Latency Constrained Aggregation in Sensor Networks*

Luca Becchetti[1], Peter Korteweg[2], Alberto Marchetti-Spaccamela[1], Martin Skutella[3], Leen Stougie[2,4], and Andrea Vitaletti[1]

[1] University of Rome "La Sapienza"
[2] TU Eindhoven
[3] University of Dortmund
[4] CWI Amsterdam

Abstract. A sensor network consists of sensing devices which may exchange data through wireless communication; sensor networks are highly energy constrained since they are usually battery operated. Data aggregation is a possible way to save energy consumption: nodes may delay data in order to aggregate them into a single packet before forwarding them towards some central node (sink). However, many applications impose constraints on data freshness; this translates into latency constraints for data arriving at the sink.

We study the problem of data aggregation to minimize maximum energy consumption under latency constraints on sensed data delivery and we assume unique transmission paths that form a tree rooted at the sink. We prove that the off-line problem is strongly NP-hard and we design a 2-approximation algorithm. The latter uses a novel rounding technique.

Almost all real life sensor networks are managed on-line by simple distributed algorithms in the nodes. In this context we consider both the case in which sensor nodes are synchronized or not. We consider distributed on-line algorithms and use competitive analysis to assess their performance.

1 Introduction

A sensor network consists of sensor nodes and one or more central nodes or *sinks*. Sensor nodes are able to monitor events, to process the sensed information and to communicate the sensed data. Sinks are powerful base stations which gather data sensed in the network; sinks either process this data or act as gateways to other networks. Sensors send data to the sink through multi-hop communication.

A particular feature of sensor nodes is that they are battery powered, making sensor networks highly energy constrained. Replacing batteries on hundreds of

* Supported by EU Integrated Project AEOLUS (FET-15964), EU project ADONET (MRTN-CT-2003-504438), EU COST-action 293, MIUR-FIRB project VICOM, Dutch project BRICKS, DFG Focus Program 1126, "Algorithmic Aspects of Large and Complex Networks", grant SK 58/5-3, MIUR-FIRB Israel-Italy project RBIN047MH9.

Y. Azar and T. Erlebach (Eds.): ESA 2006, LNCS 4168, pp. 88–99, 2006.

nodes, often deployed in inaccessible environments, is infeasible or too costly and, therefore, the key challenge in a sensor network is the reduction of energy consumption. Energy consumption can be divided into three domains: sensing, communication and data processing [1]. Communication is most expensive because a sensor node spends most of its energy in data transmission and reception [7]. This motivates the study of techniques to reduce overall data communication, possibly exploiting processing capabilities available at each node. Data aggregation is one such technique. It consists of aggregating redundant or correlated data in order to reduce the overall size of sent data, thus decreasing the network traffic and energy consumption.

Most literature on sensor networks assumes *total aggregation*, i.e. data packets are assumed to have the same size and aggregation of two or more incoming packets at a node results in a single outgoing packet. Observe that even if this might be considered a simplistic assumption, it allows us to provide an upper bound on the expected benefits of data aggregation in terms of power consumption. We refer here to a selection of papers, focused on the algorithmic side of the problem [3, 6, 10, 9, 11, 12]. However, these papers mainly focus on empirical and technical aspects of the problem.

We concentrate on data aggregation in sensor networks under constraints on the *latency* of sensed events; this means that data should be communicated to the sinks within a specified time after being sensed. Preliminary results are given in [8, 15]. In both cases formal proofs of the performance are not provided.

Time synchronization, in the sense of the existence of a common clock for the nodes, may or may not be a requirement of the sensor network. Therefore, we will consider both the *synchronous model* and the *asynchronous model*.

Contributions of the paper. A sensor network is naturally represented by a graph whose nodes are the sensors and the arcs the wireless communication links. Data aggregation, latency constraints and energy savings, give rise to a large variety of graph optimization problems depending on the following issues.

- Transmission energy and time can be seen as functions of the size of the packet and the transmission arc. Typically, these are concave functions exhibiting economies of scale in the size of the packets sent.
- The latency may depend on the (types of) sensor data or on the sensor nodes.
- Sensor networks can be modelled as synchronous or asynchronous systems.
- Data is delivered to one or more sinks.
- The overlay routing paths connecting nodes to the sinks can be fixed a priori, (e.g. a tree or a chain) or may also be chosen as part of the optimization process.
- There might be several objective functions; the most natural ones are to minimize the maximum energy consumption over all nodes or to maximize the amount of sensed data arriving at the sinks with a given energy constraint.

By considering the above issues, we formulate the sensor problem in a combinatorial optimization setting, which allows us to derive, what we believe to be, the first results on worst case analysis for on-line algorithms on wireless sensor networks, as opposed to mainly empirical current results.

We concentrate here on a basic subclass of latency constrained data aggregation problems. We assume that transit times and transit costs, in terms of energy consumption, are functions of the arcs only, modeling the situation of total aggregation, while the objective is to minimize the maximum transit cost per node over all nodes. There is only one sink and the transmission paths from the nodes to the sink are unique, forming an intree with the sink as the root. The tree is a typical routing topology in sensor networks; see [4, 6, 10, 13, 14].

