Finding the Size of a Radio Network with Short Labels

Barun Gorain* Andrzej Pelc†

May 1, 2017

Abstract

The number of nodes of a network, called its size, is one of the most important network parameters. Knowing the size (or a good upper bound on it) is a prerequisite of many distributed network algorithms, ranging from broadcasting and gossiping, through leader election, to rendezvous and exploration. A radio network is a collection of stations, called nodes, with wireless transmission and receiving capabilities. It is modeled as a simple connected undirected graph whose nodes communicate in synchronous rounds. In each round, a node can either transmit a message to all its neighbors, or stay silent and listen. At the receiving end, a node $v$ hears a message from a neighbor $w$ in a given round, if $v$ listens in this round, and if $w$ is its only neighbor that transmits in this round. If $v$ listens in a round, and two or more neighbors of $v$ transmit in this round, a collision occurs at $v$. Two scenarios are considered in the literature: if nodes can distinguish collision from silence (the latter occurs when no neighbor transmits), we say that the network has the collision detection capability, otherwise there is no collision detection.

We consider the task of size discovery: finding the size of an unknown radio network with collision detection. All nodes have to output the size of the network, using a deterministic algorithm. Nodes have labels which are (not necessarily distinct) binary strings. The length of a labeling scheme is the largest length of a label.

We concentrate on the following problem:

What is the shortest labeling scheme that permits size discovery in all radio networks of maximum degree $\Delta$?

Our main result states that the minimum length of such a labeling scheme is $\Theta(\log \log \Delta)$. The upper bound is proven by designing a size discovery algorithm using a labeling scheme of length $O(\log \log \Delta)$, for all networks of maximum degree $\Delta$. The matching lower bound is proven by constructing a class of graphs (in fact even of trees) of maximum degree $\Delta$, for which any size discovery algorithm must use a labeling scheme of length at least $\Omega(\log \log \Delta)$ on some graph of this class.

keywords: radio network, collision detection, network size, labeling scheme

*Département d’informatique, Université du Québec en Outaouais, Gatineau, Québec J8X 3X7, Canada. baruniitg123@gmail.com
†Département d’informatique, Université du Québec en Outaouais, Gatineau, Québec J8X 3X7, Canada. pelc@uqo.ca. Partially supported by NSERC discovery grant 8136–2013 and by the Research Chair in Distributed Computing at the Université du Québec en Outaouais.
1 Introduction

1.1 The model and the problem

The number of nodes of a network, called its size, is one of the most important network parameters. The knowledge of the size (or of a good upper bound on it) by nodes of a network or by mobile agents operating in it, is a prerequisite of many distributed network algorithms, ranging from broadcasting and gossiping, through leader election, to rendezvous and exploration.

A radio network is a collection of stations, called nodes, with wireless transmission and receiving capabilities. It is modeled as a simple connected undirected graph. As it is usually assumed in the algorithmic theory of radio networks \cite{3,14,15}, all nodes start simultaneously and communicate in synchronous rounds. In each round, a node can either transmit a message to all its neighbors, or stay silent and listen. At the receiving end, a node \( v \) hears a message from a neighbor \( w \) in a given round, if \( v \) listens in this round, and if \( w \) is its only neighbor that transmits in this round. If \( v \) listens in a round, and two or more neighbors of \( v \) transmit in this round, a collision occurs at \( v \). Two scenarios are considered in the literature: if nodes can distinguish collision from silence (the latter occurs when no neighbor transmits), we say that the network has the collision detection capability, otherwise there is no collision detection.

We consider the task of size discovery: finding the size of an unknown radio network with collision detection. All nodes have to output the size of the network, using a deterministic algorithm. Nodes have labels which are (not necessarily distinct) binary strings. These labels are given to (otherwise anonymous) nodes by an oracle knowing the network, whose aim is to help the nodes in executing a size discovery algorithm using these labels. Such informative labeling schemes, also referred to as advice given to nodes, have been previously studied, e.g., in the context of ancestor queries \cite{1}, MST computation \cite{11}, and topology recognition \cite{13}, for wired networks. The length of a labeling scheme is the largest length of a label. A priori, every node knows only its own label.

In this paper we concentrate on the problem of finding a shortest labeling scheme permitting size discovery in radio networks with collision detection. Clearly, some labels have to be given to nodes, because otherwise (in anonymous radio networks) no communication is possible. On the other hand, labeling schemes of length \( \Theta(\log n) \), for \( n \)-node networks, are certainly enough, as the size of the network can be then coded in the labels. Our aim is to answer the following question.

What is the shortest labeling scheme that permits size discovery in all radio networks of maximum degree \( \Delta \)?

1.2 Our results

Our main result states that the minimum length of a labeling scheme that permits size discovery in all radio networks of maximum degree \( \Delta \), is \( \Theta(\log \log \Delta) \). The upper bound is proven by designing a size discovery algorithm using a labeling scheme of length \( O(\log \log \Delta) \), for all networks of maximum degree \( \Delta \). The matching lower bound is proven by constructing a class of graphs (in fact even of trees) of maximum degree \( \Delta \), for which any size discovery algorithm must use a labeling scheme of length at least \( \Omega(\log \log \Delta) \) on some graph of this class.
1.3 Related work

Algorithmic problems in radio networks modeled as graphs were studied for such distributed tasks as broadcasting [3, 15], gossiping [3, 14] and leader election [5, 23]. In some cases [3, 14], the model without collision detection was used, in others [17, 23], the collision detection capability was assumed.

Providing nodes of a network, or mobile agents circulating in it, with information of arbitrary type (in the form of binary strings) that can be used by an algorithm to perform some network task, has been proposed in [1, 4, 6–12, 16, 20–22, 24]. This approach was referred to as algorithms using informative labeling schemes, or equivalently, algorithms with advice. When advice is given to nodes, two variations are considered: either the binary string given to nodes is the same for all of them [18] or different strings may be given to different nodes [11, 13], as in our present case. If strings may be different, they can be considered as labels assigned to (otherwise anonymous) nodes. Several authors studied the minimum length of labels required for a given network problem to be solvable, or to solve a network problem in an efficient way. The framework of advice or of labeling schemes permits us to quantify the amount of needed information, regardless of the type of information that is provided and of the way the algorithm subsequently uses it.

In [9], the authors compared the minimum size of advice required to solve two information dissemination problems, using a linear number of messages. In [22], given a distributed representation of a solution for a problem, the authors investigated the number of bits of communication needed to verify the legality of the represented solution. In [10], the authors established the size of advice needed to break competitive ratio 2 of an exploration algorithm in trees. In [11], it was shown that advice of constant size permits to carry out the distributed construction of a minimum spanning tree in logarithmic time. In [14], short labeling schemes were constructed with the aim to answer queries about the distance between any pair of nodes. In [7], the advice paradigm was used for online problems. In the case of [24], the issue was not efficiency but feasibility: it was shown that \( \Theta(n \log n) \) is the minimum size of advice required to perform monotone connected graph clearing.

There are two papers studying the size of advice in the context of radio networks. In [20], the authors studied radio networks for which it is possible to perform centralized broadcasting in constant time. They proved that \( O(n) \) bits of advice allow to obtain constant time in such networks, while \( o(n) \) bits are not enough. In [19], the authors considered the problem of topology recognition in wireless trees without collision detection. Similarly to the present paper, they investigated short labeling schemes permitting to accomplish this task. It should be noted that the results in [19] and in the present paper are not comparable: [19] studies a harder task (topology recognition) in a weaker model (no collision detection), but restricts attention only to trees, while the present paper studies an easier task (size discovery) in a stronger model (with collision detection) but our results hold for arbitrary networks.

2 Preliminaries

According to the definition of labeling schemes, a label of any node should be a finite binary string. For ease of comprehension, in our positive result, we present our labels in a more structured way, namely as sequences \((a, b, c, d)\), where \(a\) is a binary string of length 7, and each of \(b, c\) and \(d\) is a pair whose first term is a binary string, and the second term is a bit. Each of the components \(a, b, c, d,\) is later used in the size discovery algorithm in a particular way. It is well known that such a sequence \((a, b, c, d)\) can be unambiguously coded as a single binary string whose length is
a constant multiple of the sum of lengths of all binary strings that compose it. Hence, presenting labels in this more structured way and skipping the details of the encoding does not change the order of magnitude of the length of the constructed labeling schemes.

