Gathering with Minimum Completion Time in Sensor Tree Networks

Jean-Claude Bermond, Luisa Gargano, Adele Rescigno

Gathering with Minimum Completion Time in Sensor Tree Networks

Jean–Claude Bermond*
MASCOTTE
joint project CNRS-INRIA-UNSA
2004 Route des Lucioles
BP 93, F-06902 Sophia-Antipolis
France
E-mail: bermond@sophia.inria.fr

Luisa Gargano†
Dipartimento Informatica ed Applicazioni
Universitá di Salerno
84084 Fisciano (SA)
Italy
E-mail: lg@unisa.it

Adele A. Rescigno
Dipartimento Informatica ed Applicazioni
Universitá di Salerno
84084 Fisciano (SA)
Italy
E-mail: rescigno@unisa.it

*Partially supported by the CRC CORSO with France Telecom, by the European FET project AEOLUS
†Work partially done while visiting INRIA at Sophia-Antipolis

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 range of each other. We suppose that the interference range is the same as the transmission range; therefore 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 exactly one data packet to send.

Key words: Data gathering, omnidirectional antennas, time, tree sensor networks.

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 – i.e. 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 non–replentishable batteries. It is then important to bound the energy consumption of data dissemination [11, 19, 25]. 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 [27].

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 off–line data gathering in tree networks.

1.1 Network model

We adopt the network model considered in [3, 13, 18]. 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 identify 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 pairs 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 [13, 16, 27] and contrarily to [3, 5, 18], 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 directed 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, i.e., $(u, v) \in A$ 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)). Note that the interferences are not necessarily symmetric and we can have $d(u, u') = 1$. We implicitely assume that $u \neq u'$, that is, a node sends to at most one neighbor. This is not restrictive, as we will consider trees and personalized broadcast.

The off-line collision–free data gathering problem can be then stated as follows.

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 personalized broadcast 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 [9]. The problem of minimizing the delay of the gathering process has been recently recognized and studied [7, 15]. The authors of [13] 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 [13] 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, [13] gives optimal gathering schedules for trees. An optimal algorithm for general networks has been presented in [16] in the case that each node has one packet of sensed data to deliver.

The work in [27] also deals with the latency of data gathering under the assumption of directional antennas; the difference with [13] 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, 4, 5, 8], 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 $d_I = d_T = 1$). 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 [8] 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 [11, 14], data aggregation [17, 20, 21, 22], or efficient data flow [12, 19, 24]. Papers [10, 23, 26] consider the minimization of the gathering delay in conjunction with the energy spent to complete the process.

A preliminary version (extended abstract) of this paper appears in [2].

1.3 Paper Overview.

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

In Section 3 we shortly illustrate an optimal algorithm in case G is a tree with only one subtree. It is obtained by using the optimal algorithm on the line with the BS s as one of its endpoints. The result on the line was first presented in [13].

In Section 4 we give an optimal algorithm in case the graph is a tree T with *one specific data item to be sent to each node* and a closed formula (see Theorem 2)

2 Mathematical Formulation

We now formally formulate the one-to-all personalized broadcast problem. Let $G = (V, A)$ be the communication network and 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 vector $\mathbf{w}$ represents the vector of the weights of the nodes in V.

We need to schedule the transmissions in order to create $w(v)$ collision–free routes from s to node v, for each $v \in V - \{s\}$. We define the schedule by assigning time-labels to the arcs involved in transmissions: if arc $(u, v) \in A$ has label t then u transmits to v at time t.

Definition 1. *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 2. *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, $N(u) = \{x \mid (u, x) \in A\} = \{x \mid (x, u) \in A\}$.

In the above terminology, conditions(i) and (ii) of the Introduction give the following definition.

Definition 3. *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* $L(u, v)$*),*
- $L(u, v) \cap L(v, w) = \emptyset$, *for each* $w \in N(v)$ *(condition (i)),*
- $L(u, v) \cap L(w, z) = \emptyset$, *for each* $w \in N(v) \cup \{v\}$, $z \in N(w)$ *(condition (ii)).*

Definition 4. *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 vector 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 an 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. *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: $t_b = 1$, $t_e = 2$, $t_f = 4$, $t_c = 5$, $t_g = 6$, $t_d = 8$, $t_{s_2} = 9$, $t_h = 10$, $t_a = 11$, $t_\ell = 12$, $t_{s_1} = 13$.*

As an example, we have

$\mathbf{p}_b = (s, s_1, a, b)$, *with* $L_{\mathbf{p}_b}(s, s_1) = t_b = 1$, $L_{\mathbf{p}_b}(s_1, a) = 2$, $L_{\mathbf{p}_b}(a, b) = 3$

$L(s, s_1) = \{L_{\mathbf{p}_b}(s, s_1), L_{\mathbf{p}_c}(s, s_1), L_{\mathbf{p}_d}(s, s_1), L_{\mathbf{p}_a}(s, s_1), L_{\mathbf{p}_{s_1}}(s, s_1)\} = \{1, 5, 8, 11, 13\}$,

$L(s, s_2) = \{L_{\mathbf{p}_e}(s, s_2), L_{\mathbf{p}_f}(s, s_2), L_{\mathbf{p}_g}(s, s_2), L_{\mathbf{p}_h}(s, s_2), L_{\mathbf{p}_\ell}(s, s_2), L_{\mathbf{p}_{s_2}}(s, s_2)\}$

$= \{2, 4, 6, 9, 10, 12\}$.

Notice that minimizing the largest label assigned to any arc of G 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 Trees with one subtree

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 tree T with only one subtree ($m = 1$) and where node weights are arbitrary non negative integers, that is, $w(v) \geq 0$ for each $v \neq s$. The optimal algorithm was already given for the line in [13] (Theorem 4.1). For sake of completness we restate, in Fig. 2, the algorithm in our notation since it is a starting point for the algorithm on general trees given in the next section. 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 all the required paths from s to v have been set.

An optimal algorithm for a tree T rooted at s, with one subtree and maximum level n can be easily obtained:

1) associate to T, the line $\mathcal{L}$ with nodes $s = 0, 1, \cdots, n$, where the node i has a weight $w(i)$ equal to the sum of the weights of the nodes at level i in T (i.e., nodes at distance i from s in T);

2) set a path from s to one of the $w(i)$ nodes at level i in T every time the algorithm for $\mathcal{L}$ set one of the $w(i)$ paths to node i in $\mathcal{L}$.

Theorem 1. **[13]** *Let T be a tree with one subtree and let $w(i) \geq 0$ be the sum of the weights of nodes at level i for $i = 1, \ldots, n$. Then the optimal gathering time is*

$$\mathcal{T}^*(T) = \max_{1 \leq i \leq n} M_i, \qquad \text{where} \qquad M_i = \begin{cases} w(1) + 2w(2) + 3\sum_{j \geq 3} w(j) & \text{if } i = 1, \\ 2w(2) + 3\sum_{j \geq 3} w(j) & \text{if } i = 2, \\ i - 3 + 3\sum_{j \geq i} w(j) & \text{if } i \geq 3. \end{cases}$$

When $w(i) > 0$, for $i = 1, \ldots, n$, Theorem 1 provides a simpler form of the optimal label (i.e. minimum time).

Corollary 1. *Let T be a tree with one subtree and let $w(i) \geq 1$, for $i = 1, \ldots, n$, then* $\mathcal{T}^*(T) = M_1 = w(1) + 2w(2) + 3\sum_{j=3}^{n} w(j)$.

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 $\langle T, \mathbf{w}, s \rangle$.

Definition 5. *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*

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

In particular, if $w(u) = 1$ for each $v \in V(T) - \{s\}$ then $|T| = |V| - 1$.

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. Through this section we assume $m \geq 2$; the case $m = 1$ was discussed in Section 3.

Definition 6. *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 $i = 1, \ldots, m$. We have $|T_i| = 1 + \alpha_i + \beta_i$.

The following definition gives an order on the subtrees based on their shadow. This order will induce a priority between subtrees and will be useful in the algorithm as we will choose among the subtrees to which the source can send a message the one with the highest priority.

Definition 7. *(Order of priority) 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 both* $\tau_i = \tau_j$ *and* $|T_i| = |T_j|$*(they are not necessarily isomorphic).*

Definition 8. *Let* $m \geq 2$ *and* $T_1 \preceq T_2 \preceq \ldots \preceq T_m$. *Define*

$$\epsilon_T = \begin{cases} 1 & \text{if } T_1 = T_2 \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad \Delta_{i,j} = \begin{cases} |T_i| + |T_j| + \beta_i - 1 & \text{if } i, j = 1, \ldots, m, \text{ with } i \neq j \\ 0 & \text{if } i > m \text{ or } j > m \end{cases}.$$

We devote the next subsections to prove the following result.

Theorem 2. *Suppose each node of a tree* T *has one data item and let* n *denote the number of vertices of* T. *Let* $m \geq 2$ *and* $T_1 \preceq T_2 \preceq \ldots \preceq T_m$ *we have that the optimal gathering time is*

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

Recall that if $m = 2$ then we assume $\Delta_{1,3} = 0$. Furthermore, by Corollary 1, when $m = 1$ then the number of steps is $\mathcal{T}^*(T) = \tau_1 = 1 + 2\alpha_1 + 3\beta_1$ (coherent with the theorem).

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

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 6). 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.

- *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 (i.e. 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 $t' + 2$.

Unless otherwise stated, in the following we assume that the subtrees are numbered according to the ranking given in Definition 7, that is $T_1, \cdots, T_m$ is a reordering of the subtrees of T such that $T_1 \preceq \cdots \preceq T_m$.

The SCF labeling algorithm is given in Fig.3. 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 (cont.). *The solution $(\mathcal{P}, L)$ given in the Example is the same one gets by applying the TREE-labeling algorithm on the tree T of Fig.1.*

The table given in Fig. 4 reports how the TREE-coloring algorithm sets, step by step, the paths from the source s to the nodes of the tree T given in Fig. 1.

In the table we refer to T_k, for $k = 1, 2$, as the subtree rooted at s_k. Each row of the table shows for a fixed time t: the ordering of subtrees T_1 and T_2 at the beginning of time-step t; the index k of the

subtree to which a t-path is set at time-step t; the algorithm's point corresponding to time-step t; the t-path in T_k which is set at time-step t; the updated values of t_k, τ_k, α_k and β_k after time-step t.

Recall that at the beginning of the algorithm we have $\alpha_1 = 1$, $\beta_1 = 3$, $\tau_1 = 12$, $\alpha_2 = 5$, $\beta_2 = 0$, $\tau_2 = 11$. Furthermore, the minimum step to set a path in T_1 and T_2 is $t_1 = t_2 = 1$.

