Gathering with Minimum Delay in Tree Sensor Networks

Jean–Claude Bermond[1] * and Luisa Gargano[2] ** and Adele A. Rescigno[2]

[1] MASCOTTE, joint project CNRS-INRIA-UNSA, Sophia-Antipolis, France
[2] Dip. di Informatica ed Applicazioni, Universitá di Salerno, 84084 Fisciano, Italy

Abstract. Data gathering is a fundamental operation in wireless sensor networks in which data packets generated at sensor nodes are to be collected at a base station. In this paper we suppose that each sensor is equipped with an half–duplex interface; hence, a node cannot receive and transmit at the same time. Moreover, each node is equipped with omnidirectional antennas allowing the transmission over distance R. The network is a multi-hop wireless network and the time is slotted so that one–hop transmission of one data item consumes one time slot. We model the network with a graph where the vertices represent the nodes and two nodes are connected if they are in the transmission/interference range of each other. Due to interferences a collision happens at a node if two or more of its neighbors try to transmit at the same time. Furthermore we suppose that an intermediate node should forward a message as soon as it receives it. We give an optimal collision free gathering schedule for tree networks whenever each node has at least one data packet to send.

1 Introduction

A wireless sensor network is a multi-hop wireless network formed by a large number of low-cost sensor nodes, each equipped with a sensor, a processor, a radio, and a battery. Due to the many advantages they offer – e.g. low cost, small size, and wireless data transfer – wireless sensor networks become attractive to a vast variety of applications like space exploration, battlefield surveillance, environment observation, and health monitoring.

A basic activity in a sensor network is the systematic gathering of the sensed data at a base station for further processing. A key challenge in such operation is due to the physical limits of the sensor nodes, which have limited power and un–replenishable batteries. It is then important to bound the energy consumption of data dissemination [10, 18, 24]. However, an other important factor to consider in data gathering applications is the *latency* of the information dissemination process. Indeed, the data collected by a node can frequently change thus making essential that they are received by the base station as soon as it is possible without being delayed by collisions [26].

* Partially supported by the CRC CORSO with France Telecom, by the European FET project AEOLUS

** Work partially done while visiting INRIA at Sophia-Antipolis

Another application, which motivates this work, concerns the use in telecommunications networks a problem asked by FRANCE TELECOM about "how to provide Internet connection to a village" (see [6]). Here we are given a set of communication devices placed in houses in a village (for instance, network interfaces that connect computers to the Internet). They require access to a gateway (for instance, a satellite antenna) to send and receive data through a multi-hop wireless network. Therefore, this problem is the same as data collection in sensor network. Here the main objective is to minimize the delay.

In this paper, we will study optimal–time data gathering in tree networks.

1.1 Network model

We adopt the network model considered in [3, 12, 17]. In this model each node is equipped with an half–duplex interface, hence,

i) *a node cannot receive and transmit at the same time.*

Moreover, each node is equipped with omni directional antennas allowing transmission over a distance R. This implies that for any given node in the network, we can individuate its neighbors as those nodes within distance R from it, that is, within its transmission/interference range. In this model,

ii) *a* collision *happens at a node x if two or more of its neighbors try to transmit at the same time.*

However, simultaneous transmissions among pair of nodes can successfully occur whenever conditions i) and ii) of the above interference model are respected. The time is slotted so that one–hop transmission of one data item consumes one time slot; the network is assumed to be synchronous. Moreover, following [12, 15, 26] and contrarily to [3, 17], we assume that no buffering is done at intermediate nodes, that is each node forwards a message as soon as it receives it. Finally, it is assumed that the only traffic in the network is due to data to be collected, thus data transmissions can be completely scheduled.

Summarizing, the network can be represented by means of a direct graph $G = (V, A)$ where V represents the sensors (devices) nodes and A the set of possible calls; i.e. an arc $(u, v) \in A$ if v is in the transmission/interference range of u. Throughout this paper we assume that all nodes have the same transmission range, hence the graph G is a directed symmetric graph, e.g., $(u, v) \in A$ if and only if $(v, u) \in A$. The fact that there is no collision can be expressed by the fact that two calls (u, v) and (u', v') are compatible (can be done in the same time slot) iff $d(u, v') \geq 2$ and $d(u', v) \geq 2$ (i.e., both u and v', and u' and v have not to be neighbors, by ii)).

The collision–free data gathering problem can be then stated as follows [26].

Data Gathering. *Given a graph $G = (V, A)$ and a base station (BS) s, for each $v \in V - \{s\}$, schedule the multi-hop transmission of the data items sensed at v to s so that the whole process is collision–free, and the time when the last data is received by s is minimized.*

We will actually study the related one–to-all personalized broadcast problem in which the BS wants to communicate different data items to each other node in the network.

One–to-all personalized broadcast: *Given a graph G and a BS s, for each node $v \neq s$, schedule the multi-hop transmission from s to v of the data items destined to v so that the whole process is collision–free, and the time when the last data item is received at the corresponding destination node is minimized.*

Solving the above dissemination problem is equivalent to solve data gathering in sensor networks. Indeed, let T denote the delay, that is, the largest time–slot used by a dissemination algorithm; a gathering schedule with delay T consists in scheduling a transmission from node y to x during slot t iff the broadcasting algorithm schedules a transmission from node x to y during slot $T - t + 1$, for any t with $1 \leq t \leq T$.

It should be noticed that our algorithms are centralized requiring the BS perform a distinct topology learning phase and schedule broadcasting. When requirements are more stringent, these algorithms may no longer be practical. However, they still continue to provide a lower bound on the data collection time of any given collection schedule.

1.2 Related work

Much effort has been devoted to the study of efficient data gathering algorithms taking into consideration various aspects of sensor networks [8]. The problem of minimizing the delay of the gathering process has been recently recognized and studied. The authors of [12] first afford such a problem; they use the same model for sensor networks adopted in this paper. The main difference with our work is that [12] mainly deals with the case when nodes are equipped with directional antennas, that is, only the designed neighbor of a transmitting node receives the signal while its other neighbors can safely receive from different nodes. Under this assumption, [12] gives optimal gathering schedules for trees. An optimal algorithm for general networks has been presented in [15] in the case that each node has one packet of sensed data to deliver.
The work in [26] also deals with the latency of data gathering under the assumption of unidirectional antennas; the difference with [12] is the assumption of the possibility to have multiple channels between adjacent nodes. By adopting this model an approximation algorithm with performance ratio 2 is obtained.

