Short Labeling Schemes for Topology Recognition in Wireless Tree Networks

Barun Gorain and Andrzej Pelc*

Département d'informatique, Université du Québec en Outaouais, Gatineau, Québec J8X 3X7, Canada.
Email: baruniitg123@gmail.com, pelc@uqo.ca

Abstract. We consider the problem of topology recognition in wireless (radio) networks modeled as undirected graphs. Topology recognition is a fundamental task in which every node of the network has to output a map of the underlying graph i.e., an isomorphic copy of it, and situate itself in this map. In wireless networks, 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. Nodes have labels which are (not necessarily different) binary strings. The length of a labeling scheme is the largest length of a label. We concentrate on wireless networks modeled by trees, and we investigate two problems.

- What is the shortest labeling scheme that permits topology recognition in all wireless tree networks of diameter D and maximum degree Δ?
- What is the fastest topology recognition algorithm working for all wireless tree networks of diameter D and maximum degree Δ, using such a short labeling scheme?

We are interested in deterministic topology recognition algorithms. For the first problem, we show that the minimum length of a labeling scheme allowing topology recognition in all trees of maximum degree $\Delta \geq 3$ is $\Theta(\log \log \Delta)$. For such short schemes, used by an algorithm working for the class of trees of diameter $D \geq 4$ and maximum degree $\Delta \geq 3$, we show almost matching bounds on the time of topology recognition: an upper bound $O(D\Delta)$, and a lower bound $\Omega(D\Delta^{\epsilon})$, for any constant $\epsilon < 1$.

Our upper bounds are proven by constructing a topology recognition algorithm using a labeling scheme of length $O(\log \log \Delta)$ and using time $O(D\Delta)$. Our lower bounds are proven by constructing a class of trees for which any topology recognition algorithm must use a labeling scheme of length at least $\Omega(\log \log \Delta)$, and a class of trees for which any topology recognition algorithm using a labeling scheme of length $O(\log \log \Delta)$ must use time at least $\Omega(D\Delta^{\epsilon})$, on some tree of this class.

keywords: topology recognition, wireless network, labeling scheme, feasibility, tree, time

* 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

The model and the problem. Learning the topology of an unknown network by its nodes is a fundamental distributed task in networks. Every node of the network has to output a map of the underlying graph, i.e., an isomorphic copy of it, and situate itself in this map. Topology recognition can be considered as a preprocessing procedure to many other distributed algorithms which require the knowledge of important parameters of the network, such as its size, diameter or maximum degree. It can also help to determine the feasibility of some tasks that depend, e.g., on symmetries existing in the network.

We consider wireless networks, also known as radio networks. Such a network is modeled as a simple undirected connected graph $G = (V, E)$. As it is usually assumed in the algorithmic theory of radio networks [2, 12, 13], 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. We do not assume collision detection: if more than one neighbor of a node v transmits in a given round, node v does not hear anything (except the background noise that it also hears when no neighbor transmits).

In this paper, we restrict attention to wireless networks modeled by trees, and we are interested in deterministic topology recognition algorithms. Topology recognition is formally defined as follows. Every node v of a tree T must output a tree T' and a node v' in this tree, such that there exists an isomorphism f from T to T', for which $f(v) = v'$. Topology recognition is impossible, if nodes do not have any a priori assigned labels, because then any deterministic algorithm forces all nodes to transmit in the same rounds, and no communication is possible. Hence we consider labeled networks. A *labeling scheme* for a network represented by a tree $T = (V, E)$ is any function $\mathcal{L}$ from the set V of nodes into the set S of finite binary strings. The string $\mathcal{L}(v)$ is called the label of the node v. Note that labels assigned by a labeling scheme are not necessarily distinct. The *length* of a labeling scheme $\mathcal{L}$ is the maximum length of any label assigned by it. We investigate two problems.

- What is the shortest labeling scheme that permits topology recognition in all wireless tree networks of diameter D and maximum degree Δ?
- What is the fastest topology recognition algorithm working for all wireless tree networks of diameter D and maximum degree Δ, using such a short labeling scheme?