Notice that $t = 8$ corresponds to the special case 2.2.b of the algorithm; in this case all the remaining paths are set.

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.

The order in which nodes are served by the algorithm implies that (excluding the special case which is considered separately), whenever we refer to a subtree T_i we can assume that $\alpha_i = 0$ only if $\beta_i = 0$ (since the paths to nodes at level 2 in T_i are set only after that all the paths to nodes at level at least 3 are set) and that the path to the root of T_i is set only when $\alpha_i = \beta_i = 0$.

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

Proof. By definition $|T_i| = \beta_i + \alpha_i + 1$ and $\tau_i = 3\beta_i + 2\alpha_i + 1$. Hence

$$2|T_i| - 1 = 2\beta_i + 2\alpha_i + 1 \leq 3\beta_i + 2\alpha_i + 1 = \tau_i \leq 3\beta_i + 3\alpha_i = 3|T_i| - 3,$$

where the last inequality follows noticing that $|T_i| > 1$ implies $\alpha_i \geq 1$. □

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 $\beta_i < \beta_j$.*
- *$T_i = T_j$ (i.e., $\tau_i = \tau_j$ and $|T_i| = |T_j|$) iff $\alpha_i = \alpha_j$ and $\beta_i = \beta_j$.*

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

Proof. Trivially, if $|T_i| = 1$ then $\beta_j = 0$. Let then $|T_i| > 1$. If $T_i \preceq T_j$ then $\tau_i \geq \tau_j$. This implies that $3\beta_i \geq 3\beta_j + 2\alpha_j - 2\alpha_i$. From this we get

$$|T_i| = 1 + \alpha_i + \beta_i \geq 1 + \alpha_i + \frac{3\beta_j + 2\alpha_j - 2\alpha_i}{3} \geq \beta_j + \frac{2}{3}\alpha_j + \frac{1}{3}\alpha_i + 1.$$

Hence, noticing that $\alpha_i \geq 1$, we get $|T_i| \geq \beta_j + 2$. □

The quantities $\Delta_{i,j} = |T_i| + |T_j| + \beta_i - 1$, introduced in Definition 8 satisfy the following properties.

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

Proof. Recalling that $\tau_i = 3\beta_i + 2\alpha_i + 1$ and by Definition 8 we have

$$\Delta_{i,j} - \tau_i = |T_i| + |T_j| + \beta_i - 1 - (1 + 2\alpha_i + 3\beta_i) = |T_i| + |T_j| - 2|T_i| = |T_j| - |T_i|.$$

□

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

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

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

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

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

4.3 The lower bound

Definition 9. *Let T be such that $m \geq 2$ and $T_1 \preceq T_2 \preceq \ldots \preceq T_m$. Define*

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

where ϵ_T and $\Delta_{i,j}$ are defined in Definition 8 and where, if $m = 2$, then $\Delta_{1,3} = 0$.

Theorem 3. *Assuming that $m \geq 2$ and $T_1 \preceq T_2 \preceq \ldots \preceq T_m$ 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 6 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 $\tau_1 = \tau_2$, then at least $\tau_1 + 1$ labels are necessary as the first path to T_2 is set only at step 2. Note that, if $\tau_1 = \tau_2$, but $T_1 \neq T_2$ that is $|T_1| > |T_2|$, then by fact 4, $\Delta_{2,1} - \tau_2 = |T_1| - |T_2| \geq 1$. So $\Delta_{2,1} \geq \tau_2 + 1 = \tau_1 + 1$ and $\Delta_{2,1}$ is a better lower bound. If $T_1 = T_2$ (which implies $\tau_1 = \tau_2$), then $\tau_1 + \epsilon_T$ 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 $\mathcal{T}^*(T)$. □

Remark 1. *There are trees for which any of the five parameters in the definition of $Max(T)$ represents the unique exact bound and so we will have to distinguish all the cases to prove the optimality. Examples are given below.*

- $|T|$ it the unique maximum: Take 3 subtrees T_1, T_2, T_3 with $\alpha_1 = \alpha_2 = \alpha_3 = \alpha$ and $\beta_1 = \beta_2 = \beta_3 = 0$. Then $\tau_1 = 1 + 2\alpha < 4 + 3\alpha = |T|$ and $\Delta_{i,j} = |T_i| + |T_j| - 1 < |T|$.
  Fig. 5a) shows an example of this case with $\alpha_1 = \alpha_2 = \alpha_3 = 3$; here $|T| = 13$, $\tau_1 + \epsilon_T = 8$, $\Delta_{1,2} = \Delta_{2,1} = \Delta_{1,3} = 7$.

- $\tau_1 + \epsilon_T$ is the unique maximum: Take 2 subtrees T_1, T_2 with $\alpha_1 = \alpha_2 = 1$, $\beta_1 \geq 3$ and $\beta_2 \leq \beta_1$ (so $\tau_1 \geq \tau_2$). Then $|T| = 5 + \beta_1 + \beta_2 \leq 5 + 2\beta_1 < 3 + 3\beta_1 = \tau_1$ (as $\beta_1 > 2$).

$$\Delta_{2,1} \leq \Delta_{1,2} = \begin{cases} 3 + 2\beta_1 + \beta_2 < 3 + 3\beta_1 = \tau_1 & \text{if } \beta_2 < \beta_1 \text{ (here } \epsilon_T = 0), \\ 3 + 3\beta_1 < \tau_1 + \epsilon_T & \text{if } \beta_2 = \beta_1 \text{ (here } \epsilon_T = 1). \end{cases}$$

  Fig. 5b1) shows an example of this case with $\beta_1 = 3$ and $\beta_2 = 2$; here $\epsilon_T = 0$, $|T| = 10$, $\tau_1 + \epsilon_T = 12$, $\Delta_{1,2} = 11$, $\Delta_{2,1} = 10$, $\Delta_{1,3} = 0$. Fig. 5b2) shows another example of this case with $\beta_1 = \beta_2 = 3$; here $\epsilon_T = 1$, $|T| = 11$, $\tau_1 + \epsilon_T = 13$, $\Delta_{1,2} = \Delta_{2,1} = 12$, $\Delta_{1,3} = 0$.

- $\Delta_{1,2}$ is the unique maximum: Take 2 subtrees T_1 T_2, with $\alpha_1 = 1$, $\beta_1 \geq 3$, $\beta_2 = 0$, $\alpha_2 > \beta_1 + 1$ and $2\alpha_2 < 3\beta_1 + 2$ (so $\tau_2 < \tau_1$). Then $\Delta_{2,1} = |T| - 2 < |T| < |T| + \beta_1 - 2 = \Delta_{1,2}$ (as $\beta_1 > 2$) and $\tau_1 = 3 + 3\beta_1 < 2 + 2\beta_1 + \alpha_2 = \Delta_{1,2}$ (as $\beta_1 + 1 < \alpha_2$).

  Fig. 5c) shows an example of this case with $\beta_1 = 3$ and $\alpha_2 = 5$; here $|T| = 12$, $\tau_1 + \epsilon_T = 12$, $\Delta_{1,2} = 13$, $\Delta_{2,1} = 10$, $\Delta_{1,3} = 0$.

- $\Delta_{2,1}$ is the unique maximum: Take 2 subtrees T_1 T_2, with $\alpha_2 = 1$, $\beta_2 \geq 3$, $\beta_1 = 0$, $\alpha_1 < 2\beta_2 + 1$ and $2\alpha_1 > 3\beta_2 + 2$ (so $\tau_1 > \tau_2$). Then $\Delta_{1,2} = |T| - 2 < |T| < |T| + \beta_2 - 2 = \Delta_{2,1}$ (as $\beta_2 > 2$) and $\tau_1 = 1 + 2\alpha_1 < 2 + \alpha_1 + 2\beta_2 = \Delta_{2,1}$ (as $\alpha_1 < 2\beta_2 + 1$).

  Fig. 5d) shows an example of this case with $\alpha_1 = 6$ and $\beta_2 = 3$; here $|T| = 13$, $\tau_1 + \epsilon_T = 13$, $\Delta_{1,2} = 11$, $\Delta_{2,1} = 14$, $\Delta_{1,3} = 0$.

- $\Delta_{1,3}$ is the unique maximum: Take 3 subtrees T_1, T_2, T_3, with $\alpha_1 = \alpha_2 = 1$, $\beta_2 < \beta_1 - 4$ (so $\tau_1 > \tau_2$), $\beta_3 = 0$, $\beta_1 < \alpha_3 - 1$ and $2\alpha_3 < 3\beta_2 + 2$ (so $\tau_2 > \tau_3$). Then $|T| = 6 + \beta_1 + \beta_2 + \alpha_3$; $\tau_1 = 3 + 3\beta_1$; $\Delta_{1,2} = 3 + 2\beta_1 + \beta_2$; $\Delta_{2,1} = 3 + \beta_1 + 2\beta_2$; $\Delta_{1,3} = 2 + 2\beta_1 + \alpha_3$. So $|T| < \Delta_{1,3}$ (as $\beta_2 < \beta_1 - 4$), $\tau_1 < \Delta_{1,3}$ (as $\beta_1 < \alpha_3 - 1$) and $\Delta_{1,2} < \Delta_{2,1} < \Delta_{1,3}$ (as $\beta_2 < \beta_1 < \alpha_3 - 1$).

  Fig. 5d) shows an example of this case with $\beta_1 = 15$, $\beta_2 = 11$ and $\alpha_3 = 17$; here $|T| = 43$, $\tau_1 + \epsilon_T = 48$, $\Delta_{1,2} = 44$, $\Delta_{2,1} = 44$, $\Delta_{1,3} = 49$.

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 3.

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 2. *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 introduce the following definition.

Definition 10. *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

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

We denote by $\alpha'_i, \beta'_i, \tau'_i$ the parameters of T'_i. In particular (unless the special case $k = 2$, $\beta_1 = 1$, $\alpha_2 = \alpha_1 + 1$, $\beta_2 = 0$, $T_3 = \emptyset$) we have for $i = 1, \ldots, k$:

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

Fact 6. *Assume* $\langle T_1 \dots T_k \rangle$ *and that NOT* ($k = 2$, $\beta_1 = 1$, $\alpha_2 = \alpha_1 + 1$, $\beta_2 = 0$, $T_3 = \emptyset$). *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* $\tau_i = \tau_j + 1$.

Proof. Consider first 1). If $|T_j| = 1$, then $\tau'_j = 0$ and 1) trivially holds; otherwise $\tau'_i \geq \tau_i - 3$ and $\tau'_j \leq \tau_j - 2$; so $\tau'_i \geq \tau'_j$.