Fast gathering with omnidirectional antennas is considered in [1, 3–5, 7], under the assumption of possibly different transmission and interference ranges, that is, when a node transmits, all the nodes within a fixed distance d_T in the graph can receive while nodes within distance d_I ($d_I \geq d_T$) cannot listen to other transmissions due to interference (in our paper $d_I = d_T$). Lower bounds on the time to gather and NP-hardness proofs are given in [3]; an approximation algorithm with approximation factor 4 is also presented. Paper [7] presents an on–line gathering algorithm under the described model.
The case where $d_T = 1$ and where each node has one packet to transmit is solved

for the line in [1], for the uniform grids in [4] and for trees when furthermore $d_I = 1$ in [5]. All the above papers allow buffering at intermediate nodes.

Several papers deal with the problem of maximize the lifetime of the network through topology aware placement [10, 13], data aggregation [16, 19–21], or efficient data flow [11, 18, 23]. Papers [9, 22, 25] consider the minimization of the gathering delay in conjunction with the energy spent to complete the process.

1.3 Paper Overview.

We consider the model introduced in Section 1.1 and give optimal gathering schedules in case the graph modeling the network is a line or a tree.

In Section 3 we shortly illustrate an optimal algorithm in case G is a line with the BS s as one of its endpoints. The result was first presented in [12]; we report it in our settings as a starting point for our result on trees.

In Section 4 we give an optimal algorithm in case the graph is a tree T with *one data item at each node*, apart the source s.
We look at T as rooted at s and denote by $T_1, T_2, \cdots, T_m$ the subtrees of T rooted at the sons of s.

Definition 1 *For each $i = 1, \ldots, m$, we denote by:*
- *s_i the son of s which is the root of T_i and is at level 1 in T_i,*
- *α_i the number of nodes at level 2 in T_i,*
- *β_i the number of nodes at level 3 or more in T_i.*

Moreover, we define the shade of subtree T_i, *for $1 \leq i \leq m$, as $\tau_i = 1 + 2\alpha_i + 3\beta_i$.*

Let $|T_i|$ represent the number of nodes in the subtree T_i, for $i = 1, \ldots, m$.

Definition 2 *Given $i, j = 1, \ldots, m$ with $i \neq j$, we say that*
- *$T_i \prec T_j$ if either $\tau_i > \tau_j$ or $\tau_i = \tau_j$ and $|T_i| > |T_j|$,*
- *$T_i = T_j$ if $\tau_i = \tau_j$ and $|T_i| = |T_j|$.(they are not necessarily isomorphic)*

Definition 3 *Assume that $T_1 \preceq T_2 \preceq \ldots \preceq T_m$. Define*

$$\epsilon_T = \begin{cases} 1 & \text{if } T_1 = T_2 \\ 0 & \text{otherwise} \end{cases} \qquad \text{and} \qquad \Delta_{i,j} = |T_i| + |T_j| + \beta_i - 1,$$

for each $i, j = 1, \ldots, m$, with $i \neq j$.

Theorem 1 *Suppose each node of a tree T has one data item and let n denote the number of vertices of T. Assuming that $T_1 \preceq T_2 \preceq \ldots \preceq T_m$, we have that the optimal gathering time is*

$$\mathcal{T}^*(T) = \max\{n - 1,\ \tau_1 + \epsilon_T,\ \Delta_{1,2},\ \Delta_{2,1},\ \Delta_{1,3}\}.$$

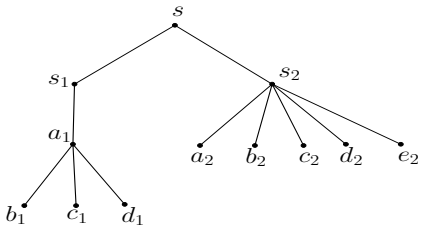

Figure 1.

Example 1. Let T be the tree in Fig.1 with BS s. We have: $m = 2$, $|T_1| = 5$, $|T_2| = 6$, $\alpha_1 = 1$, $\beta_1 = 3$, $\tau_1 = 12$, $\alpha_2 = 5$, $\beta_2 = 0$, $\tau_2 = 11$, $\epsilon_T = 0$, $\Delta_{1,2} = 13$, and $\Delta_{2,1} = 10$. Hence, $T_1 \prec T_2$ and, by Theorem 1, $\mathcal{T}^*(T) = \Delta_{1,2} = 13$.

Due to space limits some proofs are omitted from this extended abstract; a full version is in [2].

2 Mathematical Formulation

We now formally formulate the one-to-all personalized broadcast problem. Let $G = (V, A)$ be the directed symmetric graph that is obtained by replacing each edge connecting two nodes u and v of the network with two directed arcs (u, v) and (v, u). Furthermore, let $s \in V$ be a special node that will be called the *source*.

Each node $v \in V - \{s\}$ is associated with an integer weight $w(v) \geq 0$ that represents the number of data items destined to node v. The set $\mathbf{w} = \{w(v) \mid v \in V - \{s\}\}$ represents the set of the weights of the nodes in V.

We need to schedule (time–label) the transmissions in order to create $w(v)$ collision–free routes from s to node v, for each $v \in V - \{s\}$.

Definition 4 *Let $\mathbf{p} = (u_0, \cdots, u_h)$ be a path in G. An* increasing labeling L of $\mathbf{p}$ *is an assignment of integers, $L_{\mathbf{p}}(u_0, u_1) \ldots, L_{\mathbf{p}}(u_{h-1}, u_h)$, to the arcs of $\mathbf{p}$ such that for $j = 1, \ldots, h-1$.*

$$L_{\mathbf{p}}(u_j, u_{j+1}) = L_{\mathbf{p}}(u_{j-1}, u_j) + 1$$

The labeling is called t-increasing*, for some integer $t \geq 1$, if it is increasing and $L_{\mathbf{p}}(u_0, u_1) = t$.*

Consider any set $\mathcal{P}$ of paths in G from s to (not necessarily pairwise distinct) nodes in $V - \{s\}$ together with the labellings $L_{\mathbf{p}}$, for $\mathbf{p} \in \mathcal{P}$. Notice that any arc $a \in A$ can belong to any number of paths in $\mathcal{P}$.

