Optimal gathering algorithms in multi-hop radio tree-networks with interferences

Jean-Claude Bermond⋆ and Min-Li Yu⋆⋆

[1] MASCOTTE, joint project CNRS-INRIA-UNSA,
2004 Route des Lucioles, BP 93, F-06902 Sophia-Antipolis, France
bermond@sophia.inria.fr
[2] University College of the Fraser Valley, Department of Mathematics and Statistics,
Abbotsford, BC, Canada V2S 4N2 joseph.yu@ucfv.ca

Abstract. We study the problem of gathering information from the nodes of a multi-hop radio network into a pre-defined destination node under the interference constraints. In such a network, a message can only be properly received if there is no interference from another message being simultaneously transmitted. The network is modeled as a graph, where the vertices represent the nodes and the edges, the possible communications. The interference constraint is modeled by a fixed integer $d_I \geq 1$, which implies that nodes within distance d_I in the graph from one sender cannot receive messages from another node. In this paper, we suppose that it takes one unit of time (slot) to transmit a unit-length message. A step (or round) consists of a set of non interfering (compatible) calls and uses one slot. We present optimal algorithms that give minimum number of steps (delay) for the gathering problem with buffering possibility, when the network is a tree, the root is the destination and $d_I = 1$. In fact we study the equivalent personalized broadcasting problem instead.

1 Introduction

1.1 Problem statement

The problem we consider in this paper was motivated by a question asked by France Telecom about "how to provide Internet connection to a village" (see [6]) and is related to the following scenario. Suppose 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

⋆ Partially supported by the CRC CORSO with France Telecom, by the European FET project AEOLUS, and by the INRIA associated team RESEAUXCOM with S.F.U.

⋆⋆ Partially supported by the Natural Sciences and Engineering Research Council of Canada and by the INRIA associated team RESEAUXCOM with S.F.U.

multi-hop wireless network. In this network, the devices communicate exclusively by means of radio transmissions, referred to as *calls*. A call involves a message and two devices, the *sender* and the *receiver*. The communication is subject to the following technological constraints:

Reachability constraint: in order to be reached by a call, the receiver of this call must be within reachability distance of the sender.

Interference constraint: a call may interfere with calls that are in the neighborhood of the receiver, or a message can be properly received only if no other senders are in the neighborhood of the receiver.

t-gathering problem: suppose each device of the network has a piece of information. The t-gathering consists of collecting (gathering) all these pieces of information into a special device t, called the *gathering node*, by the means of calls subject to the two constraints described before. The t-gathering problem is to realize such a constrained gathering without concatenating messages and with the minimum delay.

An equivalent formulation is the so-called

s-personalized broadcast : here a single device (the gateway in the problem of France Telecom) called source s has a different piece of information to broadcast to every other device in the network by the means of calls subject to the two constraints described before. The s-personalized broadcast is to realize such a constrained gathering without concatenating messages and with the minimum delay.

A slight variation of this problem has received much attention in the context of sensor networks. In such networks, each device contains a sensor and the gathering problem corresponds to the situation where information collected at each sensor has to be gathered to a single central device (base station). However, most of the articles are concerned with minimizing the energy consumption and allow aggregation of data. The work which is most related to ours is [11], in which reachability and interference constraints are also assumed, but most of its results apply for the case of directional antennas.

1.2 Model and assumptions

According to the model adopted in [2], the network described above is represented by an undirected graph $G = (V, E)$, where V is the set of nodes, each of which representing a communication device, and E is the set of edges, representing the pairs of nodes involved in possible calls. There is a special pre-defined node s called the source (sink in the gathering case). Let $d_G(u, v)$ indicate the distance in G, defined as the length of a shortest path between u and v. We model the reachability and the interference constraints by two positive integers, respectively $d_T \geq 1$ and $d_I \geq d_T$. An important case is $d_T = 1$, which means that a node is able to communicate only with its neighbors in the graph (or equivalently

G is the communication graph). The second parameter d_I models the interference constraint as follows: if a receiver is within distance d_I from a sender, then this node cannot receive any other message. If u sends a message m to v, then the call (u, v) interferes with every node $w \in V$ such that $d_G(u, w) \leq d_I$. Two calls are said to be *compatible* if they do not interfere with each other (otherwise, they are *incompatible*). More precisely, two calls (s_1, r_1) and (s_2, r_2), for $r_1, r_2, s_1, s_2 \in V$, are compatible if $d_G(s_1, r_2) > d_I$ and $d_G(s_2, r_1) > d_I$. Observe that one of the consequences of the interference constraint is that $s_1 \neq r_2$ and $s_2 \neq r_1$, which implies that a node is not able to send and receive messages simultaneously. A *step* (*round*) is a set of compatible calls. We assume that every occurrence of a call takes one unit of time (or one slot) and involves a one unit-length message.We also assume that buffering is possible in intermediate nodes.

In this paper, our aim is to find efficient algorithms that give optimal solutions for the s-personalized broadcast problem when $d_T = d_I = 1$ and G is a tree.

1.3 Related work

The broadcasting and gossiping problems have been widely studied for wired networks (see [15]), including models that assume no concatenation of messages (see [4]). For radio networks, the case when $d_I = 1$ is studied only for broadcasting in [10, 12] and gossiping in [8, 9, 14]. Note that broadcasting is different from our problem which is personalized broadcasting, as in the process of broadcast, the same information has to be transmitted to all the other nodes and so flooding techniques can be used. Recently the gathering problem has gained much attention. In [2], assuming an arbitrary size of information in each node, a protocol for general graphs with an approximation factor of at most 4 is presented. It is also shown that the problem of finding an optimal gathering protocol does not admit a Fully Polynomial Time Approximation Scheme if $d_I > d_T$, unless P=NP, and is NP-HARD if $d_I = d_T$. In the case where each node has exactly one unit of information to transmit (or to receive which is the case we consider), the problem is NP-HARD if $d_I > d_T$ but the complexity is unknown for $d_I = d_T$. An extension of the problem where messages can be released over time is considered in [7] and a 4-approximation algorithm is presented. In [5], optimal solutions are provided for the two-dimensional square grid with $d_T = 1$. In [1] the case of a path is considered for $d_T = 1$ and any d_I. The problem is solved when the sink (source) is at one end of the path and only partly solved when the sink is in the middle of the path.

As mentioned before, sensor networks have been the subject of many papers. But, most of them deal with minimizing the energy consumption or maximizing the life time of the sensor network. In [11] they minimize the delay but their model is slightly different from ours as each node is equipped with directional antennas and no buffering capacity is available in the nodes. Furthermore they only suppose that a node cannot receive and send simultaneously, and more precisely, this corresponds to the case in our model when $d_T = 1$, interference distance is zero and each node is not allowed to receive more than one message at a time. Under their assumptions, they give optimal (polynomial) gathering protocols for path and tree networks. Their work has been extended to general graphs in [13] for unitary messages. In [3], a companion paper to that one, the same problem as ours is considered, but no buffering is allowed. Finally, another related model can be found in [16], where the authors study the case in which steady-state flow demands between each pair of nodes have to be satisfied.