Consider now 2) and assume that $T_i \prec T_j$ and $T'_j \prec T'_i$. By Definition 7 we can have four cases:

- $\tau_i = \tau_j$ with $|T_i| > |T_j|$, and $\tau'_j = \tau'_i$ with $|T'_j| > |T'_i|$. This is impossible since it should be both $|T_i| > |T_j|$ and $|T_j| = |T'_j| + 1 > |T'_i| + 1 = |T_i|$.

- $\tau_i = \tau_j$ with $|T_i| > |T_j|$, and $\tau'_j > \tau'_i$. By the algorithm this case can occur only if $\beta_j = 0$ and $\beta_i \geq 1$. By Fact 2 this is impossible.

- $\tau_i > \tau_j$ and $\tau'_j > \tau'_i$. This is impossible by 1).

- $\tau_i > \tau_j$ and $\tau'_j = \tau'_i$ with $|T'_j| > |T'_i|$. We can have both $\tau_i > \tau_j$ and $\tau'_j = \tau'_i$ only if $\tau'_i = \tau_i - 3 = \tau_j - 2 = \tau'_j$. Hence we have $\tau_i = \tau_j + 1$ and $\beta_j = 0 < \beta_i$. Furthermore, $|T'_j| > |T'_i|$ implies $|T_j| > |T_i|$ and $\beta_i \geq 2$ (otherwise, if $\beta_i = 1$ we would get $T'_i = T'_j$). □

Fact 7. *Assume* $\langle T_1 \dots T_k \rangle$ *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* $i = 2, \dots, k$;

iii) $|T_i| \geq \beta_2 + 1$, *for each* $i = 3, \dots, k$.

Proof. Let T' be the tree resulting after $\langle T_1 \dots T_k \rangle$. If $|T_1| = 1$ then $|T_i| = 1$ for each $i = 1, \cdots, k$; hence, i), ii), and iii) hold. We assume then that $|T_1| \geq 2$ which implies $\alpha_1 \geq 1$.

We first prove ii). If $k \geq 4$ then at steps $4, \dots, k$ we did not choose T_1' which was available and so $T_2 \preceq T_3 \preceq \dots \preceq T_k \preceq T_1'$. If $k = 3$, $T_3 \preceq T_1'$ by hypothesis. So we have

$$T_i \preceq T_1' \quad \text{for each} \;\; i = 2, \dots, k.$$

Let $\Delta = 1$ if $\beta_1 \geq 1$ and 0 otherwise .

It can occur either $\tau_i > \tau_1' = \tau_1 - 2 - \Delta$ or $\tau_i = \tau_1' = \tau_1 - 2 - \Delta$ with $|T_i| \geq |T_1'| = |T_1| - 1$.

Hence, w.l.o.g. let $1 \leq \ell \leq k$ be such that

$$\tau_2 \geq \cdots \geq \tau_\ell > \tau_1 - 2 - \Delta$$

and

$$\tau_{\ell+1} = \cdots = \tau_k = \tau_1 - 2 - \Delta \qquad \text{with} \qquad |T_{\ell+1}|, \dots, |T_k| \geq |T_1| - 1. \tag{2}$$

Recalling that $\tau_i \geq \tau_1' \geq 1$ we get

$$\alpha_i \geq 1, \quad i = 1, \dots, k.$$

For any $i = 2, \dots, \ell$ we have $\tau_i = 3\beta_i + 2\alpha_i + 1 \geq 3\beta_1 + 2\alpha_1 - \Delta$. We can then deduce that

$$\beta_i \geq \beta_1 + \frac{2}{3}\alpha_1 - \frac{2}{3}\alpha_i - \frac{1+\Delta}{3}$$

and

$$|T_i| = \beta_i + \alpha_i + 1 \geq \beta_1 + \frac{2}{3}\alpha_1 - \frac{2}{3}\alpha_i - \frac{1+\Delta}{3} + \alpha_i + 1 = \beta_1 + \frac{2}{3}\alpha_1 + \frac{\alpha_i}{3} + \frac{2-\Delta}{3} \geq \beta_1 + \frac{2}{3}\alpha_1 + \frac{2}{3}, \tag{3}$$

where the last inequality holds since $\alpha_i \geq 1$ and $\Delta \leq 1$.

From this, recalling that $\alpha_1 \geq 1$, we get that $|T_i| > \beta_1 + 1$ and ii) holds for any $i \leq \ell$.

For $i = \ell + 1, \dots, k$, we have $|T_i| \geq |T_1| - 1 = \beta_1 + \alpha_1 \geq \beta_1 + 1$. Hence ii) holds for each $i = 2, \dots, k$.

In the same way we can prove iii). Indeed, if $k \geq 4$, then either $T_1' \preceq T_2'$ and so $T_4 \preceq T_1' \preceq T_2'$,

or $T'_2 \preceq T'_1$ and so by Fact 6, $\beta_2 = 0$ and so T'_2 was available at step 4 and so $T_4 \preceq T'_2$. If $k = 3$, $T_3 \preceq T'_2$ by hypothesis; in all the cases, the same proof as above with T_2 instead of T_1 works.

Let us now prove inequality i). By (3) we have that for each $i = 2, \ldots, \ell$

$$|T_i| \geq \beta_1 + \frac{2}{3}\alpha_1 + \frac{2}{3} = |T_1| - \frac{\alpha_1}{3} - \frac{1}{3}\ . \tag{4}$$

Hence, by (2) and (4) we have

$$\begin{aligned} |T| &= \sum_{i=1}^{m} |T_i| \geq \sum_{i=1}^{k} |T_i| = |T_1| + \sum_{i=2}^{\ell} |T_i| + \sum_{i=\ell+1}^{k} |T_i| \\ &\geq |T_1| + (\ell - 1)(|T_1| - \frac{\alpha_1}{3} - \frac{1}{3}) + (k - \ell)(|T_1| - 1) \\ &= k|T_1| - k + 1 - (\ell - 1)\frac{\alpha_1 - 2}{3} \\ &= \tau_1 + (k-3)\beta_1 + (k-2)\alpha_1 - (\ell - 1)\frac{\alpha_1 - 2}{3}. \end{aligned} \tag{5}$$

The function in (5) is decreasing in ℓ. Considering its minimum, at $\ell = k$, we get

$$|T| \geq \tau_1 + (k-3)\beta_1 + (k-2)\alpha_1 - (k-1)\frac{\alpha_1 - 2}{3}.$$

Recalling that $\alpha_1 \geq 1$ and $k \geq 3$, we get the desired bound $|T| \geq \tau_1 + k - 2$. □

The following theorem together with the lower bound of Theorem 3 prove Theorem 2.

Denote by $\mathcal{T}(T)$ the largest time-label assigned by algorithm TREE-labeling to any arc of T.

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

$$\mathcal{T}(T) \leq Max(T) \tag{6}$$

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 (i.e. t is an idle step).

We first analyze the special case of the algorithm in which $m = 2$, $\beta_1 = 1$, $\beta_2 = 0$, $\alpha_2 = \alpha_1 + 1$. 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^0 be the tree resulting after $\langle T_1 T_2 \rangle$, at the third step a path to T_2^0 is set. Hence, the first three

steps of the algorithm are: $\langle T_1 T_2\rangle\langle T_2^0\rangle$.

Let T^1 be the tree resulting after $\langle T_1 T_2\rangle\langle T_2^0\rangle$. Next the algorithm on T proceeds as follows

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

where T^ℓ, for $\ell \geq 2$, is the tree resulting from $T^{\ell-1}$ after the 2 steps $\langle T_1^{\ell-1} T_2^{\ell-1}\rangle$. By the hypothesis of this case, we have

$$\epsilon_T = 0, \quad |T| = |T_1| + |T_2| = 3 + \alpha_1 + \alpha_2 = 3 + 2\alpha_1 + 1 = \tau_1, \quad \text{and } \Delta_{1,2}, \Delta_{2,1} \leq |T|.$$

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

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 $T_1 \preceq T_2 \preceq \ldots \preceq T_m$). 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 $i = 1, \ldots, m$ and $\mathcal{T}(T) = |T| = Max(T)$.

Suppose now that (6) 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 (6) holds for T.

Notice that we are assuming that T does not belong to the special case (i.e., $m = 2$, $\beta_1 = 1$, $\beta_2 = 0$, $\alpha_2 = \alpha_1 + 1$, T_1 is available and T_2 is available at the next step) 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 $\Delta_{1,2} = |T_1| + |T_2| + \beta_1 - 1 \leq |T|$; hence, the first three steps of the algorithm are $\langle T_1 T_2 T_3\rangle$ (including the case $|T_3| = 0$, in which case the third step is idle).

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

$$Max(T') \leq Max(T) - 3. \tag{7}$$

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

To prove (7) we need to know how the subtrees of T evolve in those of T'. To this aim we analize the peculiarities of the subtrees of T and establish the ordering of the subtrees of T' (according to

the relation $\preceq$). In particular we will show that the two first trees obtained by the algorithm or by considering T' are $T'_1 \prec T'_2$ or $T'_2 \prec T'_1$.

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

$$|T| - |T_1| - |T_2| < \beta_1 - 1. \tag{8}$$

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

$$|T_1| < |T_2|. \tag{9}$$

$$|T'| = |T| - \begin{cases} 3 & \text{if } |T_3| > 0 \\ 2 & \text{otherwise} \end{cases}, \qquad \epsilon_{T'} = \epsilon_T = 0 \text{ (since } |T_1| < |T_2|), \qquad \tau'_1 = \tau_1 - 3 \text{ (as } \beta_1 > 1). \tag{10}$$

We note that $T'_1 \neq T'_2$, since by (9) they have different weights. Furthermore, 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. Indeed, in case $T'_2 \prec T'_1$, by Fact 6, we have $\beta_2 = 0$, $\beta_1 \geq 1$ and $\tau_1 > \tau_2$.

If $|T_3| = 0$, then T_1 and T_2 are the only subtrees in T; hence, the possible orderings on the subtrees of T' are: $T'_1 \prec T'_2$, $T'_2 \prec T'_1$.

If $|T_3| = 1$, we have $|T_4| \leq 1$. Hence, by (8) and (9) we get $\tau'_1, \tau'_2 > 1 \geq \tau_4$ and the possible orderings on the subtrees of T' are: $T'_1 \prec T'_2 \prec T'_4 = T_4$, $T'_2 \prec T'_1 \prec T'_4 = T_4$.