Our results. For the first problem, we show that the minimum length of a labeling scheme allowing topology recognition in all trees of maximum degree $\Delta \geq 3$ is $\Theta(\log \log \Delta)$. For such short schemes, used by an algorithm working for the class of trees of diameter $D \geq 4$ and maximum degree $\Delta \geq 3$, we show almost matching bounds on the time of topology recognition: an upper bound $O(D\Delta)$, and a lower bound $\Omega(D\Delta^{\epsilon})$, for any constant $\epsilon < 1$.

Our upper bounds are proven by constructing a topology recognition algorithm using a labeling scheme of length $O(\log \log \Delta)$ and using time $O(D\Delta)$. Our lower bounds are proven by constructing a class of trees for which any topology recognition algorithm must use a labeling scheme of length at least $\Omega(\log\log\Delta)$, and a class of trees for which any topology recognition algorithm using a labeling scheme of length $O(\log\log\Delta)$ must use time at least $\Omega(D\Delta^{\epsilon})$, on some tree of this class.

These main results are complemented by establishing complete answers to both problems for very small values of D or Δ. For trees of diameter $D = 3$ and maximum degree $\Delta \geq 3$, the fastest topology recognition algorithm using a shortest possible scheme (of length $\Theta(\log\log\Delta)$) works in time $\Theta(\frac{\log\Delta}{\log\log\Delta})$. The same holds for trees of diameter $D = 2$ and maximum degree *at most* Δ, for $\Delta \geq 3$. Finally, if $\Delta = 2$, i.e., for the class of lines, the shortest labeling scheme permitting topology recognition is of constant length, and the best time of topology recognition using such a scheme for lines of diameter (length) at most D is $\Theta(\log D)$.

Our results should be contrasted with those from [11], where topology recognition was studied in a different model. The authors of [11] considered wired networks in which there are port numbers at each node, and communication proceeds according to the $\mathcal{LOCAL}$ model [21], where in each round neighbors can exchange all available information without collisions. In this model, they showed a simple topology recognition algorithm working for a labeling scheme of length 1 in time $O(D)$. Thus there was no issue of optimality: both the length of the labeling scheme and the topology recognition time for such a scheme were trivially optimal. Hence the authors focused on tradeoffs between the length of (longer) schemes and the time of topology recognition. In our scenario of wireless networks, the labeling schemes must be longer and algorithms for such schemes must be slower, in order to overcome collisions.

Related work. Algorithmic problems in radio networks modeled as graphs were studied for such tasks as broadcasting [2, 13], gossiping [2, 12] and leader election [19]. In some cases [2, 12] the topology of the network was unknown, in others [13] nodes were assumed to have a labeled map of the network and could situate themselves in it.

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 to perform network tasks more efficiently has been proposed in [1, 3–10, 14, 16–18, 20]. 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 [15] or different strings may be given to different nodes [9, 11], as in the case of the present paper. If strings may be different, they can be considered as labels assigned to nodes. Several authors studied the minimum size of advice (length of labels) required to solve the respective network problem in an efficient way. The framework of advice or labeling schemes permits to quantify the amount of information that nodes need for an efficient solution of a given network problem, regardless of the type of information that is provided.

In [3] the authors investigated the minimum size of advice that has to be given to nodes to permit graph exploration by a robot. In [18], 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 [7] the authors compared the minimum size of advice required to solve two information dissemination problems, using a linear number of messages. In [8] the authors established the size of advice needed to break competitive ratio 2 of an exploration algorithm in trees. In [9] it was shown that advice of constant size permits to carry on the distributed construction of a minimum spanning tree in logarithmic time. In [12] short labeling schemes were constructed with the aim to answer queries about the distance between any pair of nodes. In [5] the advice paradigm was used for online problems. In the case of [20] 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. In [16] the authors studied radio networks for which it is possible to perform centralized broadcasting in constant time. This is the only paper studying the size of advice in the context of radio networks. In [11] the authors studied the task of topology recognition in wired networks with port numbers. The differences between this scenario and our setting of radio networks, in the context of topology recognition, was discussed in the previous section.

2 Preliminaries and organization

Throughout the paper, D denotes the diameter of the tree and Δ denotes its maximum degree. The problem of topology recognition is non-trivial only for $D, \Delta \geq 2$, hence we make this assumption from now on.

