IEEE. He received the IEEE INFOCOM 2008 best paper award, the IEEE INFOCOM 2006 best paper award, the IEEE IWQoS 2006 best student paper award, the 2005 best paper of the year award for the Journal of Communications and Networking, the 2003 best paper of the year award for Computer Networks, and the NSF CAREER award in 1996 (his INFOCOM 2005 paper was also selected as one of two runner-up papers for the best paper award).