Now, let $|T_3| > 1$. By (8) and Fact 1, we get

$$\tau_3 \leq 3|T_3| - 3 \leq 3(|T| - |T_1| - |T_2|) - 3 \leq 3(\beta_1 - 2) - 3 \leq \tau_1 - 9 - (2\alpha_1 + 1),$$

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

$$\tau_3 \leq \tau_1 - 12 = \tau'_1 - 9. \tag{11}$$

Moreover, by (8) and (9) we have

$$|T_2| \geq |T_1| + 1 \geq \beta_1 + \alpha_1 + 2 \geq \beta_1 + 3 > (|T| - |T_1| - |T_2|) + 4 \geq |T_3| + |T_4| + 4; \tag{12}$$

From (11) and (12), we obtain that:

$$T'_1 \prec T'_3, \quad T'_2 \prec T'_3;$$

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 (12).

If $|T_4| = 0$ then $|T'_4| = 0$ and we get that the only possible orderings on the the subtrees of T' are:

$$T'_1 \prec T'_2 \prec T'_3, \quad T'_2 \prec T'_1 \prec T'_3, \tag{13}$$

So let now $|T_4| \geq 1$. By (12) and Fact 1, we get

$$\tau_2 \geq 2|T_2| - 1 \geq 2(|T_3| + |T_4|) + 7 \geq 4\min\{|T_3|, |T_4|\} + 7. \tag{14}$$

Since $|T_3| > 1$ and $|T_4| \geq 1$, by Definition 6 we trivially have $4|T_4| > \tau_4$ and $4|T_3| > \tau_3 \geq \tau_4$; by (14) we get

$$\tau_2 \geq \tau_4 + 8. \tag{15}$$

From (11) and (15) and recalling that $\tau_1 \geq \tau_2$, we obtain that in the tree T', resulting after $\langle T_1 T_2 T_3 \rangle$:

$$T'_1 \prec T'_3, \quad T'_1 \prec T'_4 = T_4, \quad T'_2 \prec T'_3, \quad T'_2 \prec T'_4 = T_4.$$

Hence, by the definition of $\prec$ (cfr. Definition 7), we get that the only possible orderings on the the subtrees of T' are:

$$T'_1 \prec T'_2 \prec T'_3, \quad T'_1 \prec T'_2 \prec T'_4, \quad T'_2 \prec T'_1 \prec T'_3, \quad T'_2 \prec T'_1 \prec T'_4. \tag{16}$$

Hence, after the first 3 steps, the algorithm on T proceeds as on input T'. To compute $Max(T')$, we will use (10) and distinguish two cases:

In case $T'_1 \prec T'_2$, then it holds

$$\Delta'_{1,2} = \Delta_{1,2} - 3, \qquad \Delta'_{2,1} = \begin{cases} \Delta_{2,1} - 3 & \text{if } \beta_2 > 0 \\ |T'_2| + |T'_1| - 1 < |T'| & \text{if } \beta_2 = 0 \end{cases};$$

furthermore by (12) we have

$$\Delta'_{1,3} < \Delta_{1,2} - 3 \ \text{ if } |T'_3| \geq 1 \qquad \text{and} \qquad \Delta'_{1,4} < \Delta_{1,2} - 3 \ \text{ if } |T'_4| \geq 1.$$

and

$$\begin{aligned} Max(T') &= \max\{|T'|,\ \tau'_1 + \epsilon_{T'},\ \Delta'_{1,2},\ \Delta'_{2,1},\ \Delta'_{1,j}(j = 3, 4)\} \\ &= \max\{|T| - 3,\ \tau_1 - 3,\ \Delta_{1,2} - 3,\ \Delta_{2,1} - 3\} \\ &= Max(T) - 3. \end{aligned}$$

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

$$\Delta'_{1,2} = \Delta_{1,2} - 3, \qquad \Delta'_{2,i} = |T'_2| + |T'_i| + \beta'_2 - 1 = |T'_2| + |T'_i| - 1 < |T'| \ (i = 1, 3, 4),$$

where $\Delta'_{2,3}$ (resp. $\Delta'_{2,4}$) is to be considered when $|T'_3| \geq 1$ (resp. $|T'_4| \geq 1$). Hence

$$\begin{aligned} Max(T') &= \max\{|T'|,\ \tau'_2 + \epsilon_{T'},\ \Delta'_{2,1},\ \Delta'_{1,2},\ \Delta'_{1,j}(j = 3, 4)\} \\ &= \max\{|T| - 3,\ \tau_1 - 3,\ \Delta_{1,2} - 3, \} \\ &= Max(T) - 3. \end{aligned}$$

Case 2: $Max(T) = \Delta_{2,1} > \max\{\tau_1 + \epsilon_T, |T|\}$**.**

We first notice that by definition of $\Delta_{2,1}$ and using $\Delta_{2,1} > |T|$, we get

$$2 \leq |T| - |T_1| - |T_2| + 2 \leq \beta_2, \tag{17}$$

Using Fact 4 and $\Delta_{2,1} > \tau_1 \geq \tau_2$ we get

$$|T_2| < |T_1|.$$

Moreover, since $\Delta_{2,1} > \tau_1 + \epsilon_T$, we get

$$|T_1| + \beta_1 < |T_2| + \beta_2,$$

which also implies $T_1 \neq T_2$ and

$$\epsilon_T = 0.$$

Finally, from (17) we have $|T_3| \leq |T| - |T_1| - |T_2| \leq \beta_2 - 2$; from this and Fact 1 it follows

$$\tau_3 \leq 3|T_3| - 3 \leq 3\beta_2 - 9 \leq \tau_2 - 9 \quad \text{if } |T_3| > 1, \qquad \tau_3 = 1 < \beta_2 \leq \tau_2 - 3 \quad \text{if } |T_3| = 1. \tag{18}$$

We distinguish now two subcases on the value of β_1, this will lead to prove (6).

- $\beta_1 \geq 1$.

  The first tree steps of the algorithm are therefore $\langle T_1 T_2 T_3 \rangle$ (as in Case 1, if $|T_3| = 0$ then the third step is idle). Let T' be the tree resulting after $\langle T_1 T_2 T_3 \rangle$.

  From (17) and recalling that $\tau'_1 = \tau_1 - 3 \geq \tau_2 - 3 = \tau'_2$, we obtain that in the tree T':

  $$T'_1 \prec T'_2;$$

  furthermore, recalling that $\tau'_4 = \tau_4 \leq \tau_3$ (in the case $|T_4| \neq 0$) and by using (18) we have that if $|T_3| \geq 1$ then

  $$T'_2 \prec T'_3, T'_4.$$

  Hence, after the first 3 steps the algorithm on T proceeds as on input T'. For T' we have:

  $$T'_1 \prec T'_2 \prec T'_3 \qquad \text{or} \qquad T'_1 \prec T'_2 \prec T'_4 = T_4 \prec T'_3.$$

  Moreover,

  $$|T'| = |T| - \begin{cases} 3 & \text{if } |T_3| > 0 \\ 2 & \text{otherwise} \end{cases}, \qquad \epsilon_{T'} = \epsilon_T = 0 \text{ (since } |T_1| \neq |T_2|), \qquad \tau'_1 = \tau_1 - 3 \text{ (since } \beta_1 > 0).$$

  $$\Delta'_{1,2} = \Delta_{1,2} - 3, \qquad \Delta'_{2,1} = \Delta_{2,1} - 3,$$

  $$\Delta'_{1,3} = \Delta_{1,3} - 3 \qquad \text{if } |T_3| > 0;$$

similarly, if $T_4 \prec T_3'$, from Fact 5 one has

$$\Delta'_{1,4} = \Delta_{1,4} - 2 \leq \max\{|T|, \tau_1 + \epsilon_T\} - 3 \leq \Delta_{2,1} - 4.$$

Summarizing, it holds $Max(T') = Max(T) - 3$. Therefore,

$$\mathcal{T}(T) = 3 + \mathcal{T}(T') \leq 3 + Max(T') = Max(T).$$

- $\beta_1 = 0$.

  The first two steps of the algorithm are $\langle T_1 T_2 \rangle$. Let T' be the tree resulting after $\langle T_1 T_2 \rangle$. Recalling that $\beta_2 \geq 2$ (cfr. (17)) and that if $|T_3| > 0$ then $\tau_1 - 2 > \tau_3$ (cfr. (18)), we obtain that the first 4 steps of the algorithm are

  $$\langle T_1 T_2 \rangle \langle T'_1 T'_3 = T_3 \rangle.$$

  (where if $|T_3| = 0$ then the fourth step is idle). Let T'' be the tree resulting after $\langle T_1 T_2 \rangle \langle T'_1 T'_3 \rangle$, Using (18) we have

  $$\tau''_4 = \tau_4 \leq \tau_3 \leq \tau_2 - 9 = \tau'_2 - 6 = \tau''_2 - 6 \quad \text{if } |T_3| > 1, \qquad \tau''_4 = \tau_4 = 1 \leq \tau''_2 - 2 \quad \text{if } |T_3| = 1. \tag{19}$$

  Hence, even if $|T_3| > 0$ (and eventually $|T_4| > 0$), after the first 4 steps the algorithm on T proceeds as on input T'' where the subtrees with largest shade are T''_1 and T''_2 with

  $$T''_1 \prec T''_2 \qquad \text{or} \qquad T''_2 \prec T''_1$$

  followed eventually by T''_3 and T''_4 in some order. Moreover,

  $$|T''| = |T| - \begin{cases} 4 & \text{if } |T_3| > 0 \\ 3 & \text{otherwise} \end{cases}, \quad \epsilon_{T''} = \epsilon_T = 0 \text{ (since } \beta_2 \geq 2, \beta_1 = 0), \quad \tau''_1 = \tau_1 - 4 \text{ (since } \beta_1 = 0),$$

  Therefore $|T''| \leq \Delta_{2,1} - 4 = Max(T) - 4$ and $\tau''_1 = \tau_1 - 4 < \Delta_{2,1} - 4 = Max(T) - 4$.

Moreover, if $T_2'' \prec T_1''$

$$\tau_2'' = \tau_2 - 3 \leq \tau_1 - 3 \leq \Delta_{2,1} - 4 = Max(T) - 4.$$

Finally,

$$\Delta_{1,2}^{''}, \Delta_{1,3}^{''}, \Delta_{1,4}^{''} < |T''| \text{ (since } \beta_1 = 0), \qquad \Delta_{2,1}^{''} = \Delta_{2,1} - 4,$$

and, by Fact 3,

$$\Delta_{2,3}'', \Delta_{2,4}^{''} \leq |T_2''| + |T_3''| + |T_4''| + \beta_2'' - 1 \leq |T_2''| + |T_3''| + |T_4''| + |T_1''| - 3 < |T''|.$$

Notice that $\Delta_{1,3}''$ and $\Delta_{2,3}''$ are to be considered if $|T_3''| > 0$, and $\Delta_{1,4}''$ and $\Delta_{2,4}''$ if $|T_4| > 0$.