Definition 5 *The labeling induced by $\mathcal{P}$ on the arcs of G consists, for each $(u, v) \in A$ of the multisets*

$$L(u, v) = \{L_{\mathbf{p}}(u, v) \mid \mathbf{p} \in \mathcal{P}\}.$$

Let $N(u)$ be the set of neighbors of u in G, that is, $N(u) = \{x \mid (u, x) \in A\} = \{x \mid (x, u) \in A\}$.

Definition 6 *The labeling L induced by $\mathcal{P}$ on the arcs of G is called* strictly collision–free (SCF) *if L is increasing and, for each $(u, v) \in A$ it holds:*

- *$L(u, v)$ is a set (e.g, any integer has at most one occurrence in $L(u, v)$),*
- *$L(u, v) \cap L(w, u) = \emptyset$, for each $w \in N(u)$,*
- *$L(u, v) \cap L(w, z) = \emptyset$, for each $w \in N(v) \cup \{v\}$, $z \in N(w)$.*

Definition 7 *An* instance of SCF labeling *is a triple $\langle G, \mathbf{w}, s\rangle$ where G is the graph, s is the source, and $\mathbf{w}$ is the set of weights of the nodes in G.*
A feasible solution *for $\langle G, \mathbf{w}, s\rangle$ is a pair $(\mathcal{P}, L)$ where:*

– *$\mathcal{P}$ is a set of $w(v)$ paths (not necessarily distinct) from s to v in G, for each $v \in V - \{s\}$;*
– *L is a SCF–labeling induced by $\mathcal{P}$.*

An optimal solution *$(\mathcal{P}^*, L^*)$ is a feasible solution minimizing the largest label given to any arc of G.*
The value attained by the optimal solution $(\mathcal{P}^*, L^*)$ for $\langle G, \mathbf{w}, s\rangle$ is denoted by $\mathcal{T}^*(\langle G, \mathbf{w}, s\rangle)$ (or simply by $\mathcal{T}^*(G)$ when $\mathbf{w}$ and s are clear from the context).

Example 2. In the tree T of Fig.1, let $w(u) = 1$ for each $u \neq s$. A feasible solution for $\langle T, \mathbf{w}, s\rangle$ is the pair $(\mathcal{P}, L)$ where $\mathcal{P} = \{\mathbf{p}_u \mid \mathbf{p}_u$ is the unique path from s to u in T, $u \neq s\}$ and the SCF labeling L is such that each path $\mathbf{p}_u$ is labeled with a t_u-increasing labeling as follows: $t_{b_1} = 1$, $t_{a_2} = 2$, $t_{b_2} = 4$, $t_{c_1} = 5$, $t_{c_2} = 6$, $t_{d_1} = 8$, $t_{s_2} = 9$, $t_{d_2} = 10$, $t_{a_1} = 11$, $t_{e_2} = 12$, $t_{s_1} = 13$.
As an example, we have
$\mathbf{p}_{b_1} = (s, s_1, a_1, b_1)$, with $L_{\mathbf{p}_{b_1}}(s, s_1) = t_{b_1} = 1$, $L_{\mathbf{p}_{b_1}}(s_1, a_1) = 2$, $L_{\mathbf{p}_{b_1}}(a_1, b_1) = 3$
$L(s, s_1) = \{L_{\mathbf{p}_{b_1}}(s, s_1), L_{\mathbf{p}_{c_1}}(s, s_1), L_{\mathbf{p}_{d_1}}(s, s_1), L_{\mathbf{p}_{a_1}}(s, s_1), L_{\mathbf{p}_{s_1}}(s, s_1)\} = \{1, 5, 8, 11, 13\}$,
$L(s, s_2) = \{L_{\mathbf{p}_{a_2}}(s, s_2), L_{\mathbf{p}_{b_2}}(s, s_2), L_{\mathbf{p}_{c_2}}(s, s_2), L_{\mathbf{p}_{d_2}}(s, s_2), L_{\mathbf{p}_{e_2}}(s, s_2), L_{\mathbf{p}_{s_2}}(s, s_2)\}$
$= \{2, 4, 6, 9, 10, 12\}$.

Notice that minimizing the largest label is equivalent to minimize the time needed by the algorithm. Indeed, one can just consider solutions where all labels in $\{1, \cdots, T\}$ are used: If some integer c is never used, we can decrease by 1 the value of each label $c' \geq c + 1$ in the considered feasible solution.

3 Lines

In this section we present an optimal algorithm to solve the SCF–labeling problem for an instance $\langle G, \mathbf{w}, s\rangle$, where G is a line, s is one of its end points, and node weights are arbitrary non negative integers, that is, $w(v) \geq 0$ for each $v \neq s$. The optimal value given in Theorem 2 was already given in [12] (theorem 4.1); however, we restate the algorithm in our notation since it is a starting point for the algorithm on trees given in the next section.

Let G be the line of length n with nodes $0, 1, \cdots, n$ and let $(i, i + 1)$, for $i = 0, \cdots, n - 1$, be the connection between subsequent nodes. Assume that the source node is $s = 0$ and $w(n) > 0$ (otherwise delete the end vertices of the line with weight 0).

Property 1 *A solution $(\mathcal{P}, L)$ of $\langle G, \mathbf{w}, s\rangle$ is feasible iff*
1) The labeling L induced by $\mathcal{P}$ is increasing,
2) for each $\mathbf{p}, \mathbf{q} \in \mathcal{P}$ with $L_{\mathbf{q}}(s, 1) \geq L_{\mathbf{p}}(s, 1)$: if $\mathbf{p}$ leads from s to node h, with $1 \leq h \leq n$, then

$$L_{\mathbf{q}}(s, 1) \geq L_{\mathbf{p}}(s, 1) + \min\{3, h\}. \tag{1}$$

The following notation will be used in the algorithm description.

– *Set a path (resp. a t–path) to node v*: establish a path from s to v together with its increasing labeling (resp. t–increasing labeling);
– A node $v \neq s$ *is completed*: if $w(v)$ paths from s to v have been set.

Theorem 2 **[12]** *For a line G with nodes $\{s = 0, 1, \ldots, n\}$ and and $w(i) \geq 0$ for $i = 1, \ldots, n$, it holds*

$$\mathcal{T}^*(G) = \max_{1 \leq i \leq n} M_i,$$

where $M_1 = w(1) + 2w(2) + 3\sum_{j \geq 3} w(j)$, $M_2 = 2w(2) + 3\sum_{j \geq 3} w(j)$, and $M_i = i - 3 + 3\sum_{j \geq i} w(j)$ if $i \geq 3$.