In our algorithm, we use the subroutine \( \text{Wave}(x) \), for a positive integer \( x \), that can be implemented in radio networks with collision detection (cf. \([2]\) where a similar procedure was called Algorithm \( \text{Encoded-Broadcast} \)). We describe the subroutine below, for the sake of completeness. The aim of \( \text{Wave}(x) \) is to transmit the integer \( x \) to all nodes of the network, bit by bit. Let \( p = (a_1a_2\ldots a_k) \) be the binary representation of the integer \( x \). Consider the binary sequence \( p^* = (b_1, b_2, \ldots, b_{2k+2}) \) of length \( 2k + 2 \) that is formed from \( p \) by replacing every bit 1 by 10, every bit 0 by 00, and adding 11 at the end. For example, if \( p = (1101) \) then \( p^* = (1010001011) \). A node \( v \) initiates \( \text{Wave}(x) \) in some round \( r + 1 \) as follows: in consecutive rounds \( r + 1, \ldots, r + 2k + 2 \), node \( v \) transmits some message \( m \) in round \( r + i \), if \( b_i = 1 \), and remains silent if \( b_i = 0 \). Node \( v \) is marked and every node other than \( v \) is unmarked. An unmarked node \( w \) listens until a round \( t + 1 \) when it hears either a collision or a message, and then until two consecutive rounds occur when it hears either a collision or a message. Suppose that the second of these two rounds is round \( s \). Then \( w \) decodes \( p^* \) by putting \( b_i = 1 \) if it heard the message in round \( t + i \), and putting \( b_i = 0 \) if it heard silence in round \( t + i \). From \( p^* \) it computes unambiguously \( p \) and then \( x \). Subsequently, in consecutive rounds \( s + 1, \ldots, s + 2k + 2 \), node \( w \) transmits message \( m \) in round \( s + i \), if \( b_i = 1 \), and remains silent if \( b_i = 0 \). In round \( s + 2k + 2 \) node \( w \) becomes marked.

In this way, the subroutine \( \text{Wave}(x) \) proceeds from level to level, where the \( i \)-th level is formed by the nodes at distance \( i \) from \( v \) in the graph. Every level is involved in the subroutine during \( 2k + 2 \) rounds. Since a sequence of the form \( p^* \) cannot be a prefix of another sequence of the form \( q^* \), every node can determine when the transmissions from the previous level are finished, and can correctly decode \( x \). In our applications, no other transmissions are performed simultaneously with transmissions prescribed by \( \text{Wave}(x) \), and hence nodes can compute when a given \( \text{Wave} \) will terminate.

## 3 The Algorithm

In this section, we construct a labeling scheme of length \( O(\log \log \Delta) \) and a size discovery algorithm using this scheme and working for any radio network of maximum degree \( \Delta \).

Let \( G \) be a graph of maximum degree \( \Delta \). Let \( r \) be any node of \( G \) of degree \( \Delta \). For \( l \geq 0 \), a node is said to be in level \( l \), if its distance from \( r \) is \( l \). Let \( h \) be the maximum level in \( G \). Let \( V(l) \) be the set of nodes in level \( l \). For any node \( v \in V(l) \), let \( N(v) \) be the set of neighbors of \( v \) which are in level \( l + 1 \). Let \( U = \{v_1, v_2, \ldots, v_k\} \) be an ordered set of nodes in level \( l \). For all \( v \in V(l) \setminus U \), we define \( N'(v) = N(v) \setminus (\cup_{w \in U} N(w)) \) and for \( v_j \in U \), \( N'(v_j) = N(v_j) \setminus (\cup_{i=1}^{j-1} N(v_i)) \). An ordered subset \( U \) of \( V(l) \) is said to be an upper set at level \( l \), if for each \( v \in U \), \( N'(v) \neq \emptyset \) and \( \cup_{w \in U} N(w) = V(l + 1) \) (see Fig. 1). Below, we propose an algorithm that computes an upper set at each level \( l \), for \( 1 \leq l \leq h - 1 \).

The algorithm works in a recursive way. The first node \( v_1 \) of the set is chosen arbitrarily. At any step, let \( US(l) = \{v_1, \ldots, v_i\} \) be the set computed by the algorithm in the previous step. Let \( u_1^1, u_2^1, \ldots, u_{|N'(v_j)|}^1 \) be nodes in \( N'(v_j) \) for \( 1 \leq j \leq i \). Let \( k_j = 1 + \log[|N'(v_j)|] \). If \( V(l + 1) \setminus (\cup_{w \in US(l)} N(w)) \neq \emptyset \), then the next node in \( US(l) \) is added using the following rules.

1. Find the last node \( v_a \) in \( US(l) \) that has a common neighbor in \( \{u_1^a, u_2^a, \ldots, u_{k_a}^a\} \) with some
node \( v \in V(l) \setminus US(l) \) such that \( N(v) \setminus (\cup_{w \in US(l)} N(w)) \neq \emptyset \). Choose such a node \( v \) in \( US(l) \) that has the common neighbor \( u_b^a \) with \( v_a \), where \( b = \min\{1, 2, \cdots, k_a\} \). Add \( v \) to \( US(l) \) as the node \( v_{i+1} \).

2. If no such node in \( v_a \) exists in \( US(l) \), add any node \( v \in V(l) \setminus US(l) \) with \( N(v) \setminus (\cup_{w \in US(l)} N(w)) \neq \emptyset \) as the node \( v_{i+1} \).

The construction of \( US(l) \) is completed when \( \cup_{w \in US(l)} N(w) = V(l + 1) \).

Also, for every node \( v_i \in US(l) \), the nodes \( u^1_i, u^2_i, \cdots, u^k_i \) are assigned some unique id’s from the set \( \{1, 2, \cdots, |\log \Delta| + 1\} \). Moreover, if a node \( v_m \) is added to \( US(l) \) according to the first rule, where \( v_m \) has a common neighbor \( u^j_i \) with \( v_i \), then the node \( u^j_i \) gets the same id as \( u^j_i \). If a node \( v_m \) is added according to the 2nd rule, then \( u^j_i \) gets the id 1. These id’s will be later used to construct the labels of the nodes. In Algorithm 1 we give the pseudocode of the procedure that constructs an upper set \( US(l) \) for each level \( l \), and that assigns id’s to some nodes of \( V(l + 1) \), as explained above.

For \( 0 \leq l \leq h \), we define the weight \( W(v) \) of a node \( v \in V(l) \) as follows. If \( v \in V(h) \), we define \( W(v) = 1 \). For a node \( v \in V(l) \), where \( 0 \leq l \leq h - 1 \), we define \( W(v) = 1 + \sum_{u \in N(v)} W(u) \). Thus, for any level \( l \), the sum of the weights of nodes at level \( l \) is equal to the total number of nodes in levels \( l' \geq l \). Hence the weight of the node \( r \) is the size of the network.

We are now ready to define the labeling scheme \( \Lambda \) that will be used by our size discovery algorithm. The label \( \Lambda(v) \) of each node \( v \) contains two parts. The first part is a vector of markers that is a binary string of length 7, used to identify nodes with different properties. The second part is a vector of three tags. Each tag is a pair \((id, b)\), where \( id \) is the binary representation of an integer from the set \( \{1, 2, \cdots, |\log \Delta| + 1\} \), and \( b \) is either 0 or 1. Every node will use the tags to identify the time slot when it should transmit and what it should transmit in this particular time slot.

We first describe how the markers are assigned to different nodes of \( G \).

1. The node \( r \) gets the marker 0, and one of the nodes in level \( h \) gets the marker 1.
2. Choose any set of \( |\log \Delta| + 1 \) nodes in \( N(r) \) and give them the marker 2.
3. Let \( P \) be a simple path from \( r \) to the node with marker 1. All the internal nodes in \( P \) get the marker 3.
4. For each \( l, 0 \leq l \leq h - 1 \), all the nodes in \( US(l) \) get the marker 4. The last node of \( US(l) \) gets the marker 5 and a unique node from \( V(l + 1) \) with maximum weight in this set gets the marker 6.
Algorithm 1 \textit{ComputeSet}(l)

1: $US(l) \leftarrow \{\}$, $\text{count} \leftarrow 0$
2: $V'(l) \leftarrow V(l)$
3: for all $v \in V(l)$ do
4: \hspace{1em} $N'(v) \leftarrow N(v)$
5: \hspace{1em} end for
6: for all $v \in V'(l)$ do
7: \hspace{1em} \text{Compute}(v,1)
8: \hspace{1em} end for
9: Return $US(l)$

Algorithm 2 \textit{Compute}(v,j)