Summarizing, it holds $Max(T'') \leq \Delta_{2,1} - 4 = Max(T) - 4$. Therefore,

$$\mathcal{T}(T) = 4 + \mathcal{T}(T'') \leq 4 + Max(T'') \leq Max(T).$$

Case 3: $Max(T) = \Delta_{1,3} > \max\{\tau_1 + \epsilon_T, |T|\}$**.**

In this case we have $\beta_1 > 1$, otherwise $\Delta_{1,3} \leq |T|$; hence, the first tree steps of the algorithm are

$$\langle T_1 T_2 T_3 \rangle.$$

By definition of $\Delta_{1,3}$ and using Fact 4, we get like in Case 1

$$|T| - |T_1| - |T_3| < \beta_1 - 1, \tag{20}$$

$$|T_1| < |T_3|. \tag{21}$$

By (20) and Fact 1, we get

$$\tau_3 \leq \tau_2 \leq 3|T_2| - 3 \leq 3(|T| - |T_1| - |T_3|) - 3 \leq 3\beta_1 - 9 \leq (\tau_1 - 3) - 9 = \tau_1 - 12.$$

from which it follows

$$\tau'_2 \leq \tau_2 \leq \tau_1 - 12 \leq \tau'_1 - 9, \qquad \tau'_3 \leq \tau_3 \leq \tau_2 \leq \tau'_1 - 9. \tag{22}$$

Moreover, if $|T_4| \geq 1$ then by (20) and (21) we have

$$|T_3| \geq |T_1| + 1 \geq \beta_1 + \alpha_1 + 2 \geq \beta_1 + 3 \geq (|T| - |T_1| - |T_3|) + 5 \geq |T_2| + |T_4| + 5;$$

which, by Fact 1 (recall that $\beta_1 > 1$ implies $|T_1| > 3$ and so by (21), we have $|T_3| > 4$), implies

$$\tau_3 \geq 2|T_3| - 1 \geq 2(|T_2| + |T_4|) + 9 \geq 4\min\{|T_2|, |T_4|\} + 9.$$

By Definition 6, we trivially have $4|T_4| \geq \tau_4$ and $4|T_2| \geq \tau_2 \geq \tau_4$; then we get

$$\tau_2 \geq \tau_3 \geq \tau_4 + 9. \tag{23}$$

Let T' be the tree resulting after $\langle T_1 T_2 T_3 \rangle$. Recalling that $\beta_1 \geq 2$ and using (22) and (23), we have

$$\tau'_1 = \tau_1 - 3, \qquad \tau_4 < \tau'_2 \leq \tau_2 \leq \tau'_1 - 9, \qquad \tau_4 < \tau'_3 \leq \tau_3 \leq \tau_2 \leq \tau'_1 - 9. \tag{24}$$

From this we obtain that the only possible orderings of the first subtrees of T' are:

$$T'_1 \prec T'_2 \preceq T'_3, \qquad T'_1 \prec T'_3 \prec T'_2.$$

We notice that both sequences of steps $\langle T_1 T_2 T_3 \rangle \langle T'_1 T'_2 T'_3 \rangle$ and $\langle T_1 T_2 T_3 \rangle \langle T'_1 T'_3 T'_2 \rangle$ are possible during the execution of the algorithm on T. Hence, after the first 3 steps, the algorithm on T proceeds as on input T'; indeed if $T'_3 \prec T'_2$, Fact 6 implies $\beta_3 = 0$.
For T' we have:

$$|T'| = |T| - 3, \quad \epsilon_{T'} = \epsilon_T = 0 \text{ (by (22) and (24))}, \quad \tau'_1 = \tau_1 - 3 \text{ (by (24))},$$

$$\Delta'_{1,2} \leq \Delta_{1,2} - 3, \quad \Delta'_{2,1} \leq \begin{cases} \Delta_{2,1} - 3 & \text{if } \beta_2 > 0, \\ |T'| & \text{otherwise.} \end{cases}$$

Furthermore, if $T'_2 \prec T'_3$ then

$$\Delta'_{1,3} \leq \Delta_{1,3} - 3,$$

if otherwise $T'_3 \prec T'_2$ then, by Fact 6, we have that $\beta_3 = 0$ and

$$\Delta'_{3,1} = |T'_3| + |T'_1| - 1 < |T'|.$$

In conclusion, $Max(T') \leq Max(T) - 3$ and

$$\mathcal{T}(T) = 3 + \mathcal{T}(T') \leq 3 + Max(T') \leq Max(T).$$

Case 4: $Max(T) = \max\{\tau_1 + \epsilon_T, |T|\}$**.**

Let k be the largest integer such that the first k distinct steps of the algorithm are

$$\langle T_1 T_2 \dots T_k \rangle,$$

and, letting T' be the tree resulting after $\langle T_1 T_2 \dots T_k \rangle$, it holds $T_{k+1} \not\preceq T'_i$, for some $1 \leq i \leq k$. Therefore the k first trees are different and the $(k+1)$-th has been already chosen.

- First assume that either $k \geq 4$ or $k = 3$ and $T_3 \preceq T'_1, T'_2$. Let the ordering on T' be such that T'_i has the largest shade among all the subtrees of T', followed by T'_j and by some T'_ℓ, i.e.,

$$T'_i \preceq T'_j \preceq T'_\ell.$$

Moreover, we have that during the execution of the algorithm on T, any sequences of steps $\langle T_1 T_2 \dots T_k \rangle \langle T'_i T'_j \rangle$ is possible, for any $1 \leq i \leq k-1$ and $j \neq i$. Indeed

a) Any of the subtrees $T'_1, T'_2, \dots, T'_{k-2}$ is available at step $k+1$ and i can assume any value among $1, \dots, k-2$. Moreover, if $T'_{k-1} \prec T'_1$ then Fact 6 implies $\beta_{k-1} = 0$; hence T'_{k-1} is available at step $k+1$ and $i = k-1$ can hold.

b) if either $T'_k \prec T'_1$ or $T'_k \prec T'_2$ then by Fact 6 we have $\beta_k = 0$, and this implies that T'_k is available at step $k+2$ and j can assume value k.

We now distinguish two cases according to the value of i.

- Let $i \leq k-1$.

  By a) and b), we have that after the first k steps $\langle T_1 T_2 \dots T_k \rangle$, the algorithm on T proceeds as on input T'. For T' we have:

$$|T'| = |T| - k, \quad \tau'_i = \tau_i - 2 = \tau_1 - 3 \quad \text{if } i \geq 2 \text{ (by Fact 6)},$$

$$\tau'_1 = \tau_1 - 3 \quad \text{if } \beta_1 \geq 1 \qquad \tau'_1 = \tau_1 - 2 \quad \text{if } \beta_1 = 0.$$

  furthermore,

$$\beta_1, \beta_2 < |T_3|, \dots, |T_k| \;\; ; \;\; \beta_1 < |T_2| \;\; \text{(by Fact 7)},$$
$$\beta_i = \beta_j = 0 \quad \text{when } 2 \leq i \leq k-1 \text{ and } 3 \leq j \leq k \text{ (by Fact 6 since } T'_i \preceq T'_j \prec T'_1 \text{ or } T'_2),$$
$$\beta_j \leq |T_1| - 2, |T_2| - 2 \quad \text{for } j > k \text{ (by Fact 3 since } T_1, T_2 \preceq T_j).$$

  These inequalities imply that

$$\Delta'_{i,j}, \Delta'_{j,i}, \Delta'_{i,\ell} \leq |T'|.$$

  Hence,

$$\begin{aligned} Max(T') &= \max\{|T'|, \tau'_i + \epsilon_{T'}\} \\ &\leq \begin{cases} \max\{|T| - k, \tau_1 - 3 + \epsilon_{T'}\} & \text{if } i \geq 2 \text{ or } i = 1 \text{ and } \beta_1 \geq 1, \\ \max\{|T| - k, \tau_1 - 2 + \epsilon_{T'}\} & \text{if } i = 1 \text{ and } \beta_1 = 0. \end{cases} \end{aligned}$$

  By i) of Fact 7, we have that $|T| - k \geq \tau_1 - 2$; therefore, $Max(T') = |T| - k$ unless $i = 1$, $\beta_1 = 0$, and $\epsilon_{T'} = 1$.

  Let us consider then $i = 1$, $\beta_1 = 0$, and $T'_1 = T'_j$ (so $T_1 = T_j$). This implies that

$$|T| \geq |T_1| + \ldots + |T_j| + \ldots + |T_k| \geq 2|T_1| + k - 2 = 2(\alpha_1 + 1) + k - 2 = 2\alpha_1 + k = \tau_1 + k - 1$$

  and $Max(T') = |T| - k$, holds also in this case.

Using the inductive hypothesis on T', we can then conclude that

$$\mathcal{T}(T) = k + \mathcal{T}(T') \leq k + Max(T') \leq |T| \leq Max(T).$$

- Let $i = k$.

  In this case we have $T'_k \prec T'_1, T'_2, \ldots, T'_{k-1}$. Hence, by Fact 6 we get $\beta_1, \ldots, \beta_{k-1} \geq 1$ and $\beta_k = 0$ that imply $T'_1 \preceq \ldots \preceq T'_{k-1}$. Since a path to T'_k cannot be set at step $k+1$, we obtain that the first $k+1$ steps of the algorithm are

  $$\langle T_1 T_2 \ldots T_k \rangle \langle T'_1 \rangle.$$

  Let T'' be the tree resulting after $\langle T_1 T_2 \ldots T_k \rangle \langle T'_1 \rangle$. After the first $k+1$ steps, the algorithm on T proceeds as on input T'' where

  $$T''_k \prec T''_j \preceq T''_l \quad \text{(and we have } j = 2 \text{ or } j = k+1)$$

  i.e., the subtree with largest shade is T''_k followed by T''_j, T''_l in this order.