When each sensor node of the line has at least one request to be completed, Theorem 2 provides a simpler form of the optimal label (i.e. minimum time) .

Corollary 1 *If $w(i) \geq 1$, for $i = 1, \ldots, n$, then $\mathcal{T}^*(G) = M_1 = w(1) + 2w(2) + 3\sum_{j=3}^{n} w(j)$.*

LINE-labeling $(G, \mathbf{w}, s)$
- Set $\mathcal{P} = \emptyset$, $k = 1$.
- **while** there is a non completed node, **do**
  - Let i be the largest node which is not completed (e.g $i = \max\{j \mid 1 \leq j \leq n, w(j) > 0\}$.
  - Set a k–path to i in G, call it $\mathbf{p}_i$.
  - Let $\mathcal{P} = \mathcal{P} \cup \{\mathbf{p}_i\}$.
  - Let $w(i) = w(i) - 1$.
  - Set $k = k + \min\{3, i\}$.
- **return** $(\mathcal{P}, L)$, where L is the labeling induced by $\mathcal{P}$.

Table 1 : The SCF labeling algorithm on a line.

4 Trees

Let $T = (V, E)$ be any tree and s be a fixed node in T. We assume that each node has exactly one path to be set, i.e., $w(v) = 1$ for each $v \in V - \{s\}$ (recall that the source has weight $w(s) = 0$). We will show how to obtain an optimal labeling for $\langle T, \mathbf{w}, s \rangle$. The extention to the case $w(v) \geq 1$ can be easly obtained.

Definition 8 *Given a tree T. We shall denote by $|T|$ the* size of T *in terms of* the weights *of the nodes in T, that is*

$$|T| = \sum_{v \in V(T)} w(v).$$

Notice that $|T|$ represents the number of paths to be set in T. Since we assume that $w(v) = 1$ for each $v \in V - \{s\}$ then the algorithms starts with $|T| = |V| - 1$. Root T at s and let $T_1, T_2, \cdots, T_m$ be the subtrees of T rooted at the sons of s.

We also notice that in case $m = 1$, then T consists of a root of degree 1 and T_1 as the only subtree. A one-to-all personalized optimal broadcasting in T is obtained by applying the optimal algorithm LINE-labeling to the line L obtained from T by replacing the $w(j)$ vertices at distance j in T by a vertex j with weight $w(j)$ in L. Then by Corollary 1 the number of steps is $\mathcal{T}^*(L) = 1 + 2\alpha_1 + 3\beta_1$.

The main idea of the algorithm consists in setting, whenever that is possible, a path to a node in the subtree T_i having the largest shade value $\tau_i = 1 + 2\alpha_i + 3\beta_i$ (Definition 1). However, we have to be careful and, even if the algorithm is relatively simple, the proof of the value of gathering time in Theorem 1 is involved.

4.1 The algorithm

In order to describe the SCF labeling algorithm, we introduce the following terminology. Unless otherwise stated, in the following we assume that the subtrees are numbered according to the ranking given in Definition 2, that is $T_1, \cdots, T_m$ is a reordering of the subtrees of T such that $T_1 \preceq \cdots \preceq T_m$.

TREE-labeling $(T, \mathbf{w}, s)$ *[T has non empty subtrees $T_1, \ldots, T_m$ and root s]*

1. Set $\mathcal{P} = \emptyset$ and $t = 1$
Let $t_i = 1$ for $i = 1, \ldots, m$ *[t_i is the minimum step to set a path to T_i]*
Set $M = \{1, \ldots, m\}$ *[M represents the set of subtrees not yet completed]*

2. **while** $M \neq \emptyset$

2.1 Rename the indices in M so that for the permuted subtrees $T_1 \preceq \ldots \preceq T_{|M|}$

2.2 **if** there exists $i \leq |M|$ with $t_i \leq t$ **then**
Let i be the smallest such index (e.g. $t_1, \ldots, t_{i-1} > t$, $T_i \preceq \ldots \preceq T_{|M|}$).
if NOT ($|M| = 2$, $i = 1$, $\beta_1 = 1, \alpha_2 > \beta_2 = 0$, $t_2 \leq t + 1$) **then**
[Execute the generic step of the algorithm]
- Set a t-path to T_i and call it $\mathbf{p}$
- $\mathcal{P} = \mathcal{P} \cup \{\mathbf{p}\}$.
- If T_i is completed then $M = M - \{i\}$.
- $t_i = t + \min\{3, \ell\}$, where ℓ is the length of $\mathbf{p}$,
- Update T_i, eg.: $\tau_i = \tau_i - \min\{3, \ell\}$, $w(s_i) = w(s_i) - \begin{cases} 1 & \text{if } \ell = 1 \\ 0 & \text{oth.} \end{cases}$
$\alpha_i = \alpha_i - \begin{cases} 1 & \text{if } \ell = 2 \\ 0 & \text{oth.} \end{cases}$, $\beta_i = \beta_i - \begin{cases} 1 & \text{if } \ell \geq 3 \\ 0 & \text{oth.} \end{cases}$.

2.3 **else** *[The special case: $|M| = 2$, $i = 1$, $\beta_1 = 1, \alpha_2 > \beta_2 = 0, t_2 \leq t + 1$]*
- Set a t-path to T_1 and call it $\mathbf{p}$
- Set a $t + 1$-path to s_2 and call it $\mathbf{q}_1$
- Set a $t + 2$-path to T_2 and call it $\mathbf{q}_2$
- $\mathcal{P} = \mathcal{P} \cup \{\mathbf{p}, \mathbf{q}_1, \mathbf{q}_2\}$.
- $t_1 = t + 3$ and $t_2 = t + 4$.
- Update T_1 and T_2
(e.g. $\tau_1 = \tau_1 - 3$, $\beta_1 = 0$, $\tau_2 = \tau_2 - 3$, $w(s_2) = 0$, $\alpha_2 = \alpha_2 - 1$).
- If $\alpha_2 = 0$ then $M = \{1\}$.
- $t = t + 2$.

2.4 $t = t + 1$.