In Section 2 we formalize the problem; for a thorough understanding of the problem we have studied both the off-line and the on-line version of the problem, although the latter version is the relevant one in practice.

In Section 3 we show that the off-line problem is NP-hard and we give a 2-approximate algorithm. We remark that our approximate solution is based on a new rounding technique of the LP-relaxation of an Integer Linear Programming formulation of the problem, which might be useful for other applications.

In Section 4 we describe the distributed on-line problem, both in the synchronous and the asynchronous settings. Our main results are:

(a) *Distributed synchronous.* We present a $\Theta(\log U)$-competitive algorithm, where U is the ratio between the maximum and the minimum time that a packet can wait in its route toward the sink. We also show an $\Omega(\log U)$ lower bound, whence the proposed algorithm is best possible up to a multiplicative constant. (b) *Distributed asynchronous.* We give an $O(\delta \log U)$-competitive algorithm, where δ is the depth of the tree, which belongs to a class of algorithms for which we can prove a lower bound of $\Omega(\delta^{1-\epsilon})$ for any $\epsilon > 0$ on the competitive ratio.

Omitted proofs and more related results can be found in a full version [2] of this abstract.

Related results. In spirit [4] come closest to our paper. In [4] the authors consider optimization of TCP acknowledgement (ACK) in a multicasting tree. The problem their work addresses is a data aggregation problem. However, energy consumption is not an issue in this problem and latency is considered as a cost instead of a constraint, resulting in an objective of minimizing the sum of the total number of transmissions and the total latency of the messages.

In [5] the authors studied the optimal aggregation policy in a single-hop scenario (i.e. the graph is a star). Namely an *aggregator* performs a request and starts waiting for answers from a set of sources. The time for each source to return its data to the aggregator is independent and identically distributed according to a known distribution F. The main differences with our paper are that they assume that F is known, and they focus on a single-hop scenario.

2 The Sensor Problem Formalized

We study sensor networks $D = (V, A)$, which are *intrees* rooted at a *sink node* $s \in V$. Nodes represent sensors and arcs represent the possibility of transmission between two sensors. Given an arc $a \in A$ we denote its head and tail nodes by $\text{head}(a)$ and $\text{tail}(a)$, respectively.

Over time, n messages, $N := \{1, \ldots, n\}$, arrive at nodes and have to be sent to the sink. Message j arrives at its *release node* v_j at its *release date* r_j and must arrive at the sink via the unique $v_j - s$-path at or before its *due date* d_j. Thus, each message is completely defined by the triple (v_j, r_j, d_j). Unless otherwise stated we assume that messages are indexed by increasing due date, i.e., $d_1 \leq d_2 \leq \cdots \leq d_n$. We refer to $L_j := d_j - r_j$ as the *latency* of message j.

A *packet* is a set of messages which are sent simultaneously along an arc. More precisely, each initial message is sent as one packet. Recursively, two packets j and ℓ can be aggregated at a node v. The resulting packet has due date $d = \min\{d_j, d_\ell\}$. This definition naturally extends to the case of more packets aggregated together.

Transition of a message along an arc takes time and energy (cost). In this paper we assume that the transit time $\tau : A \to \mathbb{R}_{\geqslant 0}$ and transit cost $c : A \to \mathbb{R}_{\geqslant 0}$ are independent of packet size. We often refer to the *transit cost* of a node as the transit cost of its unique outgoing arc. This models the situation in which all messages have more or less the same size and where *total aggregation* is possible, as discussed in the introduction. For $v \in V$, let τ_v and c_v be, respectively, the total transit time and total transit cost on the path from v to s. For message j and node u on the path from v_j to s, we define *transit interval* $I_j(u)$ as the time interval during which message j can transit at node u: $I_j(u) := [r_j + \tau_{v_j} - \tau_u, d_j - \tau_u]$. In particular, $I_j(s) = [r'_j, d_j]$, where $r'_j := r_j + \tau_{v_j}$ is the *earliest possible arrival time* of j at s. We abbreviate I_j for $I_j(s)$ and call it the *arrival interval* of message j. We also write $|I|$ for the length of interval I; note that $|I_j(u)| = |I_j|$ for all j and for all u on the path from v_j to u.

Finally, we define $\delta := \max_v \tau_v$ as the depth of the network in terms of the transit time.

The objective of the sensor problem is to send all messages to the sink in such a way as to minimize the maximum transit cost per node, while satisfying the latency restrictions. Given that transit costs are independent of the size of packets sent, but linear in the number of packets sent, it is clearly advantageous to aggregate messages into packets at tail nodes of arcs.

3 The Off-Line Problem

We start by proving some properties of optimal off-line solutions.