  Moreover,

  $$|T''| = |T| - k - 1, \qquad \tau''_k = \tau'_k = \tau_k - 2 = \tau_1 - 3 \text{ (by Fact 6)} .$$

  We also have $\epsilon_{T''} = 0$. Indeed, if $j = 2$, then $T''_k \prec T''_2$ and if $j = k+1$, $T''_{k+1} = T''_k$ implies $T''_{k+1} \prec T'_1$ and therefore T''_{k+1} should have been chosen at step $k+1$ contradicting the definition of k.

  $$\Delta''_{k,j}, \Delta''_{k,l} < |T''| \text{ (since } \beta_k = 0), \qquad \Delta''_{j,k} \leq |T''_k| + |T''_j| + |T''_1| - 3 \leq |T''| - 3 \text{ (by Fact 3)}$$

  Hence, by using i) of Fact 7 we get $Max(T'') = \max\{|T''|, \tau''_k + \epsilon_{T''}, \Delta''_{k,j}, \Delta''_{j,k}, \Delta''_{k,l}\} \leq \max\{|T| - k - 1, \tau_1 - 3\} = |T| - k - 1 = |T''|$ and

  $$\mathcal{T}(T) = k + 1 + \mathcal{T}(T'') \leq k + 1 + Max(T'') \leq k + 1 + |T''| = |T| \leq Max(T).$$

- Assume now that $k = 3$ and either $T'_1 \prec T_3$ or $T'_2 \prec T_3$

(here, eventually $T_3 = \emptyset$, but excluding the special base case $T_3 = \emptyset$, $\beta_1 = 1$, $\beta_2 = 0$, $\alpha_2 = \alpha_1 + 1$). Suppose that $\beta_1 = 0$: since $k = 3$, we can deduce that $T_3 \prec T'_1$. Therefore, the hypothesis of this case implies $T'_2 \prec T_3 \prec T'_1$. Applying Fact 6 to T_1 and T_2, we get to the contradiction $\beta_1 > 0$. Hence, throughout this case we can assume

$$\beta_1 \geq 1 \quad \text{and} \quad \tau'_1 = \tau_1 - 3.$$

We will distinguish two subcases according $T_3 = \emptyset$ or not.

- Subcase A : $T_3 \neq \emptyset$ and so $|T'| = |T| - 3$

  After $\langle T_1 T_2 T_3 \rangle$, the algorithm on T proceeds as on input T' where T'_i has the largest shade among all the subtrees of T', followed by T'_j and by some T'_l, i.e.,

$$T'_i \preceq T'_j \preceq T'_l. \tag{25}$$

  Here i can be only 1 or 2; indeed T'_1 or $T'_2 \prec T'_3$ (by the hypothesis of this case) and, if $|T_4| > 0$ then $T'_4 \prec T'_1$ and $T'_4 \prec T'_2$ imply that T'_4 should have been chosen giving $k \geq 4$ a contradiction. Note, that if $i = 2$ then, applying Fact 6 to T_1 and T_2, we get $\beta_2 = 0$ and $\tau'_2 = \tau_2 - 2 = \tau_1 - 3$ and so T'_2 is available.
  Concerning j, it can take any value at most 4 and different from i. Note that if $j = 3$ then T'_3 is available since $T'_3 \prec T'_1$ or T'_2 and, by Fact 6, $\beta_j = 0$; if $j = 4$ then T_4 is obviously available.

  Furthermore,

$$\Delta'_{1,2} = \Delta_{1,2} - 3, \qquad \Delta'_{2,1} \leq \begin{cases} \Delta_{2,1} - 3 & \text{if } \beta_2 \geq 1 \\ |T| - 3 & \text{if } \beta_2 = 0 \end{cases}, \qquad \Delta'_{1,3} \leq \Delta_{1,3} - 3.$$

  Moreover, if $i = 2$ (resp. $j = 3$) then, by Fact 6, $\beta_2 = 0$ (resp. $\beta_3 = 0$), and

$$\Delta'_{2,3} < |T'| - 1, \qquad \Delta'_{3,2} < |T'| - 1, \qquad \Delta'_{3,1} < |T'| - 1.$$

Finally, by Fact 5

$$\Delta'_{i,h} = \Delta_{i,h} - 2 \leq \max\{\tau_1 + \epsilon_T, |T|\} - 3 \qquad \text{for } \ h = j, l \text{ and } h \geq 4,$$

$$\Delta'_{4,i} = \Delta_{4,i} - 2 \leq \max\{\tau_1 + \epsilon_T, |T|\} - 3.$$

Summarizing, in all the cases (for any value of h and h' we can have with the ordering (25)) it holds

$$\Delta'_{h,h'} \leq \max\{|T|, \tau_1 + \epsilon_T\} - 3.$$

Hence,

$$\mathcal{T}(T) = 3 + \mathcal{T}(T') \leq 3 + Max(T') = 3 + \max\{|T'|, \tau'_i + \epsilon_{T'}, \Delta'_{i,j}, \Delta'_{j,i}, \Delta'_{i,l}\} = \max\{|T|, \tau_1 + \epsilon_{T'}\}.$$

and in order to prove $\mathcal{T}(T) \leq Max(T)$, we point out the relation between $\max\{|T|, \tau_1 + \epsilon_{T'}\}$ and $\max\{|T|, \tau_1 + \epsilon_T\} = Max(T)$.

If $\epsilon_{T'} = 0$ or $\epsilon_{T'} = \epsilon_T = 1$ we get $\epsilon_{T'} \leq \epsilon_T$, and thus $\mathcal{T}(T) \leq Max(T)$. So it remains to deal with the case where both $\epsilon_{T'} = 1$ and $\epsilon_T = 0$. We will show that $|T'| > \tau'_i + \epsilon_{T'}$ and so $\max\{|T|, \tau_1 + \epsilon_{T'}\} = |T| = \max\{|T|, \tau_1 + \epsilon_T\}$. Indeed, if $\epsilon_{T'} = 1$ then $\tau'_i = \tau'_j$ and $|T'_i| = |T'_j|$ and this implies that $\beta'_i = \beta'_j$. We now prove that $\beta'_i = \beta'_j = 0$ and so $|T'_i| = |T'_j| = 1 + \alpha'_i$ and $|T'| \geq |T'_i| + |T'_j| \geq 2 + 2\alpha'_i \geq \tau'_i + 1$. Indeed, recalling Fact 6, we have: If $i = 2$ then $\beta_2 = 0$ and this implies $\beta'_2 = 0 = \beta'_j$; if $i = 1$ and $j \geq 3$ then $\beta_j = 0$ and this implies $\beta'_j = 0 = \beta'_1$; if $i = 1$ and $j = 2$ then if $\beta'_2 = \beta'_1 \geq 1$ it implies $\beta_2 = \beta_1$ contradicting $\epsilon_T = 0$ and so $\beta'_2 = 0 = \beta'_1$.

- Subcase B: $T_3 = \emptyset$ and so $|T'| = |T| - 2$.

(Recall that we are excluding the special base case $T_3 = \emptyset$, $\beta_1 = 1$, $\beta_2 = 0$, $\alpha_2 = \alpha_1 + 1$.)

Here the algorithm, having done $\langle T_1, T_2, \emptyset\rangle$, proceeds on T as on input T' with either $T'_1 \preceq T'_2$ or $T'_2 \prec T'_1$ in which case $\beta_2 = 0$.

Furthermore,

$$\Delta'_{1,2} = \Delta_{1,2} - 3, \qquad \Delta'_{2,1} \leq \begin{cases} \Delta_{2,1} - 3 & \text{if } \beta_2 \geq 1 \\ |T| - 3 & \text{if } \beta_2 = 0 \end{cases}$$

and

$$Max(T') \leq \max\{|T'|, \max\{\tau'_1, \tau'_2\} + \epsilon_{T'}, \Delta'_{1,2}, \Delta'_{2,1}\}.$$

We notice now that

$$\epsilon_T = \epsilon_{T'}.$$

Indeed, if $\epsilon_T = 1$ then having done $\langle T_1, T_2, \emptyset\rangle$ implies $\epsilon_{T'} = 1$; supposing $\epsilon_T = 0$ and $\epsilon_{T'} = 1$, since $\beta_1 \geq 1$ we would get $\beta_1 = 1$,$\beta_2 = 0$, $\alpha_2 = \alpha_1 + 1$ and the special case would hold. The above also implies

$$(\beta_1 + \beta_2 \geq 2) \qquad \text{or} \qquad (\beta_1 = 1,\ \beta_2 = 0, \alpha_2 < \alpha_1 + 1)$$

We have then

$$\mathcal{T}(T) \leq 3 + Max(T') \leq \max\{|T| + 1, \tau_1 + \epsilon_T\};$$

and in order to show $\mathcal{T}(T) \leq Max(T)$, it is sufficient to show $|T| + 1 \leq \tau_1 + \epsilon_T$.
Notice that as $\tau_i = 2|T_i| + \beta_i - 1$:

$$2|T| = 2|T_1| + 2|T_2| = \tau_1 + \tau_2 + 2 - \beta_1 - \beta_2.$$

Hence

$$|T|+1 = \frac{\tau_1 + \tau_2}{2} + 2 - \frac{\beta_1 + \beta_2}{2} \leq \begin{cases} \tau_1 & \text{if } \beta_1 + \beta_2 \geq 3 \\ & \text{or } \beta_1 = 2, \beta_2 = 0 \text{ (here } \tau_2 < \tau_1 \text{ holds)} \\ & \text{or } \beta_1 = \beta_2 = 1,\ \epsilon_T = 0 \text{ (here } \tau_2 < \tau_1 \text{ holds)} \\ & \text{or } \beta_1 = 1,\ \beta_2 = 0, \alpha_2 < \alpha_1 + 1 \text{ (here } \tau_2 < \tau_1 - 1 \text{ holds)} \\ \tau_1 + \epsilon_T & \text{if } \beta_1 = \beta_2 = 1,\ \epsilon_T = 1. \end{cases}$$

Recall that the last inequality holds since $|T| + 1$ is an integer and $\tau_2 \leq \tau_1$.

- Finally, consider the last possible case: $k = 2$.

This means that the first two steps of the algorithm are $\langle T_1 T_2\rangle$ and at the third step a path to T_1' can be and is set (otherwise we are in the preceding subcase B). Hence, we have $T_1' \preceq T_3$ whenever $|T_3| > 0$, and

$$\beta_1 = 0, \qquad \tau_1' = \tau_1 - 2.$$

Noticing that, by Fact 6, if $\beta_1 = 0$ then $T'_2 \not\prec T'_1$, we distinguish two cases according to the relation between T'_2 and T_3.

- If $T_3 \prec T'_2$ then $T_3 \neq \emptyset$ and the only possible orderings of the subtrees of the tree T' are:

$$T'_1 \preceq T_3 \preceq T'_2, \qquad T'_1 \preceq T_3 \preceq T_4.$$

For T' we have

$$|T'| = |T| - 2, \qquad \Delta'_{1,2}, \Delta'_{1,3}, \Delta'_{1,4} < |T| - 2 \text{ (since } \beta_1 = 0)$$

$$\Delta'_{3,1} \leq |T_3| + |T_1| + |T_2| - 4 \leq |T| - 4 \text{ (since } \beta_3 \leq \max\{0, |T_2| - 2\} \text{ by Fact 3),}$$

where $\Delta'_{1,4}$ is to be considered only if $|T_4| > 0$.