1: $\text{count} \leftarrow \text{count} + 1$
2: $v_{\text{count}} \leftarrow v$
3: $ID(v) \leftarrow \{j\}$
4: $US(l) \leftarrow US(l) \cup \{v_{\text{count}}\}$
5: for all nodes $v' \in V'(l) \setminus US(l)$ do
6: \hspace{1em} $N'(v') \leftarrow N'(v') \setminus (\cup_{w \in US(l)} N(w))$
7: \hspace{1em} if $N'(v') = \emptyset$ then
8: \hspace{1em} \hspace{1em} $V'(l) \leftarrow V'(l) \setminus \{v'\}$
9: \hspace{1em} end if
10: end for
11: if $V'(l) \neq \emptyset$ then
12: \hspace{1em} Let $N'(v) = \{u_1, u_2, \ldots, u_k\}$
13: \hspace{1em} for $i = 1$ to $\lceil \log k \rceil + 1$ do
14: \hspace{1em} \hspace{1em} if $i = 1$ then
15: \hspace{1em} \hspace{1em} \hspace{1em} $p \leftarrow j$
16: \hspace{1em} \hspace{1em} else
17: \hspace{1em} \hspace{1em} \hspace{1em} $p \leftarrow \min (\{1, 2, \ldots, \lceil \log \Delta \rceil + 1\} \setminus ID(v))$
18: \hspace{1em} \hspace{1em} end if
19: \hspace{1em} \hspace{1em} $id(u_i) \leftarrow p$
20: \hspace{1em} \hspace{1em} while there exists some $v'' \in V'(l)$ such that $u_i \in N(v'')$ do
21: \hspace{1em} \hspace{1em} \hspace{1em} \text{Compute}(v,p)
22: \hspace{1em} \hspace{1em} end while
23: \hspace{1em} $ID(v) \leftarrow ID(v) \cup \{p\}$
24: \hspace{1em} end for
25: end if
The first part of every label is a binary string $M$ of length 7, where the markers are stored. Note that a node can be marked by multiple markers. If the node is marked by the marker $i$, for $i = 0, \ldots, 6$, we have $M(i) = 1$; otherwise, $M(i) = 0$.

The second part of the label of each node $v$ is a vector $[L_1(v), L_2(v), L_3(v)]$ containing three tags, namely, the ∆-learning tag $L_1(v)$, the collision tag $L_2(v)$, and the weight-transmission tag $L_3(v)$. The assignment of the above tags is described below.

1. The ∆-learning tags will be used for learning the value of ∆ by the root $r$. The node $r$ and all the nodes with marker 2 get the ∆-learning tags as follows. The nodes with marker 2 are neighbors $w_1, w_2, \ldots, w_{\lfloor \log \Delta \rfloor + 1}$ of the node $r$. For each $i$, $1 \leq i \leq \lfloor \log \Delta \rfloor + 1$, node $w_i$ is assigned the tag $(B(i), b_i)$, where $B(i)$ is the binary representation of the integer $i$ and $b_i$ is the $i$-th bit of the binary representation of ∆. The node $r$ gets the tag $(B, 0)$, where $B$ is the binary representation of the integer $\lfloor \log \Delta \rfloor + 1$. All other nodes of $G$ get the ∆-learning tag $(0, 0)$.

2. The collision tags will be used to create collisions. For each $l$, $0 \leq l \leq h - 1$, each node in $V(l + 1)$ gets the collision tag as follows. Let $US(l) = \{v_1, v_2, \ldots, v_k\}$. For $1 \leq i \leq k$ and $1 \leq j \leq \lfloor \log |N'(v_i)| \rfloor + 1$, the node $w_j \in V(l + 1)$ gets the collision tag $id(w_j), b_m)$, where $m$ is the position of the integer $id(w_j)$ in the set $ID(v_i)$ in increasing order, and $b_m$ is the $m$-th bit of the binary representation of $|N'(v_i)|$. All other nodes $v \in V(l + 1)$ get the collision tag $(0, 0)$.

3. The weight-transmission tags will be used by nodes to transmit their weight to a unique node in the previous level. For each $l$, $0 \leq l \leq h - 1$, each node in $V(l + 1)$ gets the transmission tag as follows. Let $US(l) = \{v_1, v_2, \ldots, v_k\}$. For $1 \leq i \leq k$, let $Q_i(x) = \{u \in N'(v_i)|W(u) = x\}$. Choose any subset $\{w_1, w_2, \ldots, w_{\lfloor \log |Q_i(x)| \rfloor + 1}\}$ of $Q_i(x)$. For $1 \leq i \leq \lfloor \log |Q_i(x)| \rfloor + 1$, the node $w_i$ gets the weight-transmission tag $(B(i), b_i)$, where $B(i)$ is the binary representation of the integer $i$, and $b_i$ is the $i$-th bit of the binary representation of $|Q_i(x)|$. All other nodes $v \in V(l + 1)$ get the weight-transmission tag $(0, 0)$.

This completes the description of the labeling scheme $\Lambda$. The size discovery algorithm using this scheme consists of three procedures, namely Procedure Parameter Learning, Procedure Size Learning, and Procedure Final. The detailed descriptions of each of these procedures are given below.

Procedure Parameter Learning. The aim of this procedure is for every node in $G$ to learn three integers: $\Delta$, the number of the level to which the node belongs, and $h$. The procedure consists of two stages. In the first stage, that starts in round 1, every node with $M(2) = 1$ and $M(0) = 0$ (i.e., a neighbor of $r$ with marker 2) transmits its ∆-learning tag in round $i$, if the id in the first component of this tag is $i$. The node with $M(0) = 1$, i.e., the node $r$, collects all the tags until it received a message from a node which has the same id as the id of $r$ in the ∆-learning tag. After receiving this message, the node $r$ has learned all pairs $(B(1), b_1), \ldots, (B(m), b_m)$, where $m$ is the id of $r$ and $B(i)$ is the binary representation of the integer $i$, corresponding to the ∆-learning tag at the respective nodes. Then node $r$ computes the string $s = (b_1 b_2 \ldots b_m)$. This is the binary representation of $\Delta$.

In the second stage, after learning $\Delta$, the node $r$ initiates the subroutine Wave($\Delta$). Every node other than $r$ waits until it detects two consecutive non-silent rounds. This indicates the end of the wave at this node and happens $2m + 2$ rounds after the wave has been started by the nodes of
the previous level. The node computes \( s \), learns \( \Delta \), computes \( m = \lfloor \log \Delta \rfloor + 1 \), and sets its level number as \( j \), if the end of the wave at this node occurred in round \( m + j(2m + 2) \).

When the unique node with \( M(1) = 1 \) learns its level number (which is \( h \)), it transmits the value of \( h \) in the next round. After receiving the first message containing an integer, a node with \( M(3) = 1 \) sets \( h \) to this integer and retransmits it. When the node with \( M(0) = 1 \), i.e., the node \( r \), gets the first message after round \( m \) that contains an integer, it learns \( h \) and initiates \( \text{Wave}(h) \).

The stage and the entire procedure end in round \( t_1 = m + h(2m + 2) + h + h(2(\lfloor \log h \rfloor + 1) + 2) \). Note that after learning \( h \), every node can compute \( t_1 \) and thus knows when Procedure \textbf{Parameter Learning} ends.

Procedure \textbf{Size Learning}. This is the crucial procedure of the algorithm. Its aim is to learn the size of the graph by the node \( r \), i.e., to learn its weight \( W(r) \). This procedure consists of \( h \) phases. In the \( i \)-th phase, where \( 1 \leq i \leq h \), the participating nodes are from level \( h - i + 1 \) and from level \( h - i \). We will show by induction on \( i \) that at the end of the \( i \)-th phase, all nodes of level \( h - i \) correctly compute their weights. Thus at the end of the \( h \)-th phase, the node \( r \) will learn its weight, i.e., the size of the network. The high-level idea of the \( i \)-th phase is the following.

In order to learn its weight, a node \( v \) in \( US(h - i) \) must learn the weights of all nodes \( u \) in \( N'(v) \) and subsequently add all these weights. Weight-transmission tags are used to achieve this. The difficulty consists in preventing other neighbors in level \( h - i \) of such nodes \( u \) from adding these weights when computing their own weight, as this would result in multiple accounting (see Fig. 1). This is done using collision tags to create collisions in other such nodes, so that nodes \( z \) in \( US(h - i) \) can identify neighbors in level \( h - i + 1 \) outside of \( N'(z) \) and ignore their weights. A node transmits its weight-transmission tag in a round which is an increasing function of its weight. Since the nodes in \( US(h - i) \) do not have any knowledge about their degree, they must learn the maximum possible weight of a node in level \( h - i + 1 \), to determine how long they must wait before receiving the last message from such a node.

We now give a detailed description of the \( i \)-th phase. At the beginning of the first phase, all nodes in level \( h \) set their weight to 1. The \( i \)-th phase starts in round \( t_2(i) + 1 \), where \( t_2(1) = t_1 \), and ends in round \( t_2(i + 1) \). We will show that \( t_2(i + 1) \) will be known by every node of the graph by the end of the \( i \)-th phase, i.e., by the round \( t_2(i + 1) \).