3. **return** $(\mathcal{P}, L)$

Table 2 : The SCF labeling algorithm on trees

- *One step:* one time–slot.
- A node $v \neq s$ *is completed* if a path from s to v has been set.
- *Set a path (resp. a t-path) to T_i*: set a path (resp. a t-path) to a node v in T_i which is the furthest from s among all nodes in T_i which are not yet completed.
When we set a path to some T_i the corresponding value $|T_i|$ of the remaining weights in T_i will be decreased by one and also α_i and β_i if they are non zero.

- T_i *is completed*: if a path has been set to each node in T_i, that is $|T_i| = 0$.
- Step t is called *idle* if no t-path is set.
- T_i is *available* at step t (e.g. a t–path to T_i can be set) only if no path was set to a node v in T_i at some step t' s.t. $t' < t < t' + \min\{3, \ell(v)\}$, where $\ell(v)$ is the level of v in T. Said otherwise, if at some step t' we set a path to a node v in T_i, then T_i is not available at step $t' + j$ where $1 \leq j < \min\{3, \ell(v)\}$. in particular if v is at a level at least 3, then T_i is not available at steps $t' + 1$ and $t' + 2$.

The SCF labeling algorithm is given in Tab.2. Following is an informal description of the behavior of the algorithm during a generic step $t \geq 1$: Let T_i be an available subtree that precedes all the other available subtrees of T according to the order relation $\preceq$; set a t–path to T_i; update the shade of T_i.

Example 3. The solution $(\mathcal{P}, L)$ given in Example 2 is the same one gets by applying the TREE-labeling algorithm on the tree T of Fig.1. Notice that steps $t = 8, 9, 10$ correspond to the special case of point 2.3 of the algorithm.

The TREE-labeling algorithm sets, at step t, a t-path to T_i only if T_i is available. We can then conclude that

Lemma 1 *The solution $(\mathcal{P}, L)$ returned by algorithm TREE-labeling on $\langle T, \mathbf{w}, s\rangle$ is feasible.*

4.2 Preliminary Results

We establish now some facts that will be used to prove the optimality of the proposed algorithm.

Fact 1 *For any subtree T_i with $|T_i| > 1$ it holds that $2|T_i| - 1 \leq \tau_i \leq 3|T_i| - 3$.*

Fact 2 *Let $T_i \preceq T_j$.*
- *If $\tau_i = \tau_j$ and $T_i \prec T_j$ then $\alpha_i > \alpha_j$ and $\beta_i < \beta_j$.*
- *$T_i = T_j$ (e.g., $\tau_i = \tau_j$ and $|T_i| = |T_j|$) iff $\alpha_i = \alpha_j$ and $\beta_i = \beta_j$.*

Fact 3 *If $T_i \preceq T_j$ then $\beta_j \leq \begin{cases} |T_i| - 2 & \text{if } |T_i| \geq 2 \\ 0 & \text{otherwise} \end{cases}$.*

The quantities $\Delta_{i,j}$ introduced in Definition 3 satisfy the following properties.

Fact 4 *For any i, j it holds $\Delta_{i,j} - \tau_i = |T_j| - |T_i|$*

Fact 5 *$\Delta_{i,j} \geq \max\{|T|, \tau_1 + \epsilon_T\}$ only if either $i = 1$ and $j = 2, 3$ or $i = 2$ and $j = 1$.*

Proof. Assume first either $i \geq 3$ or $i = 2$ and $j \geq 3$. We have $|T| - |T_i| - |T_j| \geq |T_1|$ or $|T| - |T_i| - |T_j| \geq |T_2|$. By Fact 3 we know that $\beta_i < \min\{|T_1|, |T_2|\} - 1$. Hence, in any case we get

$$|T| - \Delta_{i,j} = |T| - |T_i| - |T_j| - \beta_i + 1 > 2,$$

which implies $\Delta_{i,j} < |T| \leq \max\{|T|, \tau_1 + \epsilon_T\}$.
Assume now $i = 1$ and $j \geq 4$; supposing, by contradiction, $\Delta_{1,j} \geq |T|$ and $\Delta_{1,j} \geq \tau_1 + \epsilon_T$, we have

$$|T_2| + |T_3| \leq |T| - |T_1| - |T_j| = |T| - \Delta_{1,j} + \beta_1 - 1 \leq \beta_1 - 1 \leq |T_1| - 3. \quad (2)$$

From the assumption that $\Delta_{1,j} \geq \tau_1 + \epsilon_T$ and by Fact 4 we get $|T_1| \leq |T_j|$. This, (2), and Fact 1 imply

$$\tau_j \geq 2|T_j| - 1 \geq 2|T_1| - 1 \geq 2(|T_2| + |T_3|) + 5 > \frac{2}{3}(\tau_2 + \tau_3) + 5 \geq \frac{4}{3}\tau_3 + 5 > \tau_3$$

thus contradicting the assumption $T_3 \preceq T_j$ for any $j \geq 4$. □

4.3 A lower bound

Let T be such that $T_1 \preceq T_2 \preceq \ldots \preceq T_m$. Define

$$Max(T) = \max\{|T| = |V| - 1,\ \tau_1 + \epsilon_T,\ \Delta_{1,2},\ \Delta_{2,1},\ \Delta_{1,3}\}$$

Lemma 2 *Assuming that $T_1 \preceq T_2 \preceq \ldots \preceq T_m$, we have $\mathcal{T}^*(T) \geq Max(T)$.*

Proof. Any algorithm needs to set a path to each node, hence $\mathcal{T}^*(T) \geq |T|$. By Definition 1 and Corollary 1, the shade τ_i of T_i is the minimum label that can be assigned when only paths to the nodes in T_i are set. Since paths must be set to all nodes in each T_i, for $i = 1, \cdots, m$, and $\tau_1 \geq \tau_2 \geq \cdots \geq \tau_m$ we have that $\mathcal{T}^*(T) \geq \tau_1$.
Furthermore, if $T_1 = T_2$ then at least $\tau_1 + 1$ labels are necessary.
Consider now $\Delta_{i,j}$. For each path to a node at level at least 3 in T_i no path to some other node in T_i can be set in the following 2 steps. Moreover, at most one of the following two steps can be used to set a path to T_j, except for the eventual step in which a path to the root of T_j is set and immediately after a path to some other node in T_j is set. The remaining step can be used to set a path to some T_ℓ with $\ell \neq i, j$. Hence, any algorithm has at least $\beta_i - 1 - \sum_{\ell \neq i,j} |T_\ell|$ idle steps, which implies $\mathcal{T}^*(T) \geq |T| + \beta_i - 1 - \sum_{\ell \neq i,j} |T_\ell| = \Delta_{i,j}$. By Fact 5, we get that $Max(T)$ lower bounds $\mathcal{T}^*(T)$. □

4.4 Optimality

We show now that the SFC–labeling algorithm for trees is optimal, that is, the maximum label assigned to any arc of T is $\mathcal{T}(T) \leq Max(T)$ thus matching the lower bound of Theorem 2.