According to the definition of labeling schemes, a label of any node should be a finite binary string. For ease of comprehension, we present our labels in a more structured way, as either finite sequences of binary strings, or pairs of such sequences, where each of the component binary strings is later used in the topology recognition algorithm in a particular way. It is well known that a sequence $(s_1, \ldots, s_k)$ of binary strings or a pair (σ_1, σ_2) of such sequences can be unambiguously coded as a single binary string whose length is a constant multiple of the sum of lengths of all binary strings s_i 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.

Let T be any rooted tree with root r, and let $L(T)$ be a labeling scheme for this tree. We say that a node u in T *reaches* r within time τ using algorithm $\mathcal{A}$ if there exists a simple path $u = u_0, u_1, \cdots, u_{k-1}, u_k = r$ and a sequence of integers $t_0 < t_1 < \cdots < t_{k-1} \leq \tau$, such that in round t_i, the node u_i is the only child of its parent u_{i+1} that transmits and the node u_{i+1} does not transmit in round t_i, according to algorithm $\mathcal{A}$.

We define the history $H(\mathcal{A}, \tau)$ of the root r of the tree T as the labeled subtree of T which is spanned by all the nodes that reach r within time τ, using algorithm $\mathcal{A}$. The history $H(\mathcal{A}, \tau)$ is the total information that node r can learn about the tree T in time τ, using algorithm $\mathcal{A}$.

3 A lower bound on the length of labeling schemes

As mentioned in the Introduction, topology recognition without any labels cannot be performed in any tree because no information can be successfully transmitted in an unlabeled radio network. Hence, the length of a labeling scheme permitting topology recognition must be a positive integer. In this section we show a lower bound $\Omega(\log\log\Delta)$ on the length of labeling schemes that permit topology recognition for all trees with maximum degree $\Delta \geq 3$.

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

The following result shows that any labeling scheme allowing topology recognition in trees of maximum degree Δ must have length $\Omega(\log\log\Delta)$.

Theorem 1. *For any tree $T \in \mathcal{T}$ consider a labeling scheme LABEL(T). Let TOPO be any topology recognition algorithm that solves topology recognition for every tree $T \in \mathcal{T}$ using the scheme LABEL(T). Then there exists a tree $T' \in \mathcal{T}$, for which the length of the scheme LABEL(T') is $\Omega(\log\log\Delta)$.*

4 Time for maximum degree $\Delta \geq 3$ and diameter $D \geq 4$

In this section, we present our main results on the time of topology recognition, using the shortest possible labeling schemes (those of length $\Theta(\log\log\Delta)$) for trees of maximum degree $\Delta \geq 3$ and diameter $D \geq 4$. We propose an algorithm using a labeling scheme of length $\Theta(\log\log\Delta)$ and working in time $O(D\Delta)$, and prove an almost matching lower bound $\Omega(D\Delta^{\epsilon})$ on the time of such schemes, for any constant $\epsilon < 1$.

4.1 The main algorithm

Let T be a rooted tree of diameter D and maximum degree Δ. It has either a central node or a central edge, depending on whether D is even or odd. If D is even, then the central node is the unique node in the middle of every simple path of length D, and if D is odd, then the central edge is the unique edge in the middle of every simple path of length D. For the sake of description, we choose the central node or one of the endpoints of the central edge as the root r of T. Let $h = \lceil D/2 \rceil$ be the height of this tree. The *level* of any node v is its distance from the root. For any node v we denote by T_v the subtree of T rooted at v.

We propose an algorithm that solves topology recognition in time $O(D\Delta)$, using a labeling scheme of length $O(\log\log\Delta)$. The structure of the tree will be transmitted bottom up, so that the root learns the topology of the tree, and then transmits it to all other nodes. The main difficulty is to let every node know the round number ρ in which it has to transmit, so that it is the only node among its siblings that transmits in round

ρ, and consequently its parent gets the message. Due to very short labels, ρ cannot be explicitly given to the node as a part of its label. We overcome this difficulty by carefully coding ρ for a node v, using the labels given to the nodes of the subtree rooted at v, so that v can unambiguously decode ρ.