In round \( t_2(i) + 1 \) (which starts the \( i \)-th phase) the unique node \( u' \) of level \( h - i + 1 \) with \( M(6) = 1 \) (which is a node of this level with maximum weight), initiates \( \text{Wave}(W(u')) \). Every node in \( G \) learns the value \( x_i \), which is the maximal weight of a node in level \( h - i + 1 \), by round \( t_2(i) = t_2(i) + 2h(2(\lfloor \log x_i \rfloor + 1) + 1) \). Since every node knows \( h \) and \( t_2(i) \), and it learns \( x_i \) during the wave subroutine, it can compute the value \( t_2'(i) \) by which \( \text{Wave}(W(u')) \) is finished. After learning this integer, every node in level \( h - i + 1 \) and every node in level \( h - i \) maintains a variable \textit{status} which can be either \textit{complete} or \textit{incomplete}. (The variable \textit{status} is proper to a particular phase. In what follows we consider \textit{status} for phase \( i \).) Initially the status of every node in level \( h - i \) with \( M(4) = 1 \) (the nodes in \( US(h - i) \)) and of every node in level \( h - i + 1 \) is \textit{incomplete}. The initial status of the nodes in level \( h - i \) with \( M(4) = 0 \) is \textit{complete}.

At any time, only incomplete nodes will participate in this phase. The nodes with \( M(4) = 0 \), i.e., the nodes outside \( US(h - i) \), set their weights to 1 and never participate in this phase.

After learning its weight, a node \( v \) in \( US(h - i) \) gets status \textit{complete} and transmits a stop message in a special round. All the nodes in \( N'(v) \) learn this stop message either by receiving it or by detecting a collision in this special round, and become \textit{complete}. Thus the nodes in \( N'(v) \) never transmit in subsequent rounds, and this prevents multiple accounting of the weights.
Let $z$ be a node in level $h - i + 1$ with status incomplete.
If the id in the collision tag of $z$ is a positive integer $e$, then $z$ performs the following steps.

- The node $z$ transmits its collision tag for the first time in the $i$-th phase in round $t'_2(i) + e$.
  After that, the node $z$ transmits its collision tag in every round $t'_2(i) + e + j\tau_i$, where $\tau_i = \lfloor \log \Delta \rfloor + 1 + x_i(\lfloor \log \Delta \rfloor + 1) + 1$, and $j \geq 1$, until it gets a stop message or detects a collision in round $t'_2(i) + j'\tau_i$, for some integer $j' \geq 1$. In the latter case, node $z$ updates its status to complete.

If the id in the weight-transmission tag of $z$ is a positive integer $e'$, then $z$ performs the following steps.

- The node $z$ transmits the pair $(t, W(z))$, where $t$ is its weight-transmission tag and $W(z)$ is its weight, for the first time in the $i$-th phase in round $t'_2(i) + (W(z) - 1)(\lfloor \log \Delta \rfloor + 1) + e'$. After that the node $z$ transmits $(t, W(z))$ in every round $t'_2(i) + (W(z) - 1)(\lfloor \log \Delta \rfloor + 1) + e' + j\tau_i$, where $\tau_i = \lfloor \log \Delta \rfloor + 1 + x_i(\lfloor \log \Delta \rfloor + 1) + 1$, and $j \geq 1$, until it gets a stop message or detects a collision in the round $t'_2(i) + j'\tau_i$, for some integer $j' \geq 1$. In the latter case, it updates its status to complete.

Let $z'$ be a node with $M(4) = 1$, i.e., a node in $US(h - i)$. The node $z'$ (with status incomplete) performs the following steps.

- If $z'$ does not detect any collision in the time interval $[t'_2(i) + (j - 1)\tau_i, t'_2(i) + (j - 1)\tau_i + \lfloor \log \Delta \rfloor + 1]$, for some integer $j \geq 1$, then the node changes its status to complete. In this interval, the node $z'$ received the collision tags from the nodes in $N'(z')$. Suppose that the node $z'$ learns the pairs $(B(g_1), b_1), (B(g_2), b_2), \cdots, (B(g_k), b_k)$, where $B(g_1), B(g_2), \cdots, B(g_k)$ are the binary representations of the integers $g_1, g_2, \cdots, g_k$, respectively, in the increasing order, corresponding to the collision tags of the respective nodes. The node $z'$ computes $s' = (b_1b_2\cdots b_k)$. Let $d$ be the integer whose binary representation is $s'$. The integer $d$ is the size of $N'(z')$. Then $z'$ waits until round $t'_2(i) + j\tau_i$. By this time, all nodes in level $h - i + 1$ that transmitted according to their collision tags and weight-transmission tags, have already completed all these transmissions. If $z'$ detects any collision in the time interval $[t'_2(i) + (j - 1)\tau_i + \lfloor \log \Delta \rfloor + 2, t'_2(i) + j\tau_i + 1]$, it changes its status back to incomplete. Otherwise, for $1 \leq f \leq x_i$, let $(B(1), b_1), (B(2), b_2), \cdots, (B(f(a)), b_{g(f)}$ be the weight-transmission tags that the node $z'$ received from a node with weight $f$, where $B(a)$ is the binary representation of the integer $a$. Let $s_f = (b_1b_2\cdots b_{g(f)})$ and let $d_f$ be the integer whose binary representation is $s_f$. The integer $d_f$ is the total number of nodes of weight $f$ in $N'(z')$.

The node $z'$ computes the value $\sum_f d_f$. If the node $z'$ had received any message from a node which is not in $N'(z')$, then the sum $\sum_f d_f$ cannot be equal to the integer $d$, and hence the node learns that there is a danger of multiple accounting of weights. In that case, the node changes its status back to incomplete.

Otherwise, if $\sum_f d_f = d$, node $z'$ assigns $W(z') = 1 + \sum_f (f \cdot d_f)$. After computing $W(z')$, the node $z'$ transmits a stop message in round $t'_2(i) + j\tau_i$. If $z'$ is the node with $M(5) = 1$ (i.e., the last node of $US(h - i)$), then after sending the stop message, it initiates $Wave(T)$, where $T$ is the current round number. After learning $T$ from $Wave(T)$, every node in $G$ computes $t_2(i + 1) = T + 2h(2(\lfloor \log T \rfloor + 1) + 2)$. This is the round by which $Wave(T)$ is finished. In this round, the $i$-th phase of the procedure is finished as well.
At the end of the \( h \)-th phase, the node \( r \) learns its weight, sets \( n = W(r) \) and the procedure ends.

Procedure \textbf{Final}: After computing \( n \), the node \( r \) initiates \( \text{Wave}(n) \). Every node in \( G \) computes the value of \( n \), and outputs it. The procedure ends after all nodes output \( n \).

Now our algorithm can be succinctly formulated as follows:

\begin{algorithm}
\begin{algorithmic}
\State \textbf{Parameter Learning}
\State \textbf{Size Learning}
\State \textbf{Final}
\end{algorithmic}
\end{algorithm}

The proof of the correctness of Algorithm \textbf{Size Discovery} is split into two lemmas.

\textbf{Lemma 3.1} Upon completion of the Procedure \textbf{Parameter Learning}, every node in \( G \) correctly computes \( \Delta \), \( h \), and its level number. Moreover, every node computes the round number \( t_1 = m + h(2m + 2) + h(2(\lfloor \log h \rfloor + 1) + 2) \) by which the procedure is over.

\textbf{Proof}: After round \( m = \lfloor \log \Delta \rfloor + 1 \), the node \( r \) learns all pairs \( (B(1), b_1), \ldots, (B(m), b_m) \), where \( B(i) \) is the binary representation of the integer \( i \), corresponding to the \( \Delta \)-learning tags at the respective nodes with \( M(2) = 1 \). According to the assignment of the tags to the nodes, the binary string \( s = (b_1b_2\ldots b_m) \) is the binary representation of the integer \( \Delta \). Therefore, the node \( r \) correctly learns \( \Delta \).

After learning \( \Delta \), the node \( r \) initiate \( \text{Wave}(\Delta) \). For every level \( i \geq 1 \), the wave ends at the nodes in level \( i \) in round \( m + i(2m + 2) \). The nodes learn the value of \( \Delta \) from the wave and calculate their level number. The node in the \( h \)-th level for which \( M(1) = 1 \), learns \( h \) and transmits the value of \( h \) along the path with nodes for which \( M(3) = 1 \). The node \( r \) learns \( h \), and initiate \( \text{Wave}(h) \). Every node computes \( h \) from the wave. Knowing \( m \) and \( h \) every node computes \( t_1 \). \hfill \Box

\textbf{Lemma 3.2} At the end of the \( i \)-th phase of the Procedure \textbf{Size Learning}, every node in level \( h - i \) correctly computes its weight.

\textbf{Proof}: We prove this lemma in two steps. First, we prove the following two claims, and then we prove the lemma by induction using these claims.