We first recall that we are in the hypothesis that the weight of each node is 1. The order in which nodes are chosen as end–points of the paths set by the algorithm implies that the largest label assigned to an arc of T is always to be searched among those assigned to the arcs outgoing the root s of T. Therefore, it coincides with the largest t for which a t–path is set in T.

Lemma 3 *Let t denote the largest integer such that a t–path is set in T during the execution of the SFC–labeling algorithm. The largest label assigned by the algorithm to any arc of T is $\mathcal{T}(T) = t$.*

By the above Lemma, we need to show that the largest t such that a t–path is set in T is upper bounded by $Max(T)$. The proof will proceed by induction. We will consider the first steps of the algorithm mainly those which send to different subtrees (before the step where we send again to a subtree to which we already sent) and we will apply the induction on the tree T' obtained by deleting the vertices completed in these first steps. For that we give the following definition.

Definition 9 *We denote the fact that the algorithm on T starts by setting k paths to pairwise different subtrees of T (that is, it sets a t–path to some node v_i in T_i, for $i = 1, \ldots, k$) by*

$$\langle T_1 \ldots T_k \rangle$$

We denote by T' the updated tree, resulting from $\langle T_1 \ldots T_k \rangle$, that is, T' has subtrees $T'_1 \ldots, T'_k, T'_{k+1} \ldots, T'_m$, where

- *T'_i denotes the updated subtree T_i after the i–path to v_i has been set (that is, $w'(v_i) = 0$ and $|T'_i| = |T_i| - 1$, for $i = 1, \ldots, k$*
- *$T'_{k+1} = T_{k+1}, \ldots, T'_m = T_m$.*

Notice that the subtrees $T'_1 \ldots, T'_m$, are not necessarily ordered according to the relation $\preceq$. Let $i_1, i_2, \cdots, i_m$ be a permutation of $1, \ldots, m$ such that $T'_{i_1} \preceq \ldots \preceq T'_{i_m}$; we will always consider permutations that maintain the original order on equal subtrees, that is

$$\text{if } \; T'_{i_j} = T'_{i_\ell} \quad \text{then} \quad i_j < i_\ell. \tag{3}$$

We denote by $\alpha'_i, \beta'_i, \tau'_i$ the parameters of T'_i. In particular (unless the special case $k = 2$, $\beta_1 = 1, \alpha_2 > \beta_2 = 0$, $T_3 = \emptyset$, and T_2 is available) we have for $i = 1, \ldots, k$:

$$\alpha'_i = \alpha_i - \begin{cases} 1 & \text{if } \beta_i = 0,\ \alpha_i \geq 1 \\ 0 & \text{otherwise} \end{cases}, \quad \beta'_i = \beta_i - \begin{cases} 1 & \text{if } \beta_i \geq 1 \\ 0 & \text{otherwise} \end{cases},$$

$$\tau'_i = \tau_i - \begin{cases} 3 & \text{if } \beta_i \geq 1 \\ 2 & \text{if } \beta_i = 0,\ \alpha_i \geq 1 \\ 1 & \text{if } |T_i| = 1 \\ 0 & \text{if } T_i = \emptyset \end{cases}.$$

Fact 6 *Assume $\langle T_1 \ldots T_k \rangle$ and that NOT ($k = 2$, $\beta_1 = 1, \alpha_2 > \beta_2 = 0$, $T_3 = \emptyset$, and T_2 is available). For any $1 \leq i < j \leq k$.*

1) If $\tau_i > \tau_j$ then $\tau'_i \geq \tau'_j$;
2) if $T_i \prec T_j$ and $T'_j \prec T'_i$ then $\beta_j = 0$, $\beta_i \geq 2$, $|T_i| < |T_j|$, and $\tau_i = \tau_j + 1$.

Fact 7 *Assume $\langle T_1 \ldots T_k \rangle$ with either $k \geq 4$ or $k = 3$ and $T_3 \preceq T'_1, T'_2$:*

i) $|T| \geq \tau_1 + k - 2$;
ii) $|T_i| \geq \beta_1 + 1$, for each $i = 2, \ldots, k$;
iii) $|T_i| \geq \beta_2 + 1$, for each $i = 3, \ldots, k$.

The following lemma together with lower bound of Lemma 2 prove Theorem 1.

Lemma 4 *Assume $T_1 \preceq \ldots \preceq T_m$. The solution returned by algorithm TREE-labeling satisfies*

$$\mathcal{T}(T) \leq Max(T) \tag{4}$$

Proof. At any step of the algorithm the tree can have any number $m \geq 1$ of subtrees of positive weight. When we say that the algorithm sets a t–path to a subtree T_i and $|T_i| = 0$ at step t, this means that no t–path is actually set (e.g. t is an idle step).
We first analyze the special case of the algorithm in which $m = 2$, $\beta_1 = 1$, $\beta_2 = 0$ and T_2 is available. So $\tau_1 > \tau_2$ and $\alpha_1 \geq \alpha_2 - 1$. The first two steps of the algorithm are $\langle T_1 T_2 \rangle$, where the path set to T_2 is a path to s_2 (the root of T_2). Let T' be the tree resulting after $\langle T_1 T_2 \rangle$, at the third step a path to T_2' is set. Hence, the first three steps of the algorithm are: $\langle T_1 T_2 \rangle \langle T_2' \rangle$