1.4 Main result

In this paper, we deal with the situation when G is a tree T with N vertices and with a source (or root) s and $d_T = d_I = 1$ which can be viewed as a generalization of the results of [11] and [13]. In their case the only constraint is that a node cannot receive and transmit at the same time (which can be viewed as $d_I = 0$). They proved that the minimum number of steps is either $N-1$ or $2n_1 - 1$ where n_1 is the size of the biggest subtree.

Here we need to consider not only subtrees, but also subsubtrees. Indeed, when $d_I = 1$, two calls in two different branches are incompatible only if they have the same sender. If two calls (s_1, r_1) and (s_2, r_2) in the same path are incompatible and the arcs are in the order: $s, \ldots, s_1, r_1, \ldots, s_2, r_2, \ldots$, then $d(r_1, s_2) \leq 1$. Otherwise two calls in the same path are compatible if they are separated by at least two arcs.

Here we will have roughly three different forms of trees. Either the tree looks like a path with a big sub-sub-tree formed by the vertices at distance ≥ 2 from s, in which case we will need roughly 3 times the size of this big sub-component. Or the tree has only a big component but inside this component the sub-components are somewhat balanced in which case we need roughly 2 times the size of this big component. In the remaining case of a relatively balanced tree (an example being a spider or generalized star) we need $N-1$ steps.

To state more precisely our main result, let assume that $deg(s) = m$. Let r_1, r_2, ..., r_m be the neighbors of s, and T_i be the subtree of T with root r_i, where $1 \leq i \leq m$. The size of T_i is simply $|T_i| = n_i$. Similarly let $r_{i,j}$ be the neighbors of r_i and $T_{i,j}$ be the subtree with root $r_{i,j}$. The size of $T_{i,j}$ will be denoted by $|T_{i,j}| = n_{i,j}$. Furthermore, we will assume that the $T_{i,j}$'s are ordered according to their sizes. So $n_{i,1} = max\ n_{i,j}$

Let $M_i = max\{2n_i - 1, n_i + 2n_{i,1} - 1\}$. For the rest of the paper, subtrees are ordered according to the values of M_i: $M_1 \geq M_2 \geq M_3 \geq \ldots \geq M_m$. In case of equality the order is determined by the sizes.

Theorem 1. *When $d_T = d_I = 1$ and T is a tree, the minimum number of steps to complete a personalized broadcasting (or gathering) is equal to* $\max\{N - 1, M_1 + \epsilon\}$*, where $\epsilon = 1$ if $M_1 = M_2$ and 0 otherwise.*

Although the lower bound is easy to prove and the minimum time can be expressed in a simple formula, in order to obtain optimal algorithms many different situations are needed to be considered and a lot of experiments were performed before the arrival to the final optimal algorithms.

2 Lower bounds and Basic algorithms.

For the rest of the paper we will simply denote by $g(T)$ (instead of $g(T, s, d_T, d_I)$ used in [2]) the minimum number of steps required to complete the personalized broadcast from s (gathering to s) of one unitary message to each node of T under the interference constraint defined by $d_I = 1$.

2.1 Lower bounds.

Proposition 1. $g(T) \geq max\{N-1, M_1 + \epsilon\}$

Proof. We exhibit different sets of incompatible calls which must be scheduled in different steps (or rounds).

Consider the calls on the arcs (s, r_i) and they are all incompatible and there are $N-1$ of them, as this is the number of messages needed to be sent by the source. So $N-1$ is a lower bound for $g(T)$.

Similarly, for each i, the n_i calls on the arc (s, r_i) and the $n_i - 1$ arcs leaving r_i, are all incompatible. Their number is $2n_i - 1$. So $2n_i - 1$ is a lower bound for $g(T)$.

Consider also the following incompatible calls : those on the arc (s, r_i) and there are n_i of them, the $n_{i,1}$ calls on the arc $(r_i, r_{i,1})$, and the $n_{i,1} - 1$ on the arcs leaving $r_{i,1}$. Altogether we have $n_i + 2n_{i,1} - 1$ incompatible calls and this is also a lower bound for $g(T)$.

Hence, M_i and therefore M_1 is a lower bound. If $M_1 = M_2$, then any algorithm starts calling one of r_1 or r_2 only at step 2 or after, and so it needs at least $M_1 + 1$ steps.

In the next subsections, we present algorithms that perform personalized broadcasting, which will give optimal solutions when there is only one subtree and will also be used for the general case, in particular when there are two subtrees, by applying them to each subtree. We describe the algorithms for one subtree T_i rooted in r_i. We call T_i a type 1 subtree if $M_i = 2n_i - 1$. Otherwise, it is called a type 2 subtree.

2.2 CASE 1: T_i is a subtree of type 1.

We first present an algorithm for a type 1 subtree T_i. In this case recall that $M_i = 2n_i - 1$.

Let X^t denote the set of vertices to which the source has sent a message before step t (that is at the end of step $t-1$) and let T_i^t be the subtree obtained from T_i by deleting X^t. Similarly denote by $T_{i,j}^t$ the component obtained from $T_{i,j}$ by deleting the vertices of X^t. Let $n_i^t = |T_i^t|$ and $n_{i,j}^t = |T_{i,j}^t|$.

The idea of the algorithm is the following: the source sends every odd step to r_i a message destinated to a leaf of a big component of T_i, in order to guarantee that at any step there is no component having more than half of the vertices (or $n_{i,j}^t \leq n_i^t/2$). Also in two consecutive odd steps, the source will send to different components of T_i in order to be able to do compatible calls efficiently in even steps in different components. We first describe the algorithm, then use an example to illustrate it and finally we prove that it is valid and takes M_i steps (which is the lower bound as $M_i \geq N - 1 = n_i - 1$).

Algorithm A: Personalized broadcasting for a subtree of type 1

At the beginning $X^1 = \emptyset$ and $T_i^1 = T_i$.

- During an odd step $t = 2k - 1$, $k = 1, 2, \ldots, n_i$

Let T_{i,j_k}^t be the largest component of T_i^t not chosen at the preceding odd step (that is $j_k \neq j_{k-1}$) and let x_k be a leaf in this component. The source s sends the message m_k for x_k on the arc (s, r_i). Then we update $X^{t+1} = X^t \cup x_k$ and $T_i^{t+1} = T_i^t - x_k$.

During the odd steps, both r_i and the $r_{i,j}$ are inactive.