\textbf{Claim 1}: In the \( i \)-th phase of Procedure \textbf{Size learning}, if the status of a node \( v_p \in US(h - i) \) is changed from \textit{incomplete} to \textit{complete} in the time interval \( [t_2(i) + (j - 1)\tau_i, t_2(i) + j\tau_i - 1] \), for some integer \( j \geq 1 \), and remains \textit{complete} forever, then the node \( v_p \) correctly computes \( W(v) \) in round \( t_2(i) + j\tau_i \), provided that all nodes of level \( h - i + 1 \) know their weight at the beginning of the \( i \)-th phase.

In order to prove this claim, suppose that the status of \( v_p \in US(h - i) \) is changed from \textit{incomplete} to \textit{complete} from \textit{incomplete} in the interval \( [t_2(i) + (j - 1)\tau_i, t_2(i) + j\tau_i - 1] \), for some integer \( j \geq 1 \). Since the status of \( v_p \) is \textit{complete} in round \( t_2(i) + j\tau_i \), the node \( v_p \) did not detect any collision in the above time interval. Suppose that \( v_p \) received messages only from nodes in \( N'(v_p) \). In round \( t_2(i) + (j - 1)\tau_i + \lfloor \log \Delta \rfloor + 1 \), the node computes the integer \( d \) from the collision tags of the nodes from which it received messages in the time interval \( [t_2(i) + (j - 1)\tau_i, t_2(i) + (j - 1)\tau_i + \lfloor \log \Delta \rfloor + 1] \). According to the labeling scheme, the bits in the collision tags of the nodes in \( N'(v_p) \) were assigned in such a way that the string \( s' \) formed by these bits is the binary representation of the integer \( |N'(v_p)| \). After that, the nodes in \( N'(v_p) \) whose weight-transmission tag contains a positive integer
as the id, transmit their tags to \( v_p \) one by one. Let \( X_f \subseteq N'(v_p) \) be the the set of nodes in \( N'(v_p) \) with weight \( f \), for \( 1 \leq f \leq x_i \). The weight-transmission tags are given to \( (\lceil \log |X_f| \rceil + 1) \) nodes in \( X_f \) in such a way that the binary string formed by the bits of the weight transmission tags of these nodes in the increasing order of their ids is the binary representation of the integer \( |X_f| \). Hence, for \( 1 \leq f \leq x_i \), \( \sum_f |X_f| = d \), as the sum of the numbers of nodes in \( N'(v_p) \) with different weights is equal to the total number of nodes in \( N'(v_p) \). If the node \( v_p \) received messages only from the nodes in \( N'(v_p) \), it learns \( \sum_f |X_f| = d \), and hence correctly computes \( W(v_p) = 1 + \sum_f f |X_f| \).

Otherwise, there exists a node \( u \in N(v_p) \cap N'(v_p) \), for some node \( v_q \in US(h - i) \) with \( q < p \), such that the id in the weight-transmission tag of \( u \) is non-zero. Then the integer \( \sum_f |X_f| \) that \( v_q \) computes cannot be equal to the integer \( d \), as explained above, and the node \( v_p \) changes its status back to \emph{incomplete}. This is a contradiction. Therefore, the node \( v_p \) correctly computes its weight at the end of round \( t_2'(i) + j \tau_i - 1 \), which proves the claim.

**Claim 2:** Let \( US(h - i) = \{v_1, v_2, \ldots, v_k\} \). In the \( i \)-th phase of Procedure \textbf{Size Learning}, each node \( v_j \) changes its status from \emph{incomplete} to \emph{complete} during the time interval \([t_2'(i) + (q_j - 1) \tau_i + 1, t_2'(i) + q_j \tau_i]\), for some \( q_j \leq j \), and remains \emph{complete} forever.

We prove this claim by induction on \( j \). As the base case, we prove that in the time interval \([t_2'(i) + 1, t_2'(i) + \tau_i - 1]\), the status of the node \( v_1 \in US(h - i) \) is changed from \emph{incomplete} to \emph{complete}. According to the labeling scheme and to the construction of the set \( US(h - i) \), \( (\lceil \log |N'(v_1)| \rceil + 1) \) nodes from \( N'(v_1) \) have distinct positive ids in their collision tags, and all other nodes from \( N'(v_1) \) have the id 0. Hence, the node \( v_1 \) detects no collision in the time interval \([t_2'(i) + 1, t_2'(i) + (\lceil \log \Delta \rceil + 1)]\), and it changes its status to \emph{complete}. In the next \( x_i(\lceil \log \Delta \rceil + 1) \) rounds, the nodes of level \( h - i + 1 \), with positive ids in their weight-transmission tags, transmit. Since the ids in the weight-transmission tags of \( (\lceil \log |N'(v_1)| \rceil + 1) \) nodes are distinct positive integers, and \( N(v_1) = N'(v_1) \), the node \( v_1 \) does not detect any collision. Also, since the node \( v_1 \) received messages only from nodes in \( N'(v_1) \), therefore \( \sum_f |X_f| = d \), for \( 1 \leq f \leq x_i \), and hence \( v_1 \) remains \emph{complete} forever.

Suppose by induction that Claim 2 holds for nodes \( v_1, \ldots, v_{j-1} \). Let \( y = \max\{q_1, q_2, \ldots, q_j\} \). Consider the following two cases.

Case 1: There exists an integer \( q_{j+1} \leq y \), such that the node \( v_{j+1} \) changes its status from \emph{incomplete} to \emph{complete} during the time interval \([t_2'(i) + (q_{j+1} - 1) \tau_i + 1, t_2'(i) + q_{j+1} \tau_i]\), for some \( q_{j+1} \leq y \), and remains \emph{complete} forever.

In this case the claim holds for \( v_{j+1} \) because \( q_{j+1} \leq y \leq j + 1 \).

Case 2: Case 1 does not hold.

Therefore, the status of \( v_{j+1} \) is \emph{incomplete} in round \( t_2'(i) + y \tau_i \). The status of all the nodes in \( N'(v_{j+1}) \) is \emph{incomplete} in this round as well, as they did not received any stop message from \( v_{j+1} \) or detected any collision in round \( t_2'(i) + y \tau_i \).

The status of the nodes in \( N(v_{j+1}) \setminus N'(v_{j+1}) \) is \emph{complete}, as \( N(v_{j+1}) \setminus N'(v_{j+1}) \subseteq \bigcup_{i=1}^j N(v_i) \) and the nodes \( v_1, v_2, \ldots, v_j \) are \emph{complete}. Consider the time interval \([t_2'(i) + y \tau_i + 1, t_2'(i) + (y + 1) \tau_i - 1]\). In this time interval, the node \( v_{j+1} \) receives messages only from the nodes in \( N'(v_{j+1}) \). Since the positive ids in the collision tags and the positive ids in the weight-transmission tags are unique for the nodes in \( N'(v_{j+1}) \), the node \( v_{j+1} \) does not detect any collision in the interval \([t_2'(i) + y \tau_i + 1, t_2'(i) + (y + 1) \tau_i - 1]\). Also, since the node \( v_{j+1} \) received messages only from nodes in \( N'(v_{j+1}) \), therefore \( \sum_j d_j = d \), and hence \( v_{j+1} \) remains \emph{complete} forever. Since \( y \leq j \), we have \( y + 1 \leq j + 1 \). Therefore, the proof of the claim follows by induction.

Now we prove the lemma by induction on the phase number. According to the definition of the
weight of a node, all the nodes in level $h$ have weight 1. Therefore, by Claim 2, at the end of round $t_2(1) + j_1$, the node $v_j$ in $US(h-1)$ becomes complete, and hence by Claim 1, it correctly computes its weight, since all the nodes in level $h$ already know their weight which is 1. This implies that all the nodes in level $h - 1$ correctly compute their weights at the end of phase 1. Suppose that for $i \geq 1$, all the nodes in level $h - i$ correctly compute their weights at the end of phase $i$. Then by Claim 2, all the nodes in $US(h - i - 1)$ become complete in the $(i + 1)$-th phase, and hence by Claim 1 they correctly compute their weights in this phase. Therefore, the lemma follows by induction.

Applying Lemma 3.2 for $i = h$, we get the following corollary.

**Corollary 3.1** Upon completion of Procedure Size Learning, the node $r$ correctly computes the size of the graph.

Now we are ready to formulate our main positive result.

**Theorem 3.1** The length of the labeling scheme used by Algorithm Size Discovery on a graph of maximum degree $\Delta$ is $O(\log \log \Delta)$. Upon completion of this algorithm, all nodes correctly output the size of the graph.

**Proof:** According to the labeling scheme $\Lambda$, the label of every node has two parts. The first part is a vector $M$ of constant length and each term of $M$ is one bit. The second part is a vector $L$ containing three tags, each of which is of length $O(\log \log \Delta)$. Therefore, the length of the labeling scheme $\Lambda$ is $O(\log \log \Delta)$.