Lemma 1. *There exists a minimum cost solution such that:*

(i) whenever two messages are present together at the same node, they stay together until they reach the sink;
(ii) a message never waits at an intermediate node, i.e., a node different from its release node and the sink;
(iii) the time when a packet of messages arrives at the sink is the earliest due date of any message in that packet.

Proof. (i): Repeatedly apply the argument that whenever two messages are together at the same node but split up afterwards, keeping the one arriving later at the sink with the other message does not increase cost.

(ii): Use (i) and repeatedly apply the following argument. Whenever a packet of messages arrives at an intermediate node and waits there, changing the solution by shifting this waiting time to the tail node of the incoming arc does not increase cost.

(iii): Follows similarly as (ii) by interpreting the time between the arrival of a packet at the sink and earliest due date as waiting time. □

Theorem 1. *The off-line sensor problem is strongly NP-hard.*

The proof of Theorem 1 involves a non-trivial reduction from the Satisfiability Problem and is deferred to the full version of the paper.

We give an ILP-formulation of the problem, based on Lemma 1, and show that rounding the optimal solution of the LP-relaxation yields a 2-approximation algorithm. For every message-arc pair $\{i, a\}$, we introduce a binary decision variable x_{ia}, which is set to 1 if and only if arc a is used by some message j which arrives at s at time d_i. We use the notation $j_{\min}$ for the smallest index i such that $d_i \geq r'_j$ and a_j for the first arc on the (unique) $v_j - s$- path.

$$\begin{aligned} &\min z \\ &\text{s.t. } z \geq c(a) \textstyle\sum_{i=1}^{n} x_{ia} \quad \forall a \in A, \\ &\quad \textstyle\sum_{i=j_{\min}}^{j} x_{ia_j} \geqslant 1 \quad \forall\, 1 \leq j \leq n, \\ &\quad x_{ia} \geqslant x_{ia'} \quad \forall\, 1 \leq i \leq n \;\forall\, a, a' \in A \text{ with } \mathrm{head}(a') = \mathrm{tail}(a), \\ &\quad x_{ia} \in \{0,1\} \quad \forall\, 1 \leq i \leq n \;\forall\, a \in A. \end{aligned} \tag{1}$$

The first set of constraints ensures that z is at least the transit cost of any node. The second set of constraints forces each message to leave its release node in time to reach the sink before its due date. By the third set of constraints a message does not wait at intermediate nodes.

In the following lemma we develop a tool for rounding the corresponding LP-relaxation, which is obtained by replacing the integrality constraints with non-negativity constraints $x_{ia} \geqslant 0$.

Lemma 2. *Let* $\alpha_1, \ldots, \alpha_n \in \mathbb{R}_{\geqslant 0}$ *and* $\beta_1, \ldots, \beta_n \in \{0, 1\}$ *with*

$$\textstyle\sum_{i=j}^{k} \alpha_i \geqslant 1 \quad \Longrightarrow \quad \sum_{i=j}^{k} \beta_i \geqslant 1 \quad \forall 1 \leq k \leq n \;\forall 1 \leq j \leq k. \tag{2}$$

By decreasing some of the β_i*'s from* 1 *to* 0*, one can enforce the inequality*

$$\textstyle\sum_{i=1}^{n} \beta_i \leqslant 2 \sum_{i=1}^{n} \alpha_i \tag{3}$$

while maintaining property (2)*. Moreover, this can be done in linear time.*

Proof. Consider the β_i's in order of increasing index. If $\beta_i = 1$, then round it down to 0, unless this yields a violation of (2). It is not difficult to see that this greedy algorithm can be implemented to run in linear time. It remains to be proven that inequality (3) holds for the resulting numbers $\beta_1, \ldots, \beta_n$.

For $h \in \{1, \ldots, n\}$, let $\bar{h} := \min\{i > h \mid \beta_i = 1\}$; if $\beta_i = 0$ for all $i > h$ or $h = n$, then $\bar{h} := n + 1$. Similarly, let $\underline{h} := \max\{i < h \mid \beta_i = 1\}$; if $\beta_i = 0$ for all $i < h$ or $h = 1$, then $\underline{h} := 0$. We prove the following generalization of (3):

$$\sum_{i=1}^{h} \beta_i \leqslant 2 \sum_{i=1}^{\bar{h}-1} \alpha_i \quad \forall\, 1 \leq h \leq n. \tag{4}$$

By contradiction, consider the smallest index h violating (4). Since h is chosen minimally, it must hold that $\beta_h = 1$; rounding β_h down to 0 would yield a violation of (2). In particular this would yield

$$\sum_{i=\underline{h}+1}^{\bar{h}-1} \alpha_i \geqslant 1 \tag{5}$$

while $\sum_{i=\underline{h}+1}^{\bar{h}-1} \beta_i = 0$. Notice that $\underline{h} \geqslant 1$, since, by choice of h,

$$\sum_{i=1}^{h} \beta_i > 2 \sum_{i=1}^{\bar{h}-1} \alpha_i \overset{(5)}{\geqslant} 2 \ .$$