Finally any vertex at distance ≥ 3 from the source forwards immediately the message received at the preceding step except when it is the destination, in which case the message is stored (if it is m_l with destination x_l, then the message is forwarded to its neighbor on the path to x_l).

- During an even step $t = 2k$, $k = 1, 2, \ldots, n_i - 1$

- r_i sends to r_{i,j_k} the message m_k received at step $2k-1$ with the destination x_k in T_{i,j_k}^t.

- $r_{i,j_{k-1}}$ sends the message m_{k-1} (received at step $2k - 2$) to its neighbor on the path to x_{k-1}, except when it is the destination, in which case the message is just stored.

- Any vertex at distance ≥ 3 from the source forwards immediately the message received at the preceding step except when it is the destination, in which case the message is stored.

Example: Table 1 illustrates how algorithm A works when it is applied to the type 1 tree given in Fig. 1.

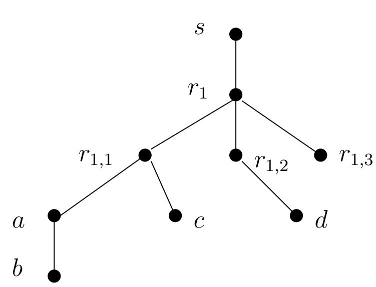

Fig. 1

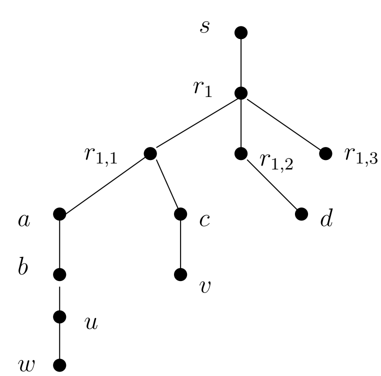

Fig. 2

step	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8
1	$s \to r_1$							
2	$r_1 \to r_{1,1}$							
3		$s \to r_1$						
4	$r_{1,1} \to a$	$r_1 \to r_{1,2}$						
5	$a \to b$		$s \to r_1$					
6		$r_{1,2} \to d$	$r_1 \to r_{1,1}$					
7				$s \to r_1$				
8			$r_{1,1} \to a$	$r_1 \to r_{1,2}$				
9					$s \to r_1$			
10					$r_1 \to r_{1,1}$			
11						$s \to r_1$		
12					$r_{1,1} \to c$	$r_1 \to r_{1,3}$		
13							$s \to r_1$	
14							$r_1 \to r_{1,1}$	
15								$s \to r_1$

Table 1. personalized broadcasting on the tree in Fig.1 with source s using Algorithm A.

Here, $N = 9$, $n_1 = 8$ and $n_{1,1} = 4$. As $M_1 = 15 = 2n_1 - 1 = n_1 + 2n_{1,1} - 1$, it is a type 1 tree. At step 1, s sends a message destinated to a leaf in $T_{1,1}$ (the largest component), for example $x_1 = b$ (we could have chosen c). So $m_1 = m(b)$, the message destinated to b. At step 2, r_1 sends m_1 to $r_{1,1}$. At step 3, s sends a message destinated to a leaf in the largest component different from $T^3_{1,1}$, namely $T^3_{1,2}$ and the only choice is $x_2 = d$. At step 4, r_1 sends to $r_{1,2}$ $m_2 = m(d)$ and $r_{1,1}$ sends m_1 to a (its neighbor on the path to b). At step 5, s sends a message destinated to a leaf in $T^5_{1,1}$ (the largest component), for example $x_3 = a$ (we could have chosen c). Also a, which is at distance 3 from s, forwards m_1 to b where it is stored. The other steps are described in table 1: we have $x_4 = r_{1,2}$ (we could have chosen $r_{1,3}$), $x_5 = c$, $x_6 = r_{1,3}$, $x_7 = r_{1,1}$ and $x_8 = r_1$. Therefore, $m_1 = m(b)$, $m_2 = m(d)$, $m_3 = m(a)$, $m_4 = m(r_{1,2})$, $m_5 = m(c)$, $m_6 = m(r_{1,3})$, $m_7 = m(r_{1,1})$ and $m_8 = m(r_1)$.

Proposition 2. *Algorithm A is valid, i.e. all the calls are compatible.*

Proof. Consider a call with a sender s and it happens in an odd step. As r_i and $r_{i,j}$ are inactive, only the source is sending among the vertices at distance at most 2 from s and so this call is compatible with the others calls whose senders are at distance ≥ 3.

Now consider a call with a sender r_i and it must happen in an even step. Suppose it is a call done at step $2k$ from r_i to r_{i,j_k}. This call is compatible with the other calls in the component T_{i,j_k}, as they involve senders at distance at least 4 from s. Indeed the preceding messages in T_{i,j_k} have been sent at step at most $2k - 4$ from r_i to r_{i,j_k} and at step at most $2k - 2$ from r_{i,j_k} to a neighbor and then forwarded. Therefore they either arrived at the destinations or at a vertex with distance at least 4 from s. They are also compatible with the calls in other components as none of them involve r_i.

If two calls are in different components $T_{i,j}$, then they are compatible as the distance from a sender to a receiver of the other call is at least 3. Finally two calls with senders in the same component $T_{i,j}$ are compatible and this follows from the fact that they are sent by $r_{i,j}$ within two steps differing by at least 4, as the same component cannot be chosen in two consecutive even steps. Because the distance between two such senders is at least 4, the distance between a sender and the other receiver is at least $3 > 1 = d_I$.

Proposition 3. *At the end of the $M_i = 2n_i - 1$ steps of the algorithm A, all the vertices of T_i have received their own messages and so the gathering time is $M_i = 2n_i - 1$.*

Proof. We first prove that at any step there is no component $T^t_{i,j}$ such that $n^t_{i,j} > \frac{n^t_i}{2}$. Indeed, it is true at step $t = 1$ as indeed T_i is type 1, $2n^1_{i,1} \leq n_i$. Suppose that the property is not true and let $t_0 = 2k_0 - 1$ be the first step at which it happens. Then there exists such a component of size strictly bigger than $\frac{n^{t_0}_i}{2}$. Hence, in the two preceding odd steps, this component was the biggest one and it should have been chosen in one of these two steps, and therefore, this component was already of size bigger than half at step $t_0 - 2 = 2k_0 - 3$ or $t_0 - 4 = 2k_0 - 5$ contradicting the choice of k_0.

Therefore at any step $t = 2k - 1$ there is a new vertex x_k to which a message can be sent. Hence, all the messages have been sent by the source at end of step $M_i = 2n_i - 1$.

Consider a message m_k which is sent by s at step $2k - 1$. If $k = n_i$ this is the last message with destination r_i and it arrives at step $2n_i - 1 = M_i$.