By Corollary 3.1 node $r$ correctly computes the size $n$ of the network upon completion of Procedure Size Learning. In Procedure Final, node $r$ initiates Wave($n$), and hence every node correctly computes $n$ upon completion of the algorithm. \qed

4 The lower bound

In this section, we show that the length of the labeling scheme used by Algorithm Size Discovery is optimal, up to multiplicative constants. We prove the matching lower bound by showing that for some class of graphs of maximum degree $\Delta$ (indeed of trees), any size discovery algorithm must use a labeling scheme of length at least $\Omega(\log \log \Delta)$ on some graph of this class.

Let $S$ be a star with the central node $r$ of degree $\Delta$. Denote one of the leaves of $S$ by $a$. For $\left\lfloor \frac{\Delta}{2} \right\rfloor \leq i \leq \Delta - 1$, we construct a tree $T_i$ by attaching $i$ leaves to $a$. The maximum degree of each tree $T_i$ is $\Delta$. Let $\mathcal{T}$ be the set of trees $T_i$, for $\left\lfloor \frac{\Delta}{2} \right\rfloor \leq i \leq \Delta - 1$, cf. Fig. 2. Hence the size of $\mathcal{T}$ is at least $\Delta^2$.

The class $\mathcal{T}$ of trees was used in [19] to prove an analogous lower bound for the problem of topology recognition (which, for the class $\mathcal{T}$, is equivalent to size discovery). However, it should be stressed that the proof of the lower bound in our present scenario is much more involved because we work under the more powerful model assuming the capability of collision detection, while [19] assumed no collision detection. The negative result under our more powerful model is more difficult to obtain because of potential possibility of acquiring information by nodes from hearing collisions.

Let $R$ be the set of leaves attached to $r$ and let $A$ be the set of leaves attached to $a$. For a tree $T \in \mathcal{T}$, consider a labeling scheme $L(T)$ of length $\beta$, and let $A$ be an algorithm that finds the size of every tree $T \in \mathcal{T}$, using $L(T)$. Let $L(T)$ assign the label $l(v)$ to each node $v$ in $T$. 

12
Let $T \in \mathcal{T}$ be any tree. We define the notion of history (a similar notion was defined in [25] for anonymous radio networks without collision detection) for each node $v$ in $T$ in round $t$. The history of a node in time $t$ is denoted by $H(v,t,L,A)$. This is the information that node $v$ acquires by round $t$, using the algorithm $A$. The action of a node $v$ in round $t+1$ is a function of the history $H(v,t,L,A)$, hence for every round $t$, if two nodes have the same history in round $t$, then they behave identically in round $t+1$. As in [25], we assume without loss of generality, that whenever a node transmits a message in round $t+1$, it sends its entire history in round $t$. We define the history by induction on the round number as follows.

**Lemma 4.1** For any tree $T \in \mathcal{T}$ consider a labeling scheme $L(T)$. Let $A$ be any algorithm that finds the size of every tree $T \in \mathcal{T}$ using the scheme $L(T)$. Then for any $t \geq 0$, we have:

1. For $v_1,v_2 \in R$, $H(v_1,t,L,A) = H(v_2,t,L,A)$, if and only if $l(v_1) = l(v_2)$.
2. For $v_1,v_2 \in A$, $H(v_1,t,L,A) = H(v_2,t,L,A)$, if and only if $l(v_1) = l(v_2)$.

**Proof:** We prove the first part of the lemma. The proof of the second part is similar. By definition, for two nodes $v_1$ and $v_2$ with different labels, we have $H(v_1,t,L,A) \neq H(v_2,t,L,A)$ for all $t \geq 0$.

To prove the converse, we use induction on $t$. Let $v_1,v_2 \in R$ such that $l(v_1) = l(v_2)$. For $t = 0$, $H(v_1,0,L,A) = l(v_1) = l(v_2) = H(v_2,0,L,A)$. Suppose that the statement is true for round $t$,
Case 1 The node \(r\) transmits and the nodes \(v_1, v_2\) do not transmit.

According to the definition of history, \(H(v_1, t + 1, L, A) = [H(v_1, t, L, A), H(r, t, L, A)]\) and \(H(v_2, t + 1, L, A) = [H(v_2, t, L, A), H(r, t, L, A)]\). This implies \(H(v_1, t + 1, L, A) = H(v_2, t + 1, L, A)\).

Case 2 The nodes \(v_1\) and \(v_2\) transmit and the node \(r\) does not transmit.

According to the definition of history, \(H(v_1, t + 1, L, A) = [H(v_1, t, L, A), \lambda]\) and \(H(v_2, t + 1, L, A) = [H(v_2, t, L, A), \lambda]\). This implies \(H(v_1, t + 1, L, A) = H(v_2, t + 1, L, A)\).

Case 3 The nodes \(v_1\) and \(v_2\), and the node \(r\) transmit.

According to the definition of history, \(H(v_1, t + 1, L, A) = [H(v_1, t, L, A), \lambda]\) and \(H(v_2, t + 1, L, A) = [H(v_2, t, L, A), \lambda]\). This implies \(H(v_1, t + 1, L, A) = H(v_2, t + 1, L, A)\).

Case 4 Neither \(v_1\) nor \(v_2\) nor \(r\) transmit. In this case, \(H(v_1, t + 1, L, A) = [H(v_1, t, L, A), \lambda]\) and \(H(v_2, t + 1, L, A) = [H(v_2, t, L, A), \lambda]\). This implies \(H(v_1, t + 1, L, A) = H(v_2, t + 1, L, A)\).

Hence the proof of the lemma follows by induction. \(\square\)

With the length of the labeling scheme \(\beta\), there can be at most \(z = 2^{\beta+1}\) possible different labels of at most this length. Let \(\mathcal{L} = \{l_1, l_2, \ldots, l_z\}\) be the set of distinct labels of length at most \(\beta\). We define the pattern of a tree \(T\) with the labeling scheme \(L(T)\) as the pair \((P(r), P(a))\), where \(P(r)\) and \(P(a)\) are defined as follows.

\[P(r) = (l(r), b_1, b_2, \ldots, b_z), \text{ where } b_i \in \{0, 1, 2\} \text{ and:}\]
\[\begin{align*}
  b_i &= 0, \text{ if no node in } R \text{ has label } l_i; \\
  b_i &= 1, \text{ if there is exactly one node in } R \text{ with label } l_i; \\
  b_i &= 2, \text{ if there are more than one node in } R \text{ with label } l_i.
\end{align*}\]

\[P(a) = (l(a), b'_1, b'_2, \ldots, b'_z), \text{ where } b'_i \in \{0, 1, 2\} \text{ and:}\]
\[\begin{align*}
  b'_i &= 0, \text{ if no node in } A \text{ has label } l_i; \\
  b'_i &= 1, \text{ if there is exactly one node in } A \text{ with label } l_i; \\
  b'_i &= 2, \text{ if there are more than one node in } A \text{ with label } l_i.
\end{align*}\]

The following lemma states that histories of the node \(r\) in trees from \(\mathcal{T}\) depend only on the pattern and not on the tree itself.

**Lemma 4.2** Let \(A\) be any algorithm that solves the size discovery problem for all trees \(T \in \mathcal{T}\) using the labeling scheme \(L(T)\). If trees \(T_1\) and \(T_2\) have the same pattern, then for any \(t \geq 0\), the node \(r\) in \(T_1\) and the node \(r\) in \(T_2\) have the same history in round \(t\).

**Proof:** Let \(T_1\) and \(T_2\) be two trees with same pattern \((P(r), P(a))\). For \(j = 1, 2\), denote the node \(r\) in \(T_j\) by \(r_j\), the node \(a\) in \(T_j\) by \(a_j\), the set \(R\) in \(T_j\) by \(R_j\), and the set \(A\) in \(T_j\) by \(A_j\). For any \(t \geq 0\), we prove the following statements by simultaneous induction. (To prove the lemma, we need only the first of them).

1. \(H(r_1, t, L, A) = H(r_2, t, L, A)\).
2. \(H(a_1, t, L, A) = H(a_2, t, L, A)\).

3. For a node \(v_1\) in \(R_1\) and a node \(v_2\) in \(R_2\) with same label, \(H(v_1, t, L, A) = H(v_2, t, L, A)\).

4. For a node \(v_1\) in \(A_1\) and a node \(v_2\) in \(A_2\) with the same label, \(H(v_1, t, L, A) = H(v_2, t, L, A)\).

Since the patterns of the two trees are the same, we have \(l(r_1) = l(r_2)\), and \(l(a_1) = l(a_2)\).

Therefore, according to the definition of the history, the above statements are true for \(t = 0\).

Suppose that all the above statements are true for round \(t\). Consider the execution of the algorithm in round \(t + 1\) as follows:

**Induction step for (1):** The actions of nodes in \(A_1\) and \(A_2\) in round \(t + 1\) do not affect the histories of the nodes \(r_1\) and \(r_2\) in this round. Hence we have the following cases in round \(t + 1\).