Thus, $\beta_{\underline{h}} = \beta_h = 1$. We get a contradiction to the choice of h:

$$\sum_{i=1}^{h} \beta_i = \sum_{i=1}^{\underline{h}-1} \beta_i + 2 \overset{(4)}{\leqslant} 2 \sum_{i=1}^{\underline{h}-1} \alpha_i + 2 \overset{(5)}{\leqslant} 2 \sum_{i=1}^{\underline{h}-1} \alpha_i + 2 \sum_{i=\underline{h}+1}^{\bar{h}-1} \alpha_i \leqslant 2 \sum_{i=1}^{\bar{h}-1} \alpha_i \ .$$

The first inequality follows from (4) since $\overline{(\underline{h} - 1)} = \underline{h}$. □

Theorem 2. *There is a polynomial time 2-approximation algorithm for the sensor problem on intree $D = (V, A)$.*

Proof. We round optimal (fractional) solution (x, z) of the LP relaxation of (1) to an integral solution $(\bar{x}, \bar{z})$. Consider the arcs in order of non-decreasing distance from s. For arc a with $\text{head}(a) = s$, set $\hat{x}_{ia} = 1 \ \forall\ i = 1, \ldots, n$. Modify these values to $\bar{x}_{1a}, \ldots, \bar{x}_{na}$ by applying Lemma 2 to $x_{1a}, \ldots, x_{na}$ and $\hat{x}_{1a}, \ldots, \hat{x}_{na}$.

For an arc a' with larger distance to s, take the arc a with $\text{head}(a') = \text{tail}(a)$ and set $\hat{x}_{ia'} := \bar{x}_{ia}\ \forall i = 1, \ldots, n$. We also modify these values into $\bar{x}_{1a'}, \ldots, \bar{x}_{na'}$ by applying Lemma 2 to the values $x_{1a'}, \ldots, x_{na'}$ and $\hat{x}_{1a'}, \ldots, \hat{x}_{na'}$. Premise (2) of Lemma 2 is satisfied for $x_{1a'}, \ldots, x_{na'}$ and $\hat{x}_{1a'}, \ldots, \hat{x}_{na'}$ since (2) holds for $x_{1a}, \ldots, x_{na}$ and $\bar{x}_{1a}, \ldots, \bar{x}_{na}$ and since $x_{ia'} \leq x_{ia}$.

By construction, the final solution $(\bar{x}, \bar{z})$ is feasible if we choose $\bar{z} = 2z$. □

4 The Distributed On-Line Problem

We consider a class of *distributed on-line* models, in which nodes communicate independently of each other, while messages are released over time. Each node

is equipped with an algorithm, which determines at what times the node sends its packets to the next node on the path to the sink. The input of each node's algorithm at any time t is restricted to the packets that have been released at or forwarded from that node in the period $[0, t]$.

We assume that all nodes are equipped with a clock to measure the latency of messages. We distinguish two distributed on-line models: In the *synchronous* model all nodes are equipped with a *common clock*, i.e. the times indicated at all clocks are identical. A common clock may facilitate synchronization of actions in various nodes. In the *asynchronous* model there is no such common clock; still, the duration of the time unit is assumed to be the same for all nodes.

We also assume in both models that each node v knows its total transit time τ_v to the sink. Moreover, for the asynchronous model we assume that all transit times $\tau(a)$ are equal, and without loss of generality we set $\tau(a) = 1\ \forall a \in A$.

4.1 The Synchronous Model

For the synchronous model we propose an algorithm based on the following simple result, the obvious proof of which we omit.

Lemma 3. *Given any interval* $[a, b]$, $a, b \in \mathbb{N}$, *let* $i^* = \max\{i \in \mathbb{N} \mid \exists k \in \mathbb{N} : k2^i \in [a, b]\}$. *Then* k^* *for which* $k^*2^{i^*} \in [a, b]$ *is odd and unique. Also,* $i^* \geq \lfloor \log_2(b - a) \rfloor$. *We use notation* $t(a, b) = k^*2^{i^*}$. □

> **Algorithm:CommonClock** (CC) Message j is sent from v_j at time $t(r'_j, d_j) - \tau_{v_j}$ to arrive at s at time $t(r'_j, d_j)$ unless some other message (packet) passes v_j in the interval $[r_j, t(r'_j, d_j) - \tau_{v_j}]$, in which case j is aggregated and the packet is forwarded directly.

First we derive a bound on the competitive ratio of CC for instances in which the arrival intervals I_j differ by at most a factor 2 in length.

Lemma 4. *If there exists an* $i \in \mathbb{N}$ *such that* $2^{i-1} < |I_j| \leq 2^i$ *for all messages* j, *then* CC *has a competitive ratio of at most* 3.