Let T^2 be the tree resulting after $\langle T_1 T_2 \rangle \langle T_2' \rangle$. Next the algorithm on T proceeds as follows

$$\langle T_1^2 T_2^2 \rangle \langle T_1^3 T_2^3 \rangle \dots \langle T_1^\ell T_2^\ell \rangle \dots \langle T_1^{\alpha_1+1} T_2^{\alpha_1+1} \rangle \langle T_1^{\alpha_1+2} \rangle.$$

where T^ℓ is the tree resulting from $T^{\ell-1}$ after the 2 steps $\langle T_1^{\ell-1} T_2^{\ell-1} \rangle$. To see this, we notice that in each T^ℓ it holds $T_1^\ell \prec T_2^\ell$, since $\tau_1^2 = \tau_1 - 3 > \tau_2 - 3 = \tau_2^2$ and $\tau_1^\ell = \tau_1^{\ell-1} - 2 > \tau_2^\ell = \max\{\tau_2^{\ell-1} - 2, 0\}$, for $\ell > 2$. Moreover, in the hypothesis of this case $\alpha_1 \geq \alpha_2 - 1$, which implies that $T_2^\ell = \emptyset$ for $\ell > \alpha_2$. Finally, by the hypothesis we have

$$\epsilon_T = 0, \quad |T| = 3 + \alpha_1 + \alpha_2 \leq 3 + 2\alpha_1 + 1 = \tau_1, \quad \text{and } \Delta_{1,2}, \Delta_{2,1} \leq |T|.$$

Hence, $Max(T) = \tau_1$; but $\mathcal{T}(T) = 3 + 2\alpha_1 + 1 = \tau_1 = Max(T)$.

The rest of the proof is devoted to show that $\mathcal{T}(T) \leq Max(T)$ for each tree. The proof is by induction on the shade of T_1, (recall that $T_1 \preceq T_2 \preceq \dots \preceq T_m$). As a base consider the trees of the special case above and trees T such that $\tau_1 = 1$; in the latter case, we have $|T_i| = 1$ for each $i = 1, \dots, m$ and $\mathcal{T}(T) = |T| = Max(T)$.

Suppose now that (4) holds for any tree in which the shade of the first subtree (according to the relation $\preceq$) is at most $\tau_1 - 1$; we prove that (4) holds for T.

Notice that we are assuming that T does not belong to the special case (e.g., $m = 2$, $\beta_1 = 1$, $\beta_2 = 0$, and T_2 is available) and that $|T_1| \geq 2$.

We separate four cases according to the value attaining $Max(T)$.

Case 1: $Max(T) = \Delta_{1,2} > \max\{\tau_1 + \epsilon_T, |T|\}$**.**
In such a case we know that $\beta_1 > 1$, otherwise $\Delta_{1,2} = |T_1| + |T_2| + \beta - 1 \leq |T|$; hence, the first tree steps of the algorithm are (including the case $|T_3| = 0$) $\langle T_1 T_2 T_3 \rangle$.

Let T' be the tree resulting after $\langle T_1 T_2 T_3 \rangle$. We will show that after the first 3 steps $\langle T_1 T_2 T_3 \rangle$, the algorithm on T proceeds as on input T' and

$$Max(T') \leq Max(T) - 3. \tag{5}$$

This implies the desired inequality $\mathcal{T}(T) = 3 + \mathcal{T}(T') \leq 3 + Max(T') = Max(T)$.

By definition of $\Delta_{1,2}$ and using $\Delta_{1,2} > |T|$, we get

$$|T| - |T_1| - |T_2| < \beta_1 - 1. \tag{6}$$

By Fact 4 and using $\Delta_{1,2} > \tau_1$, we get

$$|T_1| < |T_2|. \tag{7}$$

By (6) and Fact 1, we get

$$\tau_3 < 3|T_3| \leq 3(|T| - |T_1| - |T_2|) < 3(\beta_1 - 1) = (3\beta_1 + 2\alpha_1 + 1) - (2\alpha_1 + 4),$$

from which, since $\alpha_1 \geq 1$, it follows

$$\tau_3 < \tau_1 - 6 = \tau_1' - 3. \tag{8}$$

Moreover, by (6) and (7) we have

$$|T_2| \geq |T_1| + 1 \geq \beta_1 + \alpha_1 + 2 \geq \beta_1 + 3 > (|T| - |T_1| - |T_2|) + 4 \geq |T_3| + |T_4| + 4; \tag{9}$$

which, by Fact 1, implies $\tau_2 \geq 2|T_2| - 1 > 2(|T_3| + |T_4|) + 7 \geq 4\min\{|T_3|, |T_4|\} + 7$.

Noticing that Fact 1 implies $4|T_4| > \tau_4$ and $4|T_3| > \tau_3 \geq \tau_4$, we get

$$\tau_2 \geq \tau_4 + 8. \tag{10}$$

From (8) and (10) and recalling that $\tau_1 \geq \tau_2$, we obtain that in the tree T', resulting after $\langle T_1 T_2 T_3 \rangle$:

$$T_1' \prec T_3', \quad T_1' \prec T_4' = T_4, \quad T_2' \prec T_4' = T_4.$$

Moreover, we have

$$T_2' \prec T_3';$$

indeed, if we assume $T_2' \succeq T_3'$ we have either $|T_2| = |T_3|$ or, by Fact 6, $|T_3| > |T_2|$ contradicting (9).

We notice that $T_1' \neq T_2'$, since by (7) they have different weights. Hence, by the definition of $\prec$ (cfr. Definition 2), we get that the only possible orderings on the the subtrees of T' are:

$$T_1' \prec T_2' \prec T_3', \quad T_1' \prec T_2' \prec T_4', \quad T_2' \prec T_1' \prec T_3', \quad T_2' \prec T_1' \prec T_4'.$$

Moreover, both sequences of steps $\langle T_1 T_2 T_3 \rangle \langle T_1' T_2' \rangle$ and $\langle T_1 T_2 T_3 \rangle \langle T_2' T_1' \rangle$ are possible during the execution of the algorithm on T; in particular if $T_2' \prec T_1'$ we know by Fact 6 that $\beta_2 = 0$.

Hence, after the first 3 steps, the algorithm on T proceeds as on input T'. For T' we have: $\tau_1' = \tau_1 - 3$ (since $\beta_1 > 1$),

$$|T'| = |T| - \begin{cases} 3 & \text{if } |T_3| > 0 \\ 2 & \text{otherwise} \end{cases}, \qquad \epsilon_{T'} = \epsilon_T = 0 \text{ (since } |T_1| < |T_2|).$$

In case $T_1' \prec T_2'$, it holds $\Delta_{1,2}' = \Delta_{1,2} - 3$,

$$\Delta_{2,1}^{'} = \begin{cases} \Delta_{2,1} - 3 & \text{if } \beta_2 > 0 \\ |T_2'| + |T_1'| - 1 < |T'| & \text{if } \beta_2 = 0 \end{cases}, \qquad \Delta_{1,3}', \Delta_{1,4}' < \Delta_{1,2} - 3,$$

where the last inequality follows from (9).