A node v in T is called *heavy*, if $|V(T_v)| \geq \frac{1}{4}(\lfloor \log \Delta \rfloor + 1)$. Otherwise, the node is called *light*. Note that the root is a heavy node. For a heavy node v, choose a subtree T'_v of T_v rooted at v, of size $\lceil \frac{1}{4}(\lfloor \log \Delta \rfloor + 1) \rceil$.

First, we define the labeling scheme Λ. The label $\Lambda(v)$ of each node v contains two parts. The first part is a vector of markers that are binary strings of constant length, used to identify nodes with different properties. The second part is a vector of 5 binary strings of length $O(\log \log \Delta)$ that are used to determine the time when the node should transmit.

Below we describe how the markers are assigned to different nodes of T.

1. Mark the root r by the marker 0, and mark one of the leaves at maximum depth by the marker 1.

2. Mark all the nodes in T'_r by the marker 2.

3. Mark every heavy node by the marker 3, and mark every light node by the marker 4.

4. For every heavy node v all of whose children are light, mark all the nodes of T'_v by the marker 5.

5. For every light node v whose parent is heavy, mark all the nodes in T_v by the marker 6.

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, $M(i) = 0$.

In order to describe the second part of each label, we define an integer t_v for every heavy node $v \neq r$, and an integer z_v, for every light node v whose parent is heavy. We define t_v, for a heavy node v at level $l > 0$, to identify the time slot in which v will transmit according to the algorithm. The definition is by induction on l. For $l = 1$, let $v_1, v_2, \ldots, v_x$, be the heavy children of r. Set $t_{v_i} = i$. Suppose that t_v is defined for every heavy node v at level l. Let v be a heavy node at level l. Let $u_1, u_2, \ldots, u_y$ be the heavy children of v. We set $t_{u_1} = t_v$, and we define t_{u_j}, for $2 \leq j \leq y$, as distinct integers from the range $\{1, \ldots, y\} \setminus \{t_v\}$. This completes the definition of t_v, for all heavy nodes $v \neq r$.

We now define z_v, for a light node v whose parent is heavy, to identify the time slot in which v will transmit according to the algorithm. Let S_i be a maximal sequence of non-isomorphic rooted trees of i nodes. There are at most $2^{2(i-1)}$ such trees. Let $\mathcal{S}$ be the sequence which is the concatenation of $S_1, S_2, \ldots, S_{\lceil \frac{1}{4}(\lfloor \log \Delta \rfloor + 1) \rceil - 1}$. Let q be the length of $\mathcal{S}$. Then $q \leq 2^{2(\frac{1}{4}(\lfloor \log \Delta \rfloor + 1))} \leq \sqrt{2\Delta}$. Note that the position of any tree of i nodes in $\mathcal{S}$ is at most 2^{2i-1}. Let $\mathcal{S} = (T_1, T_2, \ldots, T_q)$. For a light node v whose parent is heavy, we define $z_v = k$, if T_v and T_k are isomorphic.

The second part of each label is a vector L of length 5, whose terms $L(i)$ are binary strings of length $O(\log\log \Delta)$. Initialize all terms $L(i)$ for every node v to 0. We now describe how some of these terms are changed for some nodes. They are defined as follows.

1. All the nodes which get $M(2) = 1$ are the nodes of T'_r. There are exactly $\lceil\frac{1}{4}(\lfloor\log \Delta\rfloor + 1)\rceil$ nodes in T'_r. All nodes in T'_r are assigned distinct ids which are binary representations of the integers 1 to $\lceil\frac{1}{4}(\lfloor\log \Delta\rfloor + 1)\rceil$. Let s be the string of length $(\lfloor\log \Delta\rfloor + 1)$ which is the binary representation of the integer Δ. Let $b_1, b_2, \cdots, b_{\lceil\frac{1}{4}(\lfloor\log \Delta\rfloor+1)\rceil}$ be the substrings of s, each of length at most 4, such that s is the concatenation of the substrings $b_1, b_2, \cdots, b_{\lceil\frac{1}{4}(\lfloor\log \Delta\rfloor+1)\rceil}$. The term $L(0)$ corresponding to a node whose id is i, is set to the pair $(B(i), b_i)$, where $B(i)$ is the binary representation of the integer i. The intuitive role of the term $L(0)$ is to code the integer Δ in the nodes of the tree T'_r.