Proof. Assume that in an optimal solution packets arrive at s at times $t_1 < \cdots < t_\ell$. Let N_h^* be the packet arriving at t_h at s. Since $t_h \in I_j\ \forall j \in N_h^*$ and $|I_j| \leq 2^i\ \forall j$, we have $I_j \subset [t_h - 2^i, t_h + 2^i] =: I\ \forall j \in N_h^*$, and $|I| = 2 \cdot 2^i$. If $t_h = k2^i$ then in the CC-solution all messages in N_h^* may arrive at s at times t_h, $t_h - 2^i$ or $t_h + 2^i$. If $t_h \neq k2^i$ then I contains two different multiples of 2^i, say $k2^i$ and $(k + 1)2^i$, such that $k2^i < t_h < (k+1)2^i$. In this case, since $|I_j| > 2^{i-1}\ \forall j$, we have $\forall j \in N_h^*$ that $I_j \cap \{k2^i, k2^i + 2^{i-1}, (k+1)2^i\} \neq \emptyset$. Lemma 3 implies that in a CC-solution every message $j \in N_h^*$ arrives at s at one of $\{k2^i, k2^i + 2^{i-1}, (k + 1)2^i\}$. Hence, $\forall h = 1, \ldots, \ell$, all messages in N_h^* arrive at s at at most 3 distinct time instants in the CC-solution. CC does not delay messages at intermediate nodes. This implies that the arcs used by messages in N_h^* are traversed by these messages at most 3 times in the CC-solution, proving the lemma. □

Theorem 3. CC *is* $\Theta(\max\{\log U, 1\})$*-competitive with* $U = \frac{\max_j |I_j|}{\max\{1, \min_j |I_j|\}}$.

Proof. For each $i \in \mathbb{N}$ with $\log(\max\{1, \min_j |I_j|\}) \leq i \leq \lceil \log(\max_j |I_j|) \rceil$, CC sends the messages in $N_i := \{j \in N \mid 2^{i-1} < |I_j| \leq 2^i\}$, at a cost of no more than 3 times the optimum, by Lemma 4. This proves $O(\max\{\log U, 1\})$-competitiveness if $\min_j |I_j| \geq 1$. In case $\min_j |I_j| = 0$ we observe that restricted to the class of messages $N_0 = \{j \in N \mid |I_j| = 0\}$ CC's cost equals the optimal cost, because there is no choice for these messages.

To prove $\Omega(\log U)$ consider a chain of 2^{n+1} nodes $u_1, \ldots, u_{2^{n+1}} = s$ for some $n \in \mathbb{N}$. Take $\tau(a) = 1$ and $c(a) = 1$ $\forall a$. For $j = 1, \ldots, n$, $v_j = u_{2^j}$, $r_j = 0$, and $d_j = 2^{n+1} - 1$. Hence $r'_j = 2^{n+1} - 2^j = k2^j$ for some odd $k \in \mathbb{N}$ and $|I_j| = 2^j - 1$. Therefore, CC makes each message j arrive at s at time r'_j, no two messages are aggregated, and the cost is $\sum_{j=1}^{n}(2^{n+1} - 2^j) = (n-1)2^{n+1} + 2$. In an optimal solution all messages are aggregated into a single packet arriving at s at time $2^{n+1} - 1$ at a cost of $2^{n+1} - 2$. Notice that $U = 2^n - 1$ in this case. □

The following theorem shows that CC is best possible (up to a multiplicative constant).

Theorem 4. *Any deterministic synchronous algorithm is* $\Omega(\log U)$*-competitive.*

Proof. Consider an intree of depth $\delta = 2^{n+1}$ with n the number of messages, and where each node, except the leaves, has indegree n. We assume $\tau(a) = 1$ for all $a \in A$. For any on-line algorithm we will construct an adversarial sequence of n messages all with latency $L = \delta$, such that there exists a node at which the adversary can aggregate all messages in a single packet, but at which none of them is aggregated by the on-line algorithm. Using a similar argument as in the proof of Lemma 1 (i) the fact that all messages can be aggregated in a single packet implies that there exists a solution such that every node sends at most one packet, hence the cost of the adversarial solution is 1, whereas the cost of the on-line algorithm is n.

Fix any on-line algorithm. Given an instance of the problem, let $W_j(u)$ be the time interval message j spent at node u by application of the algorithm, i.e. the waiting time interval of message j on u. We denote its length by $|W_j(u)|$. Note that $\sum_u |W_j(u)| \leq |I_j|$ for each message j. We notice that the waiting time of a message in a node can be influenced by the other messages that are present at that node or have passed that node before. Since the algorithms are distributed the waiting time of a message in a node is not influenced by any message that will pass the node in the future.

The adversary chooses the source node v_j with total transit time $\tau_{v_j} := \delta - 2^j$ from s, for $j = 1, \ldots, n$, so that $|I_j| = 2^j$. Thus, $U = 2^{n-1} = \delta/4$. The choice of the exact position of v_j and the release time r_j is made sequentially and, to facilitate the exposition, described in a backward way starting with message n. The proof follows rather directly from the following claim.