Otherwise r_i sends m_k at step $2k$ to r_{i,j_k}. If r_{i,j_k} is its destination, then it arrives at step $2k \leq 2n_i - 2 < M_i$, as $k < n_i$. Otherwise, m_k is sent by r_{i,j_k} on the path to x_k at step $2k + 2$ and then forwarded immediately till it reaches x_k. Let $d(s, x_k)$ be the distance between s and x_k. Note that $d(s, x_k) \geq 3$. The messages with destination on the path from s to x_k are all sent after x_k (otherwise we would have not chosen a leaf contradicting the algorithm). Therefore $k \leq n_i - d(s, x_k) + 1$. Finally m_k is received by x_k at step $2k + d(s, x_k) - 1 \leq 2n_i - d(s, x_k) + 1 \leq 2n_i - 2 = M_i - 1$ as $d(s, x_k) \geq 3$.

2.3 CASE 2: T_i is a subtree of type 2.

Here $M_i = n_i + 2n_{i,1} - 1$. So there is a component $T_{i,1}$ such that $2n_{i,1} > n_i$. The idea consists in considering a set of vertices S_i in this component such that the subtree T^*_i obtained by deleting them is of type 1 and then to apply algorithm A to $T^*_i = T_i - S_i$. For the vertices of S_i note that, in the formula for M_i, they are counted for 3. So we will send the messages destinated to them each 3 steps.

A natural way will be to send to the vertices of S_i during the first $3|S_i|$ steps of the algorithm: the source sends first a message to them at steps $3h$, where $0 \leq h \leq n_i - n^*_i - 1$ and then the message is forwarded immediately till it reaches the destination. This algorithm can be also viewed in an inductive fashion: take a leaf u in $T_{i,1}$; at step 1, the source sends to r_i the message to u and then the message is immediately forwarded; at step 2, (r_i sends it to $r_{i,1}$ and so on); at step 4 we apply the algorithm to the tree $T - u$ using either induction or the algorithm A if $T - u$ is of type 1.

This idea works perfectly for one subtree and will be in fact used later for 3 or more subtrees in Section 4.3. But unfortunately it does not lead to a solution in all the cases. For example suppose we have two subtrees. If T_1 is of type 1, then the source will send every odd step. Assume that T_2 is of type 2 with $M_2 = M_1 - 1$; so the source should first send to it at step 2. But then after 3 steps, the source has to send again at step 5; however, s is in fact busy sending to T_1 in this step.

So we will proceed in a different manner by first sending to vertices in T_i^* using Algorithm A, and then use what we call a 3-step extension to send to the rest of vertices by pushing the messages along some paths. So, messages arrive in the leaves only at the last steps of the algorithm. In fact if one thinks in terms of gathering (where the algorithm is the reverse of that for personalized broadcasting) it is more natural to send first the messages from vertices far away that are those from $S_i = T_i - T_i^*$.

We develop an algorithm that proceeds in 2 phases. In the first phase, each vertex receives an integer label which indicates the step in which this message will be sent by the source in the second phase. Therefore, in the second phase, the source will use the information from the labels given in the previous phase to send the proper message at each step. The algorithm is described below and will then be illustrated by an example. We will prove that it is valid and takes M_i steps (which is the lower bound as $M_i \geq N - 1 = n_i - 1$).

Algorithm B: Personalized broadcasting for a subtree of type 2

More precisely, let S_i be a set of σ_i vertices of $T_{i,1}$ such that, after deletion, we obtain a tree $T_i^* = T_i - S_i$ with $n_i^* = n_i - \sigma_i = |T_i^*|$ vertices. Now $M_i^* = 2n_i^* - 1 = n_i^* + 2n_{i,1}^* - 1$ where $n_{i,1}^* = n_{i,1} - \sigma_i$. Therefore, T_i^* is a type 1 subtree.

Phase 1 : Run the algorithm A on T_i^*, except that the source sends at step $t = 2k - 1$ just a label of value k $(1 \leq k \leq n_i^*)$ (not the message). Then the source sends successively to each node of S_i an unique label (in the range $[n_i^*+1, \ldots, n_i]$) by using σ_i times the following "3-step extension" ($3\sigma_i$ more steps). Order the vertices of $S_i = \{s_{n_i^*+1+h}, 0 \leq h \leq n_i - n_i^* - 1\}$ such that the following property is satisfied: for each h, $s_{n_i^*+1+h}$ is connected to $T^* \cup \{s_{n_i^*+1}, \ldots, s_{n_i^*+h}\}$. Hence there exists a path from s to $s_{n_i^*+1+h}$, where all the nodes except the last one ($s_{n_i^*+1+h}$) have already received a label. Let the vertices of this path be $u_0 = s, u_1 = r_i, u_2 = r_{i,1}, u_3, \ldots, u_{d_h} = s_{n_i^*+1+h}$, where $d_h = d(s, s_{n_i^*+1+h})$.

Do the following 3 steps in any order: in one step, do the compatible calls (u_{3p}, u_{3p+1}), in the next step, do the compatible calls (u_{3p+1}, u_{3p+2}) and in the last one, do the compatible calls (u_{3p+2}, u_{3p+3}).

During each call, each sender (if it is not the source) sends the label it has stored. Therefore at the end of the "3-step extension" each node has the label of its predecessor on the path. The source sends to r_i a new label $n_i^* + 1 + h$. Note that the calls in an extension are compatible with the calls of any other extension as they are done at different steps.

Note also that the order in which we organize the 3 steps has no importance. However for the purpose of clarity and using in theorem 4, we do the steps in an order such that the source is always sending at an odd step as soon as it becomes possible. So we do the calls (u_{3p}, u_{3p+1}) (including the call with the source as a sender) at step $2n_i^* + 3h + \epsilon$, where $\epsilon = 1$ if h is even and 0 if h is odd. Here h ranges from 0 to $\sigma_i - 1 = n_i - n_i^* - 1$. We do the calls (u_{3p+1}, u_{3p+2}) at step $2n_i^* + 3h + (1-\epsilon)$ and the calls (u_{3p+2}, u_{3p+3}) at step $2n_i^* + 3h + 2$. So the source sends at steps $2n_i^* + 1, 2n_i^* + 3, 2n_i^* + 7, \ldots, 2n_i^* + 6q + 1, 2n_i^* + 6q + 3, \ldots$ and is inactive at steps $2n_i^* + 6q + 5$.

At the end of the phase 1 of the algorithm, each node has received exactly one unique integer label ranging from 1 to n_i. Let x_k be the node which has received the value k.

Phase 2: Run the same algorithm again, but in the first part the source sends at step $t = 2k - 1$, $1 \leq k \leq n_i^*$, the message m_k destinated to x_k, and in the extensions at step $2n_i^* + 3h + \epsilon$, where $\epsilon = 1$ if h is even and 0 if h is odd, the message $m_{n_i^*+1+h}$to $x_{n_i^*+1+h}$, where $0 \leq h \leq n_i - n_i^* - 1$. Another way to describe this is that in the steps when the source s sends a message, it is $m(v)$ where v contains the smallest label and $m(v)$ has not been sent.