Hence,

$$Max(T') = \max\{|T'|, \tau'_1 + \epsilon_{T'}\} = \max\{|T| - 2, \tau_1 - 2 + \epsilon_{T'}\}.$$

So if $\epsilon_{T'} \leq \epsilon_T$ then we have $Max(T') \leq Max(T) - 2$.

Suppose now that $\epsilon_{T'} = 1$ and $\epsilon_T = 0$. We have $T'_1 = T_3$, $\alpha_3 = \alpha_1 - 1$, $\beta_1 = 0 = \beta_3$ and $|T| \geq |T_1| + |T_2| + |T_3| = |T_2| + 2\alpha_1 + 1 \geq \tau_1 + 1$; hence,

$$Max(T') = \max\{|T'|, \tau'_1 + \epsilon_{T'}\} = \max\{|T| - 2, \tau_1 - 1\} = |T| - 2 \leq Max(T) - 2.$$

In any case we get

$$\mathcal{T}(T) = 2 + \mathcal{T}(T') \leq 2 + Max(T') \leq Max(T).$$

- If $T'_2 \preceq T_3$. We distinguish various cases according to the values of β_2 and the existence or not of T_3

  - Subcase A : $\beta_2 = 0$.

    Then the algorithm on T proceeds as on input T' starting with T'_1, T'_2

$$|T'| = |T| - 2, \qquad \epsilon_{T'} = \epsilon_T \qquad \Delta'_{1,2} = \Delta'_{2,1} = |T_1| + |T_2| - 3 \leq |T| - 3, \qquad \Delta'_{1,3} \leq |T| - 3.$$

Hence, $Max(T') = \max\{|T'|, \tau'_1 + \epsilon_{T'}\} = \max\{|T| - 2, \tau_1 + \epsilon_T - 2\} = Max(T) - 2$ and

$$\mathcal{T}(T) = 2 + \mathcal{T}(T') \leq 2 + Max(T') = 2 + Max(T) - 2 = Max(T).$$

- - Subcase B : $\beta_2 > 0$ and $T_3 \neq \emptyset$.

Then the first 4 steps of the algorithm are

$$\langle T_1 T_2 \rangle \langle T'_1 T'_3 \rangle$$

where $T'_3 = T_3$. Let T'' be the tree resulting after $\langle T_1 T_2 \rangle \langle T'_1 T'_3 \rangle$.

First notice that in this case $T''_1 \preceq T''_3$ since otherwise by Fact 6 it should be $\beta_1 \geq 2$; furthermore, $T''_2 = T'_2 \preceq T'_3$ since $T'_2 \preceq T_3$ and also $T''_2 = T'_2 \preceq T_4$. Hence, after the first 4 steps the algorithm on T proceeds as on input T''. Here, the possible orderings of the subtrees of the tree T'' start with

$$T''_1 \preceq T''_2, \qquad T''_2 \preceq T''_1, \qquad T''_2 \preceq T_4.$$

For T'' we have

$$|T''| = |T| - 4,$$

$$\Delta''_{1,j} < |T''| \text{ (since } \beta_1 = 0) \qquad \text{for } j \geq 2,$$

$$\Delta''_{2,1} = \Delta_{2,1} - 4,$$

$$\Delta''_{2,j} < |T| - 4 = |T''| \text{ (since } \beta_2 \leq |T_1| - 2 \text{ by Fact 3)} \qquad \text{for } j \geq 3,$$

$$\Delta''_{4,2} < |T| - 4 = |T''| \text{ (since } \beta_4 \leq |T_1| - 2 \text{ by Fact 3).}$$

To bound $Max(T'')$, we distinguish two cases according to the relation between T''_2 and T''_1. First notice that $\epsilon_T = 0$ since $\beta_1 = 0$ and $\beta_2 > 0$.

Let us first consider $T''_1 \preceq T''_2$. Noticing that

$$\epsilon_{T''} = 1 \text{ iff } \beta_2 = 1 \text{ and } \alpha_2 = \alpha_1 - 2$$

and recalling the hypothesis $T_3 \neq \emptyset$, we have

$$\text{if } \epsilon_{T''} = 1 \quad \text{then } |T| \geq |T_1| + |T_2| + |T_3| > |T_1| + |T_2| = 2\alpha_1 + 1 = \tau_1$$

and

$$\tau''_1 + \epsilon_{T''} = \begin{cases} \tau_1 - 3 \leq |T| - 4 & \text{if } \epsilon_{T''} = 1, \\ \tau_1 - 4 & \text{if } \epsilon_{T''} = 0. \end{cases}$$

Hence,

$$\begin{aligned} Max(T'') &= \max\{|T''|, \tau''_1 + \epsilon_{T''}, \Delta''_{2,1}\} \\ &\leq \begin{cases} \max\{|T| - 4, \Delta_{2,1} - 4\} \leq Max(T) - 4 & \text{if } \epsilon_{T''} = 1, \\ \max\{|T| - 4, \tau_1 - 4, \Delta_{2,1} - 4\} \leq Max(T) - 4 & \text{if } \epsilon_{T''} = 0. \end{cases} \end{aligned} \tag{26}$$

Consider now $T''_2 \prec T''_1$. Since $T''_2 \neq T''_1$ we have $\epsilon_{T''} = 0$. Furthermore, since $T_1 \preceq T_2$, $\beta_1 = 0$ and $\beta_2 > 0$ we have $|T_1| > |T_2|$. Hence,

$$\Delta''_{2,1} = |T_1| + |T_2| + \beta_2 - 5 \geq 2|T_2| + \beta_2 - 4 = 3\beta_2 + 2\alpha_2 + 2 - 4 = \tau_2 - 3 = \tau''_2 \tag{27}$$

and

$$Max(T'') = \max\{|T''|, \Delta''_{2,1}\} = \max\{|T| - 4, \Delta_{2,1} - 4\} \leq Max(T) - 4. \tag{28}$$

By (26) and (28) we have

$$\mathcal{T}(T) = 4 + \mathcal{T}(T'') \leq 4 + Max(T'') = Max(T).$$

- - Subcase C: $T_3 = \emptyset$ and $\beta_2 \geq 2$.

After the 4 steps $\langle T_1 T_2 \rangle \langle T'_1 \emptyset \rangle$ the algorithm on T proceeds as on input T'', the possible orderings being

$$T''_1 \preceq T''_2, \qquad T''_2 \preceq T''_1.$$

Since $\beta_2 \geq 2$ we have $\Delta''_{2,1} = \Delta_{2,1} - 4$, $\Delta''_{2,1} = |T''_1| + |T''_2| + \beta_2 - 2 \geq |T''|$, $\Delta''_{1,2} < |T''|$ and $\epsilon_{T''} = \epsilon_T = 0$. So $Max(T'') = \max\{\tau''_1, \tau''_2, \Delta''_{2,1}\}$.

If $T''_1 \preceq T''_2$ then $\tau''_1 = \tau_1 - 4$ and $Max(T'') = \max\{\tau_1 - 4, \Delta_{2,1} - 4\} = \max\{\tau_1, \Delta_{2,1}\} - 4 \leq \max\{|T|, \tau_1\} - 4$ where the last inequality is by the hypothesis of Case 4.

If $T_2'' \prec T_1''$ we have like in the above subcase B the inequality (27), i.e., $\Delta_{2,1}'' \geq \tau_2''$; and, $Max(T'') = \Delta_{2,1}'' = \Delta_{2,1} - 4 \leq \max\{|T|, \tau_1\} - 4$.

Therefore

$$\mathcal{T}(T) = 4 + \mathcal{T}(T'') \leq 4 + Max(T'') = Max(T).$$

- - Subcase D: $T_3 = \emptyset$ and $\beta_2 = 1$.

If $\beta_2 = 1$ and $T_1' \preceq T_2$ then $T_1'' \prec T_2''$ and since $\beta_1'' = \beta_2'' = 0$ we have $\alpha_1'' \geq \alpha_2'' + 1$. Therefore $\tau_1'' = 1 + 2\alpha_1'' \geq 1 + \alpha_1'' + 1 + \alpha_2'' = |T''|$. As $\Delta_{2,1}'' = |T''| - 1$, $Max(T'') = \tau_1'' = \tau_1 - 4$ and

$$\mathcal{T}(T) = 4 + \mathcal{T}(T'') \leq 4 + Max(T'') = \tau_1 \leq Max(T).$$

If $T_2 \prec T_1'$ then after the first step $\langle T_1 \rangle$ on T, we get that the algorithm continues as having in input a tree $\overline{T}$ corresponding to the special case considered in the base if $\alpha_1' = \alpha_2 + 1$ and to Subcase B if $\alpha_1' < \alpha_2 + 1$. For $\overline{T}$ we have: $|\overline{T}| = |T| - 1$, and $Max(\overline{T}) = \tau_2$.

But $\tau_1 \neq \tau_2$; indeed $\tau_1 = \tau_2$ implies $2\alpha_1 = 2\alpha_2 + 3\beta_2$ impossible as $\beta_2 = 1$. So $\tau_1 \geq \tau_2 + 1$.

Hence

$$\mathcal{T}(T) = 1 + \mathcal{T}(\overline{T}) \leq 1 + Max(\overline{T}) \leq \tau_1 \leq Max(T).$$

□

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 do not exist three 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 1); 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).

Acknowledgment

We thank the referees for their very conscientious reading of the manuscript and for their very helpful remarks.

References

[1] J-C. Bermond, R. Corrêa, M. Yu, "Optimal Gathering Protocols on Paths under Interference Constraints", *Discrete Mathematics*, **309** (18), (2009) 5574–5587. (A preliminary version has been presented at CIAC06).

[2] J-C. Bermond, L. Gargano, A. Rescigno, "Gathering with Minimum Delay in Sensor Networks", *Proc. SIROCCO*, June 2008 (LNCS 5058),(2008) 262–276 .

[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), (2006) 165–183.

[4] J-C. Bermond, J. Peters, "Efficient gathering in radio grids with interference", *Proc. AlgoTel'05, Presqu'île de Giens*, (2005) 103–106.

[5] J-C. Bermond, M. Yu, "Optimal gathering algorithms in multi-hop radio tree-networks with interferences", Optimal gathering algorithms in multi-hop radio tree networks with interferences. *Ad Hoc and Sensor Wireless Networks*, **9** (1-2), (2010) 109–128. (A preliminary version has been presented at ADHOCNOW 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**, (2005) 16–18.

[7] V. Bonifaci, R. Klasing, P. Korteweg, A. Marchetti-Spaccamela, L. Stougie, "Data Gathering in Wireless Networks", *Graphs and Algorithms in Communication Networks*, A.Koster and X. Munoz editors, Springer Monograph, (2010) 357-377.