Claim. For any set of messages $\{k, \ldots, n\}$ the adversary can maintain the properties:

(i) all messages in $\{k, \ldots, n\}$ pass a path p_k with 2^k nodes;
(ii) $I_k(u) = \bigcap_{j \geq k} I_j(u)\ \forall u \in p_k$;
(iii) if $k < n$, then $W_{k+1}(u) \bigcap I_k(u) = \emptyset\ \forall u \in p_k$;
(iv) if $k < n$, then $W_i(u) \bigcap W_j(u) = \emptyset\ \forall u \in p_k$, $i = k, \ldots, n$, $j > i$.

We notice that for any message j and any node u on the path from v_j to s, $W_j(u)$ may have length 0 but is never empty; it contains at least the departure time of message j from node u.

Note that properties (i) and (ii) for $k = 1$ imply that all messages can indeed be aggregated into one packet, hence as argued above, the adversarial solution has a cost of 1. Properties (iv) and (i) for $k = 1$ imply that the on-line algorithm sends all messages separately over a common path with 2 nodes, yielding a cost of n. This proves the theorem.

We prove the claim by induction. The basis of the induction, $k = n$, is trivially verified. Suppose the claim holds for message set $\{k, \ldots, n\}$ and p_k is the path between nodes $\overline{v}$ and $\hat{v}$. We partition p_k into two sub-paths $\overline{p}$ and $\hat{p}$ consisting of 2^{k-1} nodes each, such that $\overline{v} \in \overline{p}$ and $\hat{v} \in \hat{p}$. We denote the last node of $\overline{p}$ by $\overline{u}$ and the first node of $\hat{p}$ by $\hat{u}$. We distinguish two cases with respect to the waiting times the algorithm has selected for message k in the nodes on p_k.

Case a: $\sum_{u \in \overline{p}} |W_k(u)| \geq (1/2)|I_k|$. The adversary chooses v_{k-1} with total transit time $\tau_{v_{k-1}} = \delta - 2^{k-1}$ such that its path to s traverses $\hat{p}$ but not $\overline{p}$. More precisely, we ensure that the first node message $k-1$ has in common with any other message is $\hat{u}$. This is always possible, since the node degree is n. This choice immediately makes that setting $p_{k-1} = \hat{p}$ satisfies property (i). The release time of $k-1$ is chosen so that $I_{k-1}(\hat{u})$ and $I_k(\hat{u})$ start at the same time, implying that $I_{k-1}(u)$ and $I_k(u)$ start at the same time for every $u \in \hat{p}$. Since $|I_{k-1}(u)| = |I_k(u)|/2$ we have $I_{k-1}(u) \subset I_k(u)$ for all $u \in \hat{p}$, whence property (ii) follows by induction.

Note that, as we consider distributed algorithms, message $k-1$ does not influence the waiting time of $j, j > k-1$, on $\overline{p}$ as $\hat{u}$ is the first node which both j and $k-1$ traverse. In particular, $W_k(u), \forall u \in \overline{p}$ is not influenced by $k-1$.

Now, the equal starting times of $I_{k-1}(\hat{u})$ and $I_k(\hat{u})$ together with $\sum_{u \in \overline{p}} |W_k(u)| \geq (1/2)|I_k|$ and $|I_{k-1}(\hat{u})| = |I_k(\hat{u})|/2$ imply that k will not reach $\hat{u}$ before interval $I_{k-1}(\hat{u})$ ends. This, together with the consideration above, implies property (iii).

To prove (iv), note that by induction it is sufficient to prove that $W_{k-1}(u) \cap W_j(u) = \emptyset\, \forall j > k-1\ \forall u \in \hat{p}$. Since, as just proved, $W_k(u) \cap I_{k-1}(u) = \emptyset\ \forall u \in \hat{p}$ we have $W_{k-1}(u) \cap W_k(u) = \emptyset\ \forall u \in \hat{p}$. We have by induction that, for $j > k$, $W_j(u) \cap I_{j-1}(u) = \emptyset\ \forall u \in \hat{p}$ and we just proved that $I_{k-1}(u) \subset I_{j-1}(u) \subset I_j(u)\ \forall u \in \hat{p}$, which together imply $W_{k-1}(u) \cap W_j(u) = \emptyset\ \forall j > k\ \forall u \in \hat{p}$.

Case b: $\sum_{u \in \overline{p}} |W_k(u)| < (1/2)|I_k|$. As in the previous case, the adversary chooses v_{k-1} with total transit time $\tau_{v_{k-1}} = \delta - 2^{k-1}$ such that its path to s traverses $\overline{p}$ (therefore also $\hat{p}$) but does not intersect any of the paths used by

messages $\{k, \ldots, n\}$ before it reaches $\overline{p}$ in $\overline{v}$. Again, this is always possible since the indegree of each node is n. Hence, choosing $p_{k-1} = \overline{p}$ satisfies property (i). The release time of $k-1$ is chosen so that $I_{k-1}(\overline{v})$ and $I_k(\overline{v})$ end at the same time, implying that $I_{k-1}(u)$ and $I_k(u)$ end at the same time for every $u \in \overline{p}$. Since $|I_{k-1}(u)| = |I_k(u)|/2$ we have $I_{k-1}(u) \subset I_k(u)$ for all $u \in \overline{p}$, whence property (ii) follows by induction.