Example: Consider the type 2 tree given in Fig. 2 obtained by adding three vertices u, v and w and edges (b, u), (u, w) and (c, v) to the tree in Fig.1. Here, $n_1 = 11, n_{1,1} = 7$ and $n_1^* = 8$. Hence, $M_1 = 24 = n_1 + 2n_{1,1} - 1(> 21 = 2n_1 - 1)$. Remember that by deleting the vertices u, v and w, the resulting tree is type 1. Now we illustrate algorithm B by applying it to this tree.

In phase 1, first we apply Algorithm A to the subtree obtained by deleting vertices u, v and w from the given tree (the resulting tree is exactly that of Fig.1), and send a label to each vertex in this subtree, and this takes 15 steps. The resulting labels which are those obtained in the previous example are given in the first row of Table 3. Then 3-step extension is used to extend the labels to the vertices u, v and w. Note that in this process, the labels given in the first part of 15 rounds will be changed. The 3-step extension is illustrated in Table 2. For example, steps 16, 17 and 18 are used to extend the labeling to the vertex u by moving the labels from s to u along the path $(s, r_1, r_{1,1}, a, b, u)$. We need 9 steps to complete the labeling of u, v and w, Table 3 gives the labels of vertices at the end of each 3-step extension in the phase 1 of Algorithm B. The source is not sending at step 21.

Once we have the labels for the vertices, we are able to determine which messages the source should send at different steps. Now we are ready for the second phase of the algorithm. In phase 2, we run again the same algorithm, except this time, instead of labels, at step $t = 2k - 1$, for $1 \leq k \leq 8$, the source sends the message $m(v)$, where the label of the vertex v from the first phase of the algorithm is k. For example, s sends $m_1 = m(w)$ at the first step as $x_1 = w$ or the label of w is 1, and sends $m_2 = m(d)$ at the third step, as $x_2 = d$ or the label of d is 2 and so on. Then s sends at step 17 $m(a)$ as $x_9 = a$, at step 19, $m(r_{1,1})$ as $x_{10} = r_{1,1}$, and at step 23, $m(r_1)$ as $x_{11} = r_1$. Note that the protocol is exactly the same as that of the previous example for the first 15 steps and so they are omitted in the table 4. In fact, the vertices not in $T_{1,1}$ have received

step		
16	$r_1 \to r_{1,1}$	$b \to u$
17	$s \to r_1$	$a \to b$
18	$r_{1,1} \to a$	
19	$s \to r_1$	$c \to v$
20	$r_1 \to r_{1,1}$	
21	$r_{1,1} \to c$	
22	$r_1 \to r_{1,1}$	$b \to u$
23	$s \to r_1$	$a \to b$
24	$r_{1,1} \to a$	$u \to w$

Table 2. 9 steps of 3-step extension to label u, v and w.

	r_1	$r_{1,1}$	$r_{1,2}$	$r_{1,3}$	a	b	c	d	u	v	w
labels after 15 steps	8	7	4	6	3	1	5	2	-	-	-
labels after 18 steps	9	8	4	6	7	3	5	2	1	-	-
labels after 21 steps	10	9	4	6	7	3	8	2	1	5	-
labels after 24 steps	11	10	4	6	9	7	8	2	3	5	1
names of the vertices	x_{11}	x_{10}	x_4	x_6	x_9	x_7	x_8	x_2	x_3	x_5	x_1

Table 3. Labels of vertices after the phase 1 of Algorithm B.

their messages at the end of the first 15 steps (they are the messages $m_2 = m(d)$, $m_4 = m(r_{1,2})$ and $m_6 = m(r_{1,3})$ that have arrived at their destinations). We indicate in the table 4 the steps of transmission of the other messages

Proposition 4. *Algorithm B is valid and uses M_i steps (so $g(T_i) = M_i$).*

Proof. The algorithm B is valid as during each step we have only compatible calls (that is the case for algorithm A applied to T_i^* and then the calls of each step of the extension have been designed to be compatible). At the end of the algorithm each vertex has received its message. In fact, a vertex will receive its message in the first part of the algorithm (before the 3-steps extension) if it is in $T_{i,j}$, where $j \neq 1$, and otherwise, in one of the 3-steps of the last extension. The algorithm uses $2n_i^* - 1$ steps in the first part and then $3\sigma_i$ steps for the extensions. Therefore we have altogether $2n_i^* + 3\sigma_i - 1 = (n_i^* + \sigma_i) + (n_i^* + 2\sigma_i) - 1$ steps. But $n_i^* + \sigma_i = n_i$. By definition of T_i^*, $n_i^* = 2n_{i,1}^* = 2(n_{i,1} - \sigma_i)$ so $n_i^* + 2\sigma_i = 2n_{i,1}$ and so the number of steps is $n_i + 2n_{i,1} - 1 = M_i$.

step	m_1	m_3	m_5	m_7	m_8	m_9	m_{10}	m_{11}
16	$b \to u$				$r_1 \to r_{1,1}$			
17		$a \to b$				$s \to r_1$		
18				$r_{1,1} \to a$				
19			$c \to v$				$s \to r_1$	
20						$r_1 \to r_{1,1}$		
21					$r_{1,1} \to c$			
22		$b \to u$					$r_1 \to r_{1,1}$	
23				$a \to b$				$s \to r_1$
24	$u \to w$					$r_{1,1} \to a$		

Table 4. Last 9 steps of phase 2 of Algorithm B.

3 General Algorithms

We will apply basic algorithms (A or B according to the type of subtrees) first in the case of a single subtree and then of two subtrees. For $m \geq 3$, we will use some other techniques and induction; however we will deal first with some special cases. Recall that subtrees are ordered according to the values of M_i: $M_1 \geq M_2 \geq M_3 \geq \ldots \geq M_m$. In case of equality the order is determined by the sizes.

3.1 Case of one subtree.

In that case we apply directly the basic algorithm to the tree and we get

Theorem 2. *In the case where T consists of one subtree T_1, $g(T) = M_1 > N-1$.*

3.2 Case of two subtrees.

We apply the basic algorithm to the subtree T_1. All the vertices are informed in M_1 steps. We also apply simultaneously the basic algorithm to the subtree T_2, but starting at step 2 ; all the steps are translated by one and therefore all vertices of T_2 are informed in $M_2 + 1$ steps.