[8] V. Bonifaci, P. Korteweg, A. Marchetti-Spaccamela, L. Stougie, "An Approximation Algorithm for the Wireless Gathering Problem", *Operations Research Letters* **36** (5), (2008) 605-608.

[9] C-Y Chong , S.P. Kumar, "Sensor networks: Evolution, opportunities, and challenges", *Proc. of the IEEE* **91** (8), (2003) 1247-1256.

[10] S. Coleri, P. Varaiya, "Energy Efficient Routing with Delay Guarantee for Sensor Networks", *Wireless Networks* **13** (2007), 679-690.

[11] 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.

[12] P. Floreen, P. Kaski, J. Kohonen, P. Orponen, "Exact and approximate balanced data gathering in energy-constrained sensor networks", *Theor. Comput. Sci.*, 344(1), (2005) 30-46

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

[14] D. Ganesan, R. Cristescu, B. Beferull-Lozano, "Power-efficient sensor placement and transmission structure for data gathering under distortion constraints", *ACM Trans. on Sensor Networks*, 2(2), (2006), 155-181 .

[15] L. Gargano, "Time Optimal Gathering in Sensor Networks", *Proc. SIROCCO 2007* (LNCS 4474), (2007) 7-10.

[16] L. Gargano, A.A. Rescigno, "Collision-free path coloring with application to minimum-delay gathering in sensor networks", *Discrete Applied Mathematics*, 157(8), (2009), 1858-1872.

[17] H. Gupta, V. Navda, S.R. Das,V. Chowdhary, "Vishal Chowdhary: Efficient gathering of correlated data in sensor networks", *ACM Trans. on Sensor Networks*, 4(1), (2008).

[18] L. Gasieniec, I. Potapov, Q. Xin, "Time efficient centralized gossiping in radio networks", *Theor. Comput. Sci.*, 383(1), (2007), 45-58.

[19] 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.

[20] 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.

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

[22] S. Lindsey, C. Raghavendra, "Pegasis: Power-efficient gathering in sensor wireless networks", *Proc. of IEEE Aerospace Conference*, (2002).

[23] S. Lindsey, C. Raghavendra, K.M. Sivalingam, "Data gathering algorithms in sensor networks using energy metrics", *IEEE Trans. on Parallel and Distributed Systems*, **13** (9), (2002) 924-935.

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

[25] C. Shen, C. Srisathapornphat, C. Jaikaeo, "Sensor information networking architecture and applications", *IEEE Personal Communications*, (2001) 52-59.

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

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

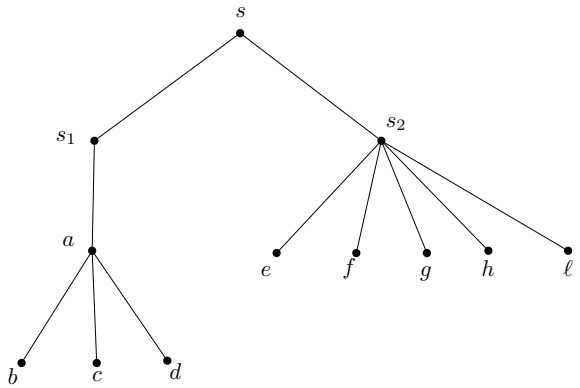

Figure 1: A tree T.

LINE-labeling $(\mathcal{L}, \mathbf{w}, s)$

- Set $\mathcal{P} = \emptyset$, $k = 1$.
- **while** there is a non completed node, **do**
  - Let v be the node at the largest level which is not completed
  - Set a k–path to v in $\mathcal{L}$, call it $\mathbf{p}_v$.
  - Let $\mathcal{P} = \mathcal{P} \cup \{\mathbf{p}_v\}$.
  - Let $w(v) = w(v) - 1$.
  - Set $k = k + \min\{3, \text{level of } v\}$.
- **return** $(\mathcal{P}, L)$, where L is the labeling induced by $\mathcal{P}$.

Figure 2 : The SCF labeling algorithm on a line.

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 (i.e. $t_1, \ldots, t_{i-1} > t$, $T_i \preceq \ldots \preceq T_{|M|}$).

2.2.a **if NOT** [Special case: $|M| = 2$, $i = 1$, $\beta_1 = 1, \alpha_2 = \alpha_1 + 1$, $\beta_2 = 0$, $t_2 \leq t + 1$] **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 $\ell = 1$, where ℓ is the length of $\mathbf{p}$, then T_i is completed and $M = M - \{i\}$.
- If $\ell > 1$, then $t_i = t + \min\{3, \ell\}$, and update T_i, i.e.: $\tau_i = \tau_i - \min\{3, \ell\}$,

$$\alpha_i = \alpha_i - \begin{cases} 1 & \text{if } \ell = 2 \\ 0 & \text{otherwise} \end{cases}, \beta_i = \beta_i - \begin{cases} 1 & \text{if } \ell \geq 3 \\ 0 & \text{otherwise} \end{cases}.$$

- $t = t + 1$.

2.2.b **else** *[Special case: $|M| = 2$, $i = 1$, $\beta_1 = 1, \alpha_2 = \alpha_1 + 1$, $\beta_2 = 0$, $t_2 \leq t + 1$]*

- 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 $\mathbf{q}_2$
- $\mathcal{P} = \mathcal{P} \cup \{\mathbf{p}, \mathbf{q}_1, \mathbf{q}_2\}$.
- **for** $i = 1$ **to** α_1

  Set a $(t + 2i + 1)$-path to T_1 and call it $\mathbf{p}$

  Set a $(t + 2i + 2)$-path to T_2 and call it $\mathbf{q}$

  $\mathcal{P} = \mathcal{P} \cup \{\mathbf{p}, \mathbf{q}\}$.
- Set a $(t + 2\alpha_1 + 3)$-path to s_1 and call it $\mathbf{p}$
- $\mathcal{P} = \mathcal{P} \cup \{\mathbf{p}\}$
- $M = \emptyset$

2.3 **else** $t = t + 1$.

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

Figure 3 : The SCF labeling algorithm on trees

t	subtrees' ordering	k	algorithm's point	t-path	t_1	τ_1	α_1	β_1	t_2	τ_2	α_2	β_2
					1	12	1	3	1	11	5	0
1	$T_1 \prec T_2$	1	2.2.a	(s, s_1, a, b)	4	9	1	2				
2	$T_2 \prec T_1$	2	2.2.a	(s, s_2, e)					4	9	4	0
3												
4	$T_2 \prec T_1$	2	2.2.a	(s, s_2, f)					6	7	3	0
5	$T_1 \prec T_2$	1	2.2.a	(s, s_1, a, c)	8	6	1	1				
6	$T_2 \prec T_1$	2	2.2.a	(s, s_2, g)					8	5	2	0
7												
8	$T_1 \prec T_2$	1	2.2.b	(s, s_1, a, d)								
9		2		(s, s_2)								
10		2		(s, s_2, h)								
11		1		(s, s_1, a)								
12		2		(s, s_2, l)								
13		1		(s, s_1)								

Figure 4 : The paths set, step by step, by the TREE-coloring algorithm

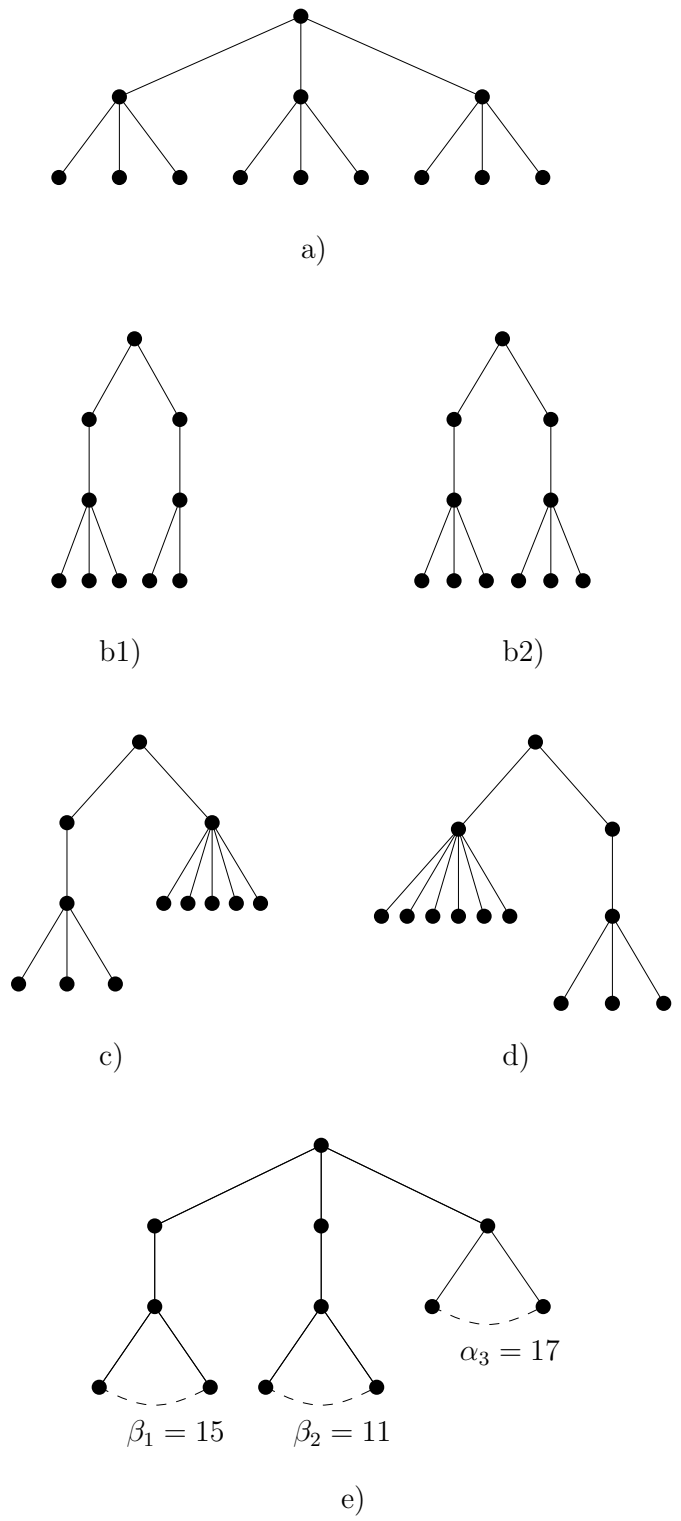

Figure 5: Trees for which exactly one among $|T|, \tau_1 + \epsilon_T, \Delta_{1,2}, \Delta_{2,1}, \Delta_{1,3}$ is the exact bound.