The equal ending times of $I_{k-1}(\overline{u})$ and $I_k(\overline{u})$ together with $\sum_{u \in \overline{p}} |W_k(u)| < 1/2|I_k|$ and $|I_{k-1}(\overline{u})| = |I_k(\overline{u})|/2$ imply that k has left $\overline{u}$ before $I_{k-1}(\overline{u})$ begins, implying property (iii). Indeed, this gives $W_{k-1}(u) \cap W_k(u) = \emptyset, \forall u \in \overline{p}$. It also implies that $k-1$ could not influence the waiting time of k on $\overline{p}$.

The proof of (iv) follows the very same lines as in Case a, with the difference that we now refer to nodes in $\overline{p}$ instead of $\hat{p}$. □

Since in the proof $U = \delta/4$ we also have the following lower bound on the competitive ratio of any deterministic synchronous algorithm.

Theorem 5. *Any deterministic synchronous algorithm is $\Omega(\log \delta)$-competitive.* □

4.2 The Asynchronous Model

In the asynchronous model nodes are equipped with a clock and a distributed algorithm. All clocks have the same time unit, but neither the time nor the start of a new time unit on clocks is synchronized. We assume that $\tau(a) = 1$ for all a, such that τ_{v_j} is equal to the number of nodes on the $v_j - s$-path.

We propose algorithm Spread Latency (SL) for this model, which divides the latency minus transmission time of each message j equally over the nodes on the $v_j - s$-path: at each node of this path the message is assigned a waiting time of $(L_j - \tau_{v_j})/\tau_{v_j}$ time units. As soon as messages appear simultaneously at the same node they get aggregated into a packet, which is sent over the outgoing arc as soon as the waiting time of at least one of its messages at that node has passed. In this way, no message is delayed due to aggregation and thus the algorithm yields a feasible solution.

Let, as in the previous subsection, $U := \frac{\max_j |I_j|}{\max\{1, \min_j |I_j|\}} = \frac{\max_j (L_j - \tau_{v_j})}{\max\{1, \min_j (L_j - \tau_{v_j})\}}$.

Theorem 6. *The algorithm* SL *is $O(\delta \max\{\log U, 1\})$-competitive.*

Proof. We prove that for all $a \in A$ the number of packets SL sends through a is at most $O(\delta \max\{\log U, 1\})$ times that number in an optimal solution. This proves the theorem.

Let $\lambda := \max\{1, \min_j (L_j - \tau_{v_j})\}$. Consider a packet P of messages sent by an optimal solution through (u, v) at t. To bound the number of packets sent by SL that contain at least one message from P, define $P_k := \{j \in P \mid 2^{k-1}\lambda \leq L_j - \tau_{v_j} < 2^k \lambda\}$, for $k = 1, \ldots, \lceil \log U \rceil$. We charge any sent packet to the message that caused the packet to be sent due to its waiting time being over. It suffices to prove that the number of packets charged to messages in P_k is $O(\delta)$.

Since the waiting time of messages $j \in P_k$ at node u is at least $2^{k-1}\lambda/\delta$, the delay between any two packets that are charged to messages in P_k is at least $2^{k-1}\lambda/\delta$. Since the optimal solution sends packet P at t through (u, v), we get

$t \in I_j(u)\ \forall j \in P$ and thus $I_j(u) \subseteq [t - 2^k\lambda, t + 2^k\lambda]\ \forall j \in P_k$. Thus, the number of packets charged to messages in P_k is at most $2 \cdot 2^k\lambda/(2^{k-1}\lambda/\delta) = 4\delta$. □

SL determines the waiting time of each message at the nodes it traverses independently of all other messages. We call such an algorithm a WI-algorithm. To be precise, in a WI-algorithm node v determines the waiting time of message j based only on the message characteristics (v_j, r_j, d_j), transit time to the sink τ_v and clock time. The following lower bound shows that the competitive ratio of SL cannot be beaten by more than a factor $\max\{\log U, 1\}$ by any other WI-algorithm. In the derivation of the lower bound we restrict to WI-algorithms that employ the same algorithm in all nodes with the same transit time to s. This is not a severe restriction, given that transit time to s is the only information about the network that a node has.

Theorem 7. *Any deterministic asynchronous WI-algorithm is $\Omega(\delta^{1-\epsilon})$- competitive for any $\epsilon > 0$.*

Proof. Consider a binary intree with root s and all leaves at distance δ from s. Let $0 \leq \lambda < 1$ be such that $\delta^{1-\lambda} \geq 3$. An adversary releases message 1 with latency L at time r_1 in a leaf v_1. Notice that there are at most δ^λ nodes where the waiting time is at least $(L - \tau_{v_1})/\delta^\lambda$. Hence, the $v_1 - s$ path contains a sub-path consisting of at least $\delta^{1-\lambda} - 2$ nodes where in each node message 1 waits less than $(L - \tau_{v_1})/\delta^\lambda$. Choose such a sub-path and let u be the node on this sub-path closest to s.