Theorem 3. *In the case where T consists of two subtrees T_1 and T_2, $g(T) = max\{M_1, M_2 + 1\}$ (this value is equal to $max\{N - 1, M_1 + \epsilon\}$ where $\epsilon = 1$ if $M_1 = M_2$ and 0 otherwise).*

Proof. Let us first prove that all the calls are compatible. The validity of Algorithm A or B covers the case when two calls belong to the same subtree. That is the case also for the calls having the source as sender; indeed both in algorithm A or B the source is sending only during some odd steps. So here the source sends to r_1 at some odd steps and to r_2 at some even steps. Finally if two calls belong to different subtrees and are not both sent by the source, then the distance between one sender and the other receiver is at least 2.

Altogether the algorithm uses $max\{M_1, M_2 + 1\} = M_1 + \epsilon$ steps. We claim that $M_1 + \epsilon \geq N - 1$, which will prove that the lower bound is attained in that case. The claim is true if $M_1 \geq N-1$. If $M_1 \leq N-2$, then $2n_1 - 1 \leq M_1 \leq N-2$ and $2n_2 - 1 \leq M_2 \leq N - 2$. That implies $n_1 + n_2 \leq N - 1$; but in fact the equalities should hold everywhere as $N - 1 = n_1 + n_2$, that is $n_1 = n_2 = \frac{N-1}{2}$ and $M_1 = M_2 = N - 2$ and therefore $M_1 + \epsilon = N - 1$

3.3 Special Case: $m > 2$ and T_1 and T_2 are of different types, and $M_1 \geq N - 1$ and $M_2 = M_1 - 1$.

Theorem 4. *Suppose T consists of at least 3 subtrees such that T_1 and T_2 are of different types and $M_1 \geq N - 1$ and $M_2 = M_1 - 1$. Then $g(T) = M_1$.*

Proof. As before we apply the basic algorithm to the subtree T_1 and simultaneously the basic algorithm to the subtree T_2, but starting at step 2. Note that the vertices of T_1 and T_2 are informed both at step M_1 (for T_2 they are informed at step $M_2 + 1 = M_1$, as $M_2 = M_1 - 1$). Note also that the source sends to r_1 or r_2 during $n_1 + n_2$ steps. More precisely, if T_1 is of type 2, the source sends all odd steps not of the form $2n_2^* + 6q + 5$ ($\leq M_1$) to r_1 (see the analysis of algorithm B) and all the even steps to r_2 as T_2 is of type 1. Otherwise, if T_1 is of type 1, the source sends all odd steps to r_1 and all the even steps not of the form $2n_2^* + 6q + 6$ ($\leq M_2 + 1 = M_1$) to r_2 as T_2 is of type 2. In both cases the source is not sending neither to r_1 nor to r_2 during $N - 1 - n_1 - n_2$ steps all before M_1 and which are 6 steps apart. We can use these steps to send messages greedily to the other subtrees by choosing a leaf in the remaining tree obtained by deleting T_1, T_2 and the vertices already chosen as destination. Then the message is forwarded immediately as it is received. Therefore a vertex of T_i, $i \geq 3$, at distance d of the source will receive the message at step $t + d - 1$, if it has been sent by the source at step t. If $d = 1$ it arrives at a step $\leq M_1$; if $d > 1$, there are $d - 1$ vertices on the path from s to it which are sent at steps at least $t + 6, \ldots, t + 6(d - 1) \leq M - 1$; so $t + (d - 1) \leq M - 1$. Therefore the algorithm uses M_1 steps which is the lower bound $max\{N - 1, M_1 + \epsilon\} = M_1$ as by hypothesis $M_1 \geq N - 1$ and $\epsilon = 0$.

3.4 General case: $m > 2$

For the case $m > 2$, when we are not in the special case of the preceding theorem 4, we apply induction on N and present algorithms which complete the personalized broadcasting in the number of steps that meet the lower bound. Therefore, the exact number of $g(T)$ is determined. We will suppose that, in our algorithms, the source sends at steps 1 and 2 to two different subtrees. Furthermore, if $M_1 \geq N - 1$ and T_1 is of type 1, the algorithm used to send messages to T_1 is the basic algorithm A (in particular the source will send to r_1 in all odd steps). We assume that $N > 4$ otherwise it is trivial and we will distinguish 3 cases :

Case A: $N - 1 > M_1$;

Case B: $M_1 \geq N - 1$ **and** T_1 **is of type 2**;

Case C: $M_1 \geq N - 1$ **and** T_1 **is of type 1**.

We will also use the following property

Property P : For a tree of type 1, $M_i = 2n_i - 1$ or $n_i = \frac{M_i+1}{2}$. For a tree of type 2, $M_i \leq 3n_i - 3$ as $n_{i,1} \leq n_i - 1$, so $n_i \geq \frac{M_i+3}{3}$. In both cases, if $n_i \geq 2$, $M_i \leq 3n_i - 3$ or $n_i \geq \frac{M_i+3}{3}$.

Algorithm for cases A and B

In both cases, choose in each T_i, $i = 1, 2, 3$ a leaf v_i and let $T'_i = T_i - v_i$ and M'_i be the maximum for T'_i ($M'_i = max\{2|T'_i| - 1, |T'_i| + 2|T'_{i,1}| - 1\}$).

The source sends in the first three steps the messages destinated to $v_i, i = 1, 2, 3$, to r_i, in an order to be given later. Then in the next steps a node which has received a message destinated to $v_i, i = 1, 2, 3$, forwards it immediately in the next step on the path till it arrives at v_i. Starting at step 4, we apply the algorithm to the tree $T' = T - \{v_1, v_2, v_3\}$ using the induction hypothesis. Suppose that in the algorithm for T' the source sends in the first step to some r_{i_1} and at the second step to r_{i_2} (with $r_{i_1} \neq r_{i_2}$ by induction hypothesis). If $i_1 \in \{1, 2, 3\}$, then in the algorithm for T the source sends first the message of v_{i_1} to r_{i_1}, and if $i_2 \in \{1, 2, 3\}$, then in the second step of the algorithm for T, the source sends the message of v_{i_2} to r_{i_2}. Otherwise the source sends in the first three steps in any order. Doing so the calls needed to reach $v_i, i = 1, 2, 3$, will not interfere each other, neither with the calls used in the algorithm for T'. Therefore, as the algorithm for T' is valid by induction hypothesis, the algorithm for T is valid; furthermore it also satisfies the induction hypothesis as the source sends in the first two steps to two different subtrees.

Theorem 5. *Suppose T consists of at least 3 subtrees and $N - 1 > M_1$. Then $g(T) = N - 1$.*

Proof. The number of steps of the algorithm for T is that for T' plus 3 and so $g(T) = g(T') + 3$. Therefore, it suffices to prove that $g(T') = N - 4$. That will follow from the following claims which prove that for, $3 \leq i \leq m$, $M'_i \leq N - 5$ and for $i = 1$ or 2, $M'_i \leq N - 4$ and at most one of M'_1 and M'_2 is $N - 4$. Therefore, by induction hypothesis the maximum is $N - 4$.