In case $T_2' \prec T_1'$, by Fact 6 we have $\beta_2 = 0$, $\beta_1 \geq 1$ and $\tau_1 > \tau_2$; hence $\tau_2' = \tau_2 - 2 = \tau_1 - 3$ and

$$\Delta_{1,2}' = \Delta_{1,2} - 3, \qquad \Delta_{2,i}' = |T_2'| + |T_i'| + \beta_2' - 1 = |T_2'| + |T_i'| - 1 < |T'| \ (i = 1, 3, 4).$$

Summarizing, in both cases $T_1' \prec T_2'$ and $T_2' \prec T_1'$, inequality (5) holds.

Due to space limits, the proofs of the other cases:
$Max(T) = \Delta_{2,1} > \max\{\tau_1 + \epsilon_T, |T|\}$, $Max(T) = \Delta_{1,3} > \max\{\tau_1 + \epsilon_T, |T|\}$, and $Max(T) = \max\{\tau_1 + \epsilon_T, |T|\}$ are omitted from this extended abstract. □

5 Conclusion

In this paper we give a relatively simple protocol for trees with $w(u) = 1$ packet to transmit. The results can be easily extended to the case where all the $w(u)$ are positive (or at least there is no more than two consecutive nodes with weights 0). It might be that the algorithm is optimal for any weight function by replacing τ_i with M_i (see Theorem 2); but the proof seems more complicated. It will be also interesting to find the complexity of the gathering problem for general graphs without buffering (with buffering it is known to be NP-hard).

References

1. J-C. Bermond, R. Corrêa, M. Yu, Gathering algorithms on paths under interference constraints, In 6th Conf. on Alg. and Compl., LNCS 3998, 115–126, 2006.
2. J-C. Bermond, L. Gargano, A. Rescigno, "Gathering with Minimum Delay in Sensor Networks",INRIA TR 2008, http://hal.inria.fr/inria-00256896/fr/.
3. J-C. Bermond, J. Galtier, R. Klasing, N. Morales, S. Pérennes, Hardness and approximation of gathering in static radio networks, Parallel Processing Letters, **16 (2)** 165–183, 2006.
4. J-C. Bermond, J. Peters, Efficient gathering in radio grids with interference, In AlgoTel'05, pages 103–106, Presqu'île de Giens, 2005.
5. J-C. Bermond, M. Yu, Optimal gathering algorithms in multi-hop radio tree-networks with interferences, manuscript 2008.
6. P. Bertin, J-F. Bresse, B. Le Sage, Accès haut débit en zone rurale: une solution "ad hoc", France Telecom R&D, 22:16–18, 2005.
7. V. Bonifaci, P. Korteweg, A. Marchetti-Spaccamela, L. Stougie, An Approximation Algorithm for the Wireless Gathering Problem, Proc. of SWAT 06, 2006.
8. C-Y Chong , S.P. Kumar, Sensor networks: Evolution, opportunities, and challenges, Proc. of the IEEE ,**91 (8)** (2003) 1247-1256.
9. S. Coleri, P. Varaiya, Energy Efficient Routing with Delay Guarantee for Sensor Networks, Wireless Networks, to appear
10. K. Dasgupta, M. Kukreja, K. Kalpakis, Topology-aware placement and role assignment for energy-efficient information gathering in sensor networks, Proc. IEEE ISCC'03 (2003) 341-348.
11. E. Falck, P. Floreen, P. Kaski, J. Kohonen J., P. Orponen, Balanced data gathering in Energy-constrained sensor networks, Proc. of ALGOSENSORS 2004, LNCS 3121, 2004, 59-70.

12. C. Florens, M. Franceschetti, R.J. McEliece, Lower Bounds on Data Collection Time in Sensory Networks, IEEE J. on Sel. Ar. in Com., **22 (6)** (2004) 1110–1120.

13. D. Ganesan, R. Cristescu, B. Beferull-Lozano, Power-efficient sensor placement and transmission structure for data gathering under distortion constraints, IPSN 2004, (2004) 142-150.

14. L. Gargano, Time Optimal Gathering in Sensor Networks, Proc. SIROCCO 2007, LNCS , Vol. 4474, pp. 7-10.

15. L. Gargano, A.A. Rescigno, Optimally Fast Data Gathering in Sensor Networks, Proc. MFCS 2006, LNCS , Vol. 4162, pp. 399-411.

16. H. Gupta, V. Navda, S.R. Das,V. Chowdhary, Efficient gathering of correlated data in sensor networks, Proc. of ACM MobiHoc'05, 2005, 402-413.

17. L. Gasieniec, I Potapov, Gossiping with Unit Messages in Known Radio Networks, IFIP TCS 2002, 193-205.

18. B. Ho, V.K. Prasanna, Constrained flow optimization with application to data gathering in sensor networks, Proc. of ALGOSENSORS 2004, LNCS **3121** (2004) 187-200.

19. C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, F. Silva, Directed diffusion for wireless sensor networking, IEEE/ACM Trans. Netw. **11(1)**, (2003) 2-16.

20. B. Krishnamachari, D. Estrin, S. Wicker, Modeling data-centric routing in wireless sensor networks, Proc. of IEEE INFOCOM 2002, (2002).

21. S. Lindsey, C. Raghavendra, Pegasis: Power-efficient gathering in sensor wireless networks, Proc. of IEEE Aerospace Conference, 2002.

22. S. Lindsey, C. Raghavendra, K.M. Sivalingam, Data gathering algorithms in sensor networks using energy metrics, IEEE Trans. on Par. and Distr. Sys.**13 (9)** (2002) 924-935.

23. K. Padmanabh, R. Roy, Multicommodoty flow fased maximum lifetime routing in wireless sensor network, Proc. of IEEE ICPADS 2006,(2006) 187-194.

24. C. Shen, C. Srisathapornphat, C. Jaikaeo, Sensor information networking architecture and applications, IEEE Personal Communications, (2001) 52-59.

25. Y. Yu, B. Krishnamachari, V. Prasanna, Energy-latency tradeoffs for data gathering in wireless sensor networks, Proc. of IEEE INFOCOM 2004, (2004).

26. X. Zhu, B. Tang, H. Gupta, Delay efficient data gathering in sensor networks, Proc. of MSN 2005, LNCS **3794** (2005) 380-389.