(a) \(r_1\) transmits, \(a_1\) does not transmit, and no node in \(R_1\) transmits.

According to the definition of history, \(H(r_1, t+1, L, A) = [H(r_1, t, L, A), \lambda]\). Since \(H(r_1, t, L, A) = H(r_2, t, L, A)\) and \(r_1\) transmits in round \(t + 1\), then \(r_2\) also transmits in round \(t + 1\). Similarly \(a_2\) does not transmit in this round, since \(a_1\) does not transmit. We prove that no node in \(R_2\) transmits in round \(t + 1\). Suppose otherwise. Let \(v_2\) be a node in \(R_2\) that transmits in round \(t + 1\), and suppose that the label of \(v_2\) is \(l_i\). Therefore, in \(P(r)\), either \(b_i = 1\), or \(b_i = 2\).

If \(b_i = 1\), then \(v_2\) is the unique node with label \(l_i\) in \(R_2\). Since the patterns of the two trees are the same, therefore, there exists a unique node \(v_1\) in \(R_1\) with label \(l_i\). Since \(H(v_1, t, L, A) = H(v_2, t, L, A)\), by the induction hypothesis for (3), therefore \(v_1\) must transmit in round \(t + 1\), which is a contradiction with the fact that no node in \(R_1\) transmits. A similar statement holds for \(b_i = 2\). Hence no node in \(R_2\) transmits in round \(t + 1\). Therefore, \(H(r_2, t + 1, L, A) = [H(r_2, t, L, A), \lambda]\). This implies that \(H(r_1, t + 1, L, A) = H(r_2, t + 1, L, A)\).

(b) \(r_1\) transmits, \(a_1\) transmits, no node in \(R_1\) transmits.

Since \(H(r_1, t, L, A) = H(r_2, t, L, A)\) and \(H(a_1, t, L, A) = H(a_2, t, L, A)\), and \(r_1\) and \(a_1\) transmit in round \(t + 1\), therefore, \(r_2\) and \(a_2\) also transmit in round \(t + 1\). Hence \(H(r_1, t + 1, L, A) = [H(r_1, t, L, A), \lambda]\) and \(H(r_2, t + 1, L, A) = [H(r_2, t, L, A), \lambda]\). This implies that \(H(r_1, t + 1, L, A) = H(r_2, t + 1, L, A)\).

(c) \(r_1\) transmits, \(a_1\) does not transmit, some nodes in \(R_1\) transmit.

Similarly as in (b), we have \(H(r_1, t + 1, L, A) = [H(r_1, t, L, A), \lambda]\) and \(H(r_2, t + 1, L, A) = [H(r_2, t, L, A), \lambda]\). This implies that \(H(r_1, t + 1, L, A) = H(r_2, t + 1, L, A)\).

(d) \(r_1\) transmits, \(a_1\) transmits, some nodes in \(R_1\) transmit.

Similarly as in (b), we have \(H(r_1, t + 1, L, A) = [H(r_1, t, L, A), \lambda]\) and \(H(r_2, t + 1, L, A) = [H(r_2, t, L, A), \lambda]\). This implies that \(H(r_1, t + 1, L, A) = H(r_2, t + 1, L, A)\).

(e) \(r_1\) does not transmit, \(a_1\) does not transmit, no node in \(R_1\) transmits.

Since \(a_1\) does not transmit in round \(t + 1\), therefore \(a_2\) does not transmit in round \(t + 1\). Also, as explained in (a), no node in \(R_2\) transmits in round \(t + 1\). Therefore, \(H(r_1, t + 1, L, A) = [H(r_1, t, L, A), \lambda]\) and \(H(r_2, t + 1, L, A) = [H(r_2, t, L, A), \lambda]\). This implies that \(H(r_1, t + 1, L, A) = H(r_2, t + 1, L, A)\).
(f) \( r_1 \) does not transmit, \( a_1 \) transmits, no node in \( R_1 \) transmits.

Since \( a_1 \) transmits in round \( t+1 \), therefore \( a_2 \) transmits in round \( t+1 \). Also, as explained in (a), no node in \( R_2 \) transmits in round \( t+1 \). Therefore, according to the definition of history, \( H(r_1, t+1, L, A) = [H(r_1, t, L, A), [H(a_1, t, L, A)], H(r_2, t+1, L, A) = [H(r_2, t, L, A), [H(a_2, t, L, A)]. \n
Since \( H(r_1, t, L, A) = H(r_2, t, L, A) \) and \( H(a_1, t, L, A) = H(a_2, t, L, A) \), therefore, \( H(r_1, t+1, L, A) = H(r_2, t+1, L, A) \).

(g) \( r_1 \) does not transmit, \( a_1 \) does not transmit, some nodes in \( R_1 \) transmit.

Let \( v_1 \) be a node in \( R_1 \) with label \( l_i \) such that \( v_1 \) transmits in round \( t+1 \). Then by Lemma \[4.1\] all the nodes in \( R_1 \) with label \( l_i \) must transmit in round \( t+1 \). Suppose that the nodes with labels \( l_{i_1}, l_{i_2}, \ldots, l_{i_k} \) transmit in round \( t+1 \).

If each of the integers \( b_{i_1}, b_{i_2}, \ldots, b_{i_k} \) is 0, then no node in \( R_1 \) transmits which contradicts the assumption of case (g).

If at least two of the integers \( b_{i_1}, b_{i_2}, \ldots, b_{i_k} \) are 1, or at least one of them is 2, then there exist at least two nodes in \( R_1 \) and at least two nodes in \( R_2 \) that transmit in round \( t+1 \). Hence, a collision is heard at the node \( r_1 \) and a collision is heard at the node \( r_2 \). Therefore, \( H(r_1, t+1, L, A) = [H(r_1, t, L, A), *] \) and \( H(r_2, t+1, L, A) = [H(r_2, t, L, A), *] \). This implies that \( H(r_1, t+1, L, A) = H(r_2, t+1, L, A) \).

Otherwise, exactly one of the integers \( b_{i_1}, b_{i_2}, \ldots, b_{i_k} \) is 1, and all others are 0. W.l.o.g. let \( b_{i_1} \) be the unique integer 1. Then there is exactly one node \( v_1 \) with label \( l_{i_1} \) in \( R_1 \) and there is exactly one node \( v_2 \) with label \( l_{i_1} \) in \( R_2 \) which transmit in round \( t+1 \). Therefore, \( H(r_1, t+1, L, A) = [H(r_1, t, L, A), H(v_1, t, L, A)] \) and \( H(r_2, t+1, L, A) = [H(r_2, t, L, A), H(v_2, t, L, A)] \).

Since, \( H(v_1, t, L, A) = H(v_2, t, L, A) \) and \( H(r_1, t, L, A) = H(r_2, t, L, A) \), therefore, \( H(r_1, t+1, L, A) = H(r_2, t+1, L, A) \).

(h) \( r_1 \) does not transmit, \( a_1 \) transmits, some nodes in \( R_1 \) transmit.

Since \( a_1 \) transmits in round \( t+1 \), therefore, \( a_2 \) transmits in round \( t+1 \). Also, since some node in \( R_1 \) transmits in round \( t+1 \), therefore some node in \( R_2 \) transmits in round \( t+1 \), as explained in (a). Therefore, a collision is heard at \( r_1 \), and a collision is heard at \( r_2 \). Hence, \( H(r_1, t+1, L, A) = [H(r_1, t, L, A), *] \) and \( H(r_2, t+1, L, A) = [H(r_2, t, L, A), *] \). This implies that \( H(r_1, t+1, L, A) = H(r_2, t+1, L, A) \).

**Induction step for (2):** This is similar to the induction step for (1).

**Induction step for (3):** For \( j = 1, 2 \), the histories of the nodes in \( R_j \) in round \( t+1 \) do not depend on the action of the node \( a_j \) and the actions of the nodes in \( A_j \) in this round. Hence we have the following cases in round \( t+1 \).

(i) The node \( r_1 \) transmits and no node in \( R_1 \) transmits.

This implies that the node \( r_2 \) transmits and no node in \( R_2 \) transmits. Therefore, \( H(v_1, t+1, L, A) = [H(v_1, t, L, A), H(r_1, t, L, A)] \) and \( H(v_2, t+1, L, A) = [H(v_2, t, L, A), H(r_2, t, L, A)] \).

Since, \( H(r_1, t, L, A) = H(r_2, t, L, A) \) and \( H(v_1, t, L, A) = H(v_2, t, L, A) \), therefore, \( H(v_1, t+1, L, A) = H(v_2, t+1, L, A) \).