Claim 1 : If $m \geq 4$, $M_4 \leq N - 5$ and so $M'_i \leq N - 5$ for $i \geq 4$.

Proof. Suppose that $M_4 \geq N-4$. By the property P, $n_i \geq \frac{M_i+3}{3}$. Hence, we have $N - 1 = n_1 + n_2 + n_3 + n_4 + \ldots n_m \geq \frac{M_1+M_2+M_3+M_4}{3} + 4 \geq \frac{4M_4}{3} + 4 \geq N + \frac{M_4}{3}$ which is a contradiction. So $M_4 \leq N - 5$.

Now, if for $i = 1, 2, 3$ some T_i is reduced to one vertex, then $M'_i = 0$. Otherwise $M'_i \leq M_i - 2 \leq N - 4$ by hypothesis.

Claim 2 : $M_3 \leq N - 3$ and so $M'_3 \leq N - 5$.

Proof. Suppose that $M_3 \geq N - 2$. By the property P, $N - 1 \geq n_1 + n_2 + n_3 \geq \frac{M_1+M_2+M_3}{3} + 3 \geq N + 1$ a contradiction.

Finally if both $M'_1 = M'_2 = N - 4$, this implies that for $i = 1, 2$ $M_i = N - 2$ and $M'_i = M_i - 2$ and so T_1 and T_2 are both of type 1 and therefore $2n_i - 1 = M_i = N-2$ and $n_1+n_2 = N-1$. But as $m \geq 3$, $N-1 > n_1+n_2$ a contradiction.

Theorem 6. *Suppose T consists of at least 3 subtrees and $M_1 \geq N-1$ and T_1 is of type 2. Then $g(T) = M_1 + \epsilon$.*

Proof. Here again $g(T) = g(T') + 3$. As T_1 is of type 2, we have $M'_1 = M_1 - 3$. We have also $M'_2 = M_2 - 2$ or $M_2 - 3$, $M'_3 = M_3 - 2$ or $M_3 - 3$ and $M'_i = M_i$ for $i \geq 4$.

Claim : $M_4 \leq M_1 - 5$ and so $M'_4 < M'_1$.

Proof. Indeed otherwise we have $M_2 \geq M_3 \geq M_4 \geq M_1 - 4$ and therefore $N - 1 = n_1 + n_2 + n_3 + n_4 + \dots n_m \geq \frac{M_1+M_2+M_3+M_4}{3} + 4 \geq \frac{4M_1}{3}$ a contradiction as $M_1 \geq N-1$.

To complete the proof of the theorem we need to distinguish various cases:

Case 1: $M_2 \leq M_1 - 2$. Then $M'_2 < M'_1$ and also $M'_3 < M'_1$. So M'_1 is the maximum only attained for T'_1 and so, as $M'_1 \geq N' - 1$, $g(T') = M'_1$ and $g(T) = g(T') + 3 = M'_1 + 3 = M_1$.

Case 2: $M_2 \geq M_1 - 1$. In that case we claim that $M_3 \leq M_1 - 4$. Indeed otherwise we have $M_3 \geq M_1 - 3$ and therefore $N-1 = n_1+n_2+n_3+n_4+\dots n_m \geq \frac{M_1+3+M_2+3+M_3+3}{3} > M_1$ a contradiction as $M_1 \geq N-1$.

Sub-case 2.a : $M_2 = M_1$ (in that case, $\epsilon = 1$) and T_2 is of type 1. Here $M'_2 = M_2 - 2 = M'_1 + 1$ and then $g(T') = M'_2$. Then $g(T) = g(T') + 3 = M'_2 + 3 = M_2 + 1 = M_1 + \epsilon$.

Sub-case 2.b : $M_2 = M_1$ (in that case, $\epsilon = 1$) and T_2 is of type 2. Here $M'_2 = M_2 - 3 = M'_1$ and then $g(T') = M'_1 + \epsilon$. Then $g(T) = g(T') + 3 = M'_1 + \epsilon + 3 = M_1 + \epsilon$.

Sub-case 2.c : $M_2 = M_1 - 1$ and T_2 is of type 2, then $M'_2 = M_2 - 3 = M'_1 - 1$ and $g(T') = M'_1$ and $g(T) = g(T') + 3 = M_1$.

Sub-case 2.d : $M_2 = M_1 - 1$ and T_2 is of type 1. This case was dealt in the preceding subsection 4.2.

Algorithm for case C ($M_1 \geq N-1$ and T_1 is of type 1)

We use similar idea as for cases A and B, but deleting 4 vertices here: two in T_1, one in T_2 and one in T_3.

We apply the basic algorithm A to T_1. Let x_1 and x_2 be the two vertices for which messages have been sent at steps 1 and 3 respectively in the algorithm A. Choose in T_2 (resp T_3) a leaf v_2 (resp v_3) in the largest component $T_{2,j}$ (resp $T_{3,j'}$). The source will send the message destinated to v_2 (resp v_3) to r_2 (resp r_3) in an order to be precised later.

Let T' be the tree obtained by deleting x_1, x_2, v_2, v_3.

We apply the induction to the tree T'. We will see that M'_1 is the maximum of the M'_i's in T'. Furthermore by the choice of x_1 we have $n'_{1,1} = n_{1,1} - 1$ and $n'_1 = n_1 - 2$ and so as $n_1 \geq 2n_{1,1}$ (T_1 being of type 1), $n'_1 \geq 2n'_{1,1}$ and therefore T'_1 is of type 1. Hence by induction hypothesis for T', the source uses Algorithm A for T'_1 and therefore sends at the first step to r_1 a message destinated to the vertex x_3 of the algorithm A applied to T_1 (note that x_3 is in the biggest component of T_1').

Suppose that in the algorithm for T' the source sends at step 2 to some r_{i_1}. Now, in the algorithm for T, at step 2 the source will send to $r_{2+\epsilon}$ a message for $v_{2+\epsilon}$, where $\epsilon = 1$ if $i_1 = 3$ and 0 otherwise (said otherwise we send to T_2 except if in the algorithm for T' in its second step we sent to T_3). Then the message is immediately forwarded in the next few steps to $v_{2+\epsilon}$ along the path from s to $v_{2+\epsilon}$. At step 4 s sends to $r_{3-\epsilon}$ a message for $v_{3-\epsilon}$ which is immediately forwarded in the next few steps to $v_{3-\epsilon}$ along the path from s to $v_{3-\epsilon}$.

All these calls do not interfere, neither with those done in T_1, as the source sends only in step 2 and 4, nor with those of the algorithm for T', as at the end of step 4 the message to $v_{2+\epsilon}$ is already at distance 3 from s. For the message sent to $v_{3-\epsilon}$ note that by the choice of ϵ the source sends in T' to $r_{3-\epsilon}$ only at a step ≥ 3, so at least 3 steps after sending in T. Therefore, as the algorithm for T' is valid by induction hypothesis, the algorithm for T is valid and satisfies the induction hypothesis (Algorithm A is used for T_1).