Let V' be the set of leaves of the subtree with root u and depth $\delta^{1-\lambda} - 2$. Then $|V'| \geq 2^{\delta^{1-\lambda}-2} \geq \delta^\lambda/4$ for any fixed $\lambda \in [0, 1)$ and δ large enough. The adversary releases messages $j = 2, \ldots, \delta^\lambda/4$ with latency L at times $r_j = r_1 + j(L - \tau_{v_j})/\delta^\lambda$ in leaf v_j, such that each $v_j - s$ path passes through a different vertex of V'. Because $\tau_{v_j} = \tau_{v_1}\ \forall j$ and we assumed that any WI-algorithm applies the same algorithm in nodes at equal distance, all messages are sent non-aggregated to and from u, whereas they are aggregated as early as possible in an optimal solution, in particular at u. □

The lower bound does not hold for arbitrary algorithms as a node may adjust the waiting time of subsequent messages that traverse that node. The following theorem shows that the lower bound remains $\Omega(\delta^{1-\epsilon})$ if release nodes do not delay subsequent messages longer than preceding messages.

Theorem 8. *Any asynchronous WI-algorithm for which the waiting time of message j at its release node is at most $\frac{L-\tau_{v_j}}{K}$ is $\Omega(K)$-competitive.*

Proof. Consider a chain which consists of two nodes v and s. We assume constant latency L for each message. The adversary releases $K - 1$ messages with an interval of $(L - \tau_{v_j})/(K - 1)$ at v. Since the waiting time of message j at v is at most $(L - \tau_{v_j})/K$, none of these messages are aggregated in the on-line solution, whereas they are all aggregated in one packet in an optimal solution. □

The theorem proves that SL is $\Omega(\delta)$-competitive. For arbitrary asynchronous algorithms we do not have any better lower bound than the one in Theorem 5.

In a full version of the paper [2] we design an algorithm with improved competitive ratio of $O(\log^3 \delta)$ for the asynchronous problem on a chain with s at one of its ends.

References

1. I. Akyildiz, W. Su, Y. Sanakarasubramaniam, and E. Cayirci. Wireless sensor networks: A survey. *Computer Networks Journal*, 38(4):393–422, 2002.
2. L. Becchetti, P. Korteweg, A. Marchetti-Spaccamela, M. Skutella, L. Stougie, and A. Vitaletti. *Latency Constrained Aggregation in Sensor Networks.* SPOR-report 2006-08, TU Eindhoven, www.win.tue.nl/bs/spor, 2006.
3. A. Boulis, S. Ganeriwal, and M. B. Srivastava. Aggregation in sensor networks: an energy - accuracy tradeoff. *Ad-hoc Networks Journal*, 1(2-3):317–331, 2003.
4. C. Brito, E. Koutsoupias, and S. Vaya. Competitive analysis of organization networks or multicast acknowledgement: how much to wait? In *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms (SODA)*, pages 627–635, 2004.
5. A. Z. Broder and M. Mitzenmacher. Optimal plans for aggregation. In *PODC*, pages 144–152, 2002.
6. A. Goel and D. Estrin. Simultaneous optimization for concave costs: single sink aggregation or single source buy-at-bulk. In *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms (SODA)*, pages 499–505, 2003.
7. W. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy efficient communication protocols for wireless microsensor networks. In *Proceedings of Hawaiian International Conference on Systems Science*, pages 3005–3014, 2000.
8. F. Hu, X. Cao, and C. May. Optimized scheduling for data aggregation in wireless sensor networks. In *International Conference on Information Technology Coding and Computing (ITCC)*, pages 557–561, 2005.
9. C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann. Impact of network density on data aggregation in wireless sensor networks. In *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS)*, pages 414–458, 2002.
10. C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Mobile Computing and Networking*, pages 56–67, 2000.
11. K. Kalpakis, K. Dasgupta, and P. Namjoshi. Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks. *Computer Networks*, 42(6):697–716, 2003.
12. S. Lindsey and C. S. Raghavendra. Pegasis: Power-efficient gathering in sensor information systems. In *Proceedings of IEEE Aerospace Conference*, pages 1125–1130, 2000.
13. S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tag: A tiny aggregation service for ad-hoc sensor networks. In *Proceedings of the 5th ACM Symposium on Operating System Design and Implementation (OSDI)*, pages 131–146, 2002.
14. S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tinydb: an acquisitional query processing system for sensor networks. *ACM Transactions on Database Systems (TODS)*, 30(1):122–173, 2005.
15. W. Yuan, V. S. Krishnamurthy, and S. K. Tripathi. Synchronization of multiple levels of data fusion in wireless sensor networks. In *Proceedings of IEEE Globecom*, pages 221–225, 2003.