2. Let v be a node with $M(3) = 1$, and $M(5) = 1$, i.e, let v be a heavy node whose all children are light. All nodes in T'_v are assigned distinct ids which are binary representations of integers 1 to $\lceil\frac{1}{4}(\lfloor\log \Delta\rfloor + 1)\rceil$. Let s be the string of length $(\lfloor\log \Delta\rfloor + 1)$ which is the binary representation of the integer t_v. Let $b_1, b_2, \cdots, b_{\lceil\frac{1}{4}(\lfloor\log \Delta\rfloor+1)\rceil}$ be the substrings of s, each of length at most 4, such that s is the concatenation of the substrings $b_1, b_2, \cdots, b_{\lceil\frac{1}{4}(\lfloor\log \Delta\rfloor+1)\rceil}$. The term $L(1)$ corresponding to a node whose id is i, is set to the pair $(B(i), b_i)$, where $B(i)$ is the binary representation of the integer i. The intuitive role of the term $L(1)$ is to code the integer t_v, for a heavy node v whose all children are light, in the nodes of the tree T'_v.

3. Let v be a node with $M(3) = 1$, i.e., a heavy node. Let u be the parent of v. If $t_u = t_v$, set $L(2) = 1$ for the node v. The intuitive role of the term $L(2)$ at a heavy node v is to tell its parent u what is the value of t_u.

4. Let v be a node with $M(4) = 1$ and $M(6) = 1$, i.e, let v be a light node whose parent is heavy. All nodes in T_v are assigned distinct ids which are binary representations of the integers 1 to p, where p is the size of T_v. Let s be the string of length at most $2p$ which is the binary representation of the integer z_v. Let $b_1, b_2 \cdots, b_p$ be the substrings of s, each of length at most 2, such that s is the concatenation of the substrings $b_1, b_2 \cdots, b_p$. The term $L(3)$ of the node whose id is i is set to the pair $(B(i), b_i)$, where $B(i)$ is the binary representation of the integer i. The intuitive role of the term $L(3)$ is to code the integer z_v, for a light node v whose parent is heavy, in the nodes of the tree T_v.

5. Let v be a node with $M(3) = 1$, i.e., a heavy node. Partition all light children u of v into sets with the same value of z_u. Consider any set $\{u_1, u_2, \ldots, u_a\}$ in this partition. Let s be the binary representation of the integer a and let $b_1, b_2, \cdots, b_{\lceil\frac{1}{4}(\lfloor\log a\rfloor+1)\rceil}$ be the substrings of s, each of length at most 4, such that s is the concatenation of the substrings $b_1, b_2, \cdots, b_{\lceil\frac{1}{4}(\lfloor\log a\rfloor+1)\rceil}$.

For node u_i, where $i \leq \lceil\frac{1}{4}(\lfloor\log a\rfloor+1)\rceil$, the term $L(4)$ is set to the pair $(B(i), b_i)$, where $B(i)$ is the binary representation of the integer i, for $1 \leq i \leq \lfloor\log a\rfloor + 1$, and b_i is the ith bit of the binary representation of a. The intuitive role of the term $L(4)$ is to force two light children v_1 and v_2 of the same heavy parent, such that $z_{v_1} = z_{v_2}$, to transmit in different rounds.

6. For any node v the term $L(5)$ is set to the binary representation of the integer $\lceil \frac{1}{4}(\lfloor \log \Delta \rfloor + 1) \rceil$. This term will be used in a gossiping algorithm that plays the role of a subroutine in our algorithm.

Notice that the length of each $L(j)$ defined above is of length $O(\log \log \Delta)$ for every node, and there is no ambiguity in setting these terms, as every term for a node is modified at most once. This completes the description of our labeling scheme whose length is $O(\log \log \Delta)$.

Algorithm `Tree Topology Recognition`

The algorithm consists of four procedures, namely Procedure `Parameter Learning`, Procedure `Slot Learning`, Procedure `T-R` and Procedure `Final`. They are called in this order by the algorithm. In the first two procedures we will use the simple gossiping algorithm `Round-Robin` which enables nodes of any graph of size at most m with distinct ids from the set $\{1, \ldots, m\}$ to gossip in time m^2, assuming that they know m and that each node with id i has an initial message μ_i. The time segment $1, \ldots, m^2$ is partitioned into m segments of length m, and the node with id i transmits in the ith round of each segment. In the first time segment, each node with id i transmits the message (i, μ_i). In the remaining $m - 1$ time segments, nodes transmit all the previously acquired information. Thus at the end of algorithm `Round-Robin`, all nodes know the entire topology of the network, with nodes labeled by pairs (i, μ_i).