Theorem 7. *Suppose T consists of at least 3 subtrees and $M_1 \geq N-1$ and T_1 is of type 1. Then $g(T) = M_1$.*

Proof. We have $g(T) = g(T')+4$, $|T'| = |T|-4$ and $M_1' = M_1-4$. So $M_1' \geq N'-1$ and therefore it suffices to prove that M_1' is the maximum in T' for completing the proof. First let us prove that $M_3 \leq M_1 - 5$ and so $M_i' \leq M_1' - 1$ for $i \geq 3$. Indeed recall that $M_1 = 2n_1 - 1$ or $n_1 = \frac{M_1+1}{2}$; Let $M_3 = M_1 - \alpha$, then $N-1 = n_1 + n_2 + n_3 + n_4 + \ldots + n_m \geq \frac{M_1+1}{2} + \frac{M_2+3}{3} + \frac{M_3+3}{3} \geq \frac{7M_1-4\alpha}{6} + \frac{5}{2}$. So as $M_1 \geq N-1$, $4\alpha \geq N+14$ and as $N > 2$, $\alpha > 4$.

To finish the proof we consider the following 3 cases.

Case 1 : $M_2 \leq M_1 - 3$. Then $M_2' \leq M_2 - 2 \leq M_1 - 5 = M_1' - 1$.

Case 2 : $M_2 \geq M_1 - 2$ and T_2 is of type 1.

We show that this can not happen. Indeed otherwise $M_2 = 2n_2 - 1$ or $n_2 = \frac{M_2+1}{2}$ and then $N-1 = n_1 + n_2 + n_3 + n_4 + \ldots + n_m \geq \frac{M_1+1}{2} + \frac{M_2+1}{2} + n_3 + n_4 + \ldots + n_m \geq N-1 + n_3 + n_4 + \ldots + n_m$ which implies $n_i = 0$ for $i \geq 3$ and so it contradicts the hypothesis that we have $m \geq 3$ subtrees.

Case 3 : $M_2 \geq M_1 - 2$ and T_2 is of type 2.

Sub-case 3.1. $M_2 = M_1 - 2$. Then, as T_2 is of type 2, $M_2' = M_2 - 3 \leq M_1 - 5 = M_1' - 1$.

Sub-case 3.2. $M_2 = M_1 - 1$. This case was done as a special case in the preceding subsection.

Sub-case 3.3. $M_2 = M_1$. This is not possible by the choice of ordering the subtree as if $M_2 = M_1$, the subtree of type 2 appears before that of type 1.

4 Conclusion

In this paper, we present efficient algorithms that give optimal solution for the gathering problem with buffering possibility, when the network is a tree with

$d_I = 1$. It should be noted that in our algorithms, the size of our buffers never exceeds 1. However with such a small buffer, we can in some cases decrease considerably the gathering time comparing to the non buffering assumption considered in [3]. An extension would be to consider a non uniform distribution of messages. Our algorithm can be easily extended to the case where a node receives or sends $w(u) > 0$ messages ; indeed it suffices to replace a vertex with $w(u)$ messages by $w(u)$ vertices with one message. However if $w(u)$ is allowed to be 0, then the problem will become much more complicated.

It would also be interesting to investigate this problem for different value of d_I or some other structures of networks. In particular it is still an open question to decide if the problem is polynomial for trees in general.

5 Acknowledgments

We would like to thank all the persons who help us with fruitful discussions in particular L. Gargano, A. Liestman, J. Peters and S. Perennes.

References

1. J.-C. Bermond, R. Correa, and M.-L. Yu. Optimal gathering protocols on paths under interference constraints. *Discrete Mathematics*, 2008, to appear.
2. J-C. Bermond, J. Galtier, R. Klasing, N. Morales, and S. Pérennes. Hardness and approximation of gathering in static radio networks. *Parallel Processing Letters*, 16(2):165–183, 2006.
3. J-C. Bermond, L. Gargano, and A.A. Rescigno. Gathering with minimum delay in tree sensor networks. In *SIROCCO 2008*, volume 5058 of *Lecture Notes in Computer Science*, pages 262–276. Springer-Verlag, June 2008.
4. J-C. Bermond, L. Gargano, A.A. Rescigno, and U. Vaccaro. Fast gossiping by short messages. *SIAM Journal on Computing*, 27(4):917–941, 1998.
5. J-C. Bermond and J. Peters. Efficient gathering in radio grids with interference. In *AlgoTel'05*, pages 103–106, Presqu'île de Giens, May 2005.
6. P. Bertin, J-F. Bresse, and 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, and L. Stougie. An approximation algorithm for the wireless gathering problem. *Operations Research Letters*, 36(5):605–608, 2008.
8. M. Christersson, L. Gasieniec, and A. Lingas. Gossiping with bounded size messages in ad-hoc radio networks. In *Proc. of ICALP'02*, volume 2380 of *Lecture Notes in Computer Science*, pages 377–389. Springer-Verlag, 2002.
9. M. Chrobak, L. Gasieniec, and W. Rytter. Fast broadcasting and gossiping in radio networks. *Journal of Algorithms*, 43(2):177–189, 2002.
10. M. L. Elkin and G. Kortsarz. Logarithmic inapproximability of the radio broadcast problem. *Journal of Algorithms*, 52(1):8–25, 2004.
11. C. Florens, M. Franceschetti, and R. McEliece. Lower bounds on data collection time in sensory networks. *IEEE Journal on Selected Areas in Communications*, 22(6):1110–1120, 2004.

12. I. Gaber and Y. Mansour. Centralized broadcast in multihop radio networks. *Journal of Algorithms*, 46(1):1–20, 2003.
13. L. Gargano and A. Rescigno. Optimally fast data gathering in sensor networks. In R. Královič and P. Urzyczyn, editors, *Proc. of MCFS'06*, volume 4162 of *Lecture Notes in Computer Science*, pages 399–411. Springer-Verlag, 2006.
14. L. Gasieniec and I. Potapov. Gossiping with unit messages in known radio networks. In *Proceedings of the IFIP 17th World Computer Congress*, pages 193–205. Kluwer, B.V., 2002.
15. J. Hromkovic, R. Klasing, A. Pelc, P. Ruzicka, and W. Unger. *Dissemination of Information in Commmunication Networks: Part I. Broadcasting, Gossiping, Leader Election, and Fault-Tolerance.* Springer Monograph. Springer-Verlag, 2004.
16. R. Klasing, N. Morales, and S. Pérennes. Complexity of bandwidth allocation in radio networks: the static case. To appear in Theoretical Computer Science.