(ii) The node \( r_1 \) transmits and some nodes in \( R_1 \) transmit.
This implies that the node $r_2$ transmits and some nodes in $R_2$ transmit. There are two cases. If $v_1$ transmits, then $v_2$ also transmits in round $t+1$. Hence $H(v_1, t+1, L, A) = [H(v_1, t, L, A), \lambda]$ and $H(v_2, t+1, L, A) = [H(v_2, t, L, A), \lambda]$ and hence $H(v_1, t+1, L, A) = H(v_2, t+1, L, A)$. If $v_1$ does not transmit, then $v_2$ does not transmit either, in round $t+1$. Hence $H(v_1, t+1, L, A) = [H(v_1, t, L, A), H(r_1, t, L, A)]$ and $H(v_2, t+1, L, A) = [H(v_2, t, L, A), H(r_2, t, L, A)]$ and hence $H(v_1, t+1, L, A) = H(v_2, t+1, L, A)$.

(iii) The node $r_1$ does not transmit and some nodes in $R_1$ transmit.

Since $r_1$ does not transmit therefore $r_2$ does not transmit. According to the definition of history, $H(v_1, t+1, L, A) = [H(v_1, t, L, A), \lambda]$ and $H(v_2, t+1, L, A) = [H(v_2, t, L, A), \lambda]$, and hence $H(v_1, t+1, L, A) = H(v_2, t+1, L, A)$.

(iv) The node $r_1$ does not transmit and no node in $R_1$ transmits.

In this case, $H(v_1, t+1, L, A) = [H(v_1, t, L, A), \lambda]$ and $H(v_2, t+1, L, A) = [H(v_2, t, L, A), \lambda]$ and hence $H(v_1, t+1, L, A) = H(v_2, t+1, L, A)$.

**Induction step for (4):** This is similar to the induction step for (3).

Therefore, the lemma follows by induction. □

**Corollary 4.1** Let $\mathcal{H}_t$ be the set of all possible histories of the node $r$ in all trees in $\mathcal{T}$, in round $t$, and let $\mathcal{P}$ be the set of all possible patterns of trees in $\mathcal{T}$. Then $|\mathcal{H}_t| \leq |\mathcal{P}|$.

The following theorem gives the lower bound $\Omega(\log \log \Delta)$ on the length of a labeling scheme for size discovery, that matches the length of the labeling scheme used by Algorithm Size Discovery.

**Theorem 4.1** For any tree $T \in \mathcal{T}$ consider a labeling scheme $L(T)$. Let $\mathcal{A}$ be any algorithm that finds the size of $T$, for every tree $T \in \mathcal{T}$, using the scheme $L(T)$. Then there exists a tree $T' \in \mathcal{T}$, for which the length of the scheme $L(T')$ is $\Omega(\log \log \Delta)$.

**Proof:** It is enough to prove the theorem for sufficiently large $\Delta$. We prove the theorem by contradiction. Suppose that there exists an algorithm $\mathcal{A}$ that solves the size discovery problem in the class $\mathcal{T}$, in time $t$, with labels of length at most $\frac{1}{4} \log \log \Delta$. There are at most $z = 2(\log \Delta)^{\frac{1}{2}}$ possible different labels of at most this length. There are at most $z^2 3^{2z}$ different possible patterns for these $z$ labels. Therefore, by Corollary 4.1, the total number of histories of the node $r$, in time $t$, over the entire class $\mathcal{T}_r$, is at most $z^2 3^{2z} \leq (2(\log \Delta)^{\frac{1}{2}})^2 3^{4(\log \Delta)^{\frac{1}{2}}} < \frac{\Delta}{2} \leq |\mathcal{T}|$, for sufficiently large $\Delta$.

Therefore, by the pigeonhole principle, there exist two trees $T', T''$ in $\mathcal{T}$ such that the history of $r$ in $T'$ in time $t$ is the same as the history of $r$ in $T''$ in round $t$. This implies that the node $r$ in $T'$ and the node $r$ in $T''$ must behave identically in every round until round $t$, hence they must output the same size. This contradicts the fact that the trees $T'$ and $T''$ have different sizes. This completes the proof. □

**5 Conclusion**

We established the minimum length $\Theta(\log \log \Delta)$ of a labeling scheme permitting to find the size of arbitrary radio networks of maximum degree $\Delta$, with collision detection, and we designed a size...
discovery algorithm using a labeling scheme of this length. Our algorithm heavily uses the collision detection capability, hence the first open question is whether our result holds in radio networks without collision detection. Secondly, in this paper we were concerned only with the feasibility of the size discovery task with short labels. The running time of our algorithm is $O(Dn^2 \log \Delta)$, for $n$-node networks of diameter $D$ and maximum degree $\Delta$. We did not try to optimize this running time. A natural open question is: what is the fastest size discovery algorithm using the shortest possible labeling scheme, i.e., a scheme of length $\Theta(\log \log \Delta)$?

References

[1] S. Abiteboul, H. Kaplan, T. Milo, Compact labeling schemes for ancestor queries, Proc. 12th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2001), 547–556.

[2] B.S. Chlebus, L. Gasieniec, A. Gibbons, A. Pelc, W. Rytter, Deterministic broadcasting in ad hoc radio networks, Distributed Computing 15 (2002), 27-38.

[3] M. Chrobak, L. Gasieniec, W. Rytter, Fast broadcasting and gossiping in radio networks, Journal of Algorithms 43 (2002):177189.

[4] R. Cohen, P. Fraigniaud, D. Ilcinkas, A. Korman, D. Peleg, Label-guided graph exploration by a finite automaton, ACM Transactions on Algorithms 4 (2008).

[5] A. Czumaj, P. Davies, Exploiting spontaneous transmissions for broadcasting and leader election in radio networks, [arXiv:1703.01859 [cs.DC]]

[6] D. Dereniowski, A. Pelc, Drawing maps with advice, Journal of Parallel and Distributed Computing 72 (2012), 132–143.

[7] Y. Emek, P. Fraigniaud, A. Korman, A. Rosen, Online computation with advice, Theoretical Computer Science 412 (2011), 2642–2656.

[8] P. Fraigniaud, C. Gavoille, D. Ilcinkas, A. Pelc, Distributed computing with advice: Information sensitivity of graph coloring, Distributed Computing 21 (2009), 395–403.

[9] P. Fraigniaud, D. Ilcinkas, A. Pelc, Communication algorithms with advice, Journal of Computer and System Sciences 76 (2010), 222–232.

[10] P. Fraigniaud, D. Ilcinkas, A. Pelc, Tree exploration with advice, Information and Computation 206 (2008), 1276–1287.

[11] P. Fraigniaud, A. Korman, E. Lebhar, Local MST computation with short advice, Theory of Computing Systems 47 (2010), 920–933.

[12] E. Fusco, A. Pelc, Trade-offs between the size of advice and broadcasting time in trees, Algorithmica 60 (2011), 719–734.

[13] E. Fusco, A. Pelc, R. Petreschi, Topology recognition with advice, Information and Computation 247 (2016), 254-265.
[14] L. Gasieniec, A. Pagourtzis, I. Potapov, T. Radzik, Deterministic communication in radio networks with large labels. Algorithmica 47 (2007), 97-117.

[15] L. Gasieniec, D. Peleg, Q. Xin, Faster communication in known topology radio networks, Distributed Computing 19 (2007), 289-300.

[16] C. Gavoille, D. Peleg, S. Pérennes, R. Raz. Distance labeling in graphs, Journal of Algorithms 53 (2004), 85-112.

[17] M. Ghaffari, B. Haeupler, M. Khabbazian. Randomized broadcast in radio networks with collision detection. Proc. 32nd Annual ACM Symposium on Principles of Distributed Computing (PODC 2013), 325-334.

[18] C. Glacet, A. Miller, A. Pelc, Time vs. information tradeoffs for leader election in anonymous trees, Proc. 27th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2016), 600-609.

[19] B. Gorain, A. Pelc, Short labeling schemes for topology recognition in wireless tree networks, arXiv:1704.01927 [cs.DC].

[20] D. Ilcinkas, D. Kowalski, A. Pelc, Fast radio broadcasting with advice, Theoretical Computer Science, 411 (2012), 1544–1557.

[21] M. Katz, N. Katz, A. Korman, D. Peleg, Labeling schemes for flow and connectivity, SIAM Journal of Computing 34 (2004), 23–40.

[22] A. Korman, S. Kutten, D. Peleg, Proof labeling schemes, Distributed Computing 22 (2010), 215–233.

[23] D. Kowalski, A. Pelc, Leader election in ad hoc radio networks: a keen ear helps, Journal of Computer and System Sciences 79 (2013), 1164-1180.

[24] N. Nisse, D. Soguet, Graph searching with advice, Theoretical Computer Science 410 (2009), 1307–1318.

[25] A. Pelc, Activating anonymous ad hoc radio networks, Distributed Computing 19 (2007), 361-371.