Procedure `Parameter Learning`

The aim of this procedure is for every node of the tree to learn the maximum degree Δ, the level of the tree to which the node belongs, and the height h of the tree.

The procedure consists of two stages. The first stage is executed in rounds $1, \ldots, m^2$, where $m = \lceil \frac{1}{4}(\lfloor \log \Delta \rfloor + 1) \rceil$, and consists of performing algorithm `Round-Robin` by the nodes with $M(2) = 1$, i.e., the nodes in T'_r. Each such node uses its id i written in the first component of the term $L(0)$, uses its label as μ_i, and takes m as the integer whose representation is given in the term $L(5)$.

After this stage, the node with $M(0) = 1$, i.e., the root r, learns all pairs $(B(1), b_1)$, ..., $(B(m), b_m)$, where $B(i)$ is the binary representation of the integer i, corresponding to the term $L(0)$ at the respective nodes. It computes the concatenation s of the strings $b_1, b_2, \ldots, b_m$. This is the binary representation of Δ.

The second stage of the procedure starts in round $m^2 + 1$. In round $m^2 + 1$, the root r transmits the message μ that contains the value of Δ. A node v, which receives the message μ at time $m^2 + i$ for the first time, sets its level as i and transmits μ. When the node u with $M(1) = 1$, i.e., a deepest leaf, receives μ in round $m^2 + j$, it sets its level as $h = j$, learns that the height of the tree is h, and transmits the pair (h, h) in the next round. Every node at level l, after receiving the message $(h, l + 1)$ (from a node of level $l + 1$) learns h and transmits the pair (h, l). After receiving the message $(h, 1)$, the root r transmits the message μ' that contains the value h. Every node learns h after receiving it for the first time and retransmits μ', if its level is less than h. The stage, and hence the entire procedure, ends in round $m^2 + 3h$.

Procedure `Slot Learning`

The aim of this procedure is for every heavy node all of whose children are light, and for every light node whose parent is heavy, to learn the time slot in which it should transmit. Moreover, at the end of the procedure, every light node v learns T_v.

Let $t_0 = m^2 + 3h$, where $m = \lceil \frac{1}{4}(\lfloor \log \Delta \rfloor + 1) \rceil$. The total number of rounds reserved for this procedure is $2m^2$. The procedure starts in round $t_0 + 1$ and ends in round $t_0 + 2m^2$. The procedure consists of two stages. The first stage is executed in rounds $t_0 + 1, \ldots, t_0 + m^2$, and consists of performing algorithm `Round-Robin` by the nodes with $L(1) \neq 0$, i.e., the nodes in T'_v, for a heavy node v all of whose children are light. Each such node uses its id i written in the first component of the term $L(1)$, uses its label as μ_i, and takes m as the integer whose representation is given in the term $L(5)$. After this stage, each node v with $M(3) = 1$ and $M(5) = 1$, i.e., a heavy node all of whose children are light, 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 term $L(1)$ at the respective nodes. It computes the concatenation s of the strings $b_1, b_2, \ldots, b_m$. This is the binary representation of the integer t_v, which will be used to compute the time slot in which node v will transmit in the next procedure.

The second stage is executed in rounds $t_0 + m^2 + 1, \ldots, t_0 + 2m^2$, and consists of performing algorithm `Round-Robin` by the nodes with $L(2) \neq 0$, i.e., the nodes in T_v, for a light node v whose parent is heavy. Each such node uses its id i written in the first component of the term $L(3)$, uses its label as μ_i, and takes m as the integer whose representation is given in the term $L(5)$. After this stage, each node v with $M(4) = 1$ and $M(6) = 1$, i.e., a light node whose parent is heavy, learns all pairs $(B(1), b_1), \ldots, (B(k), b_k)$, where $k < m$ and $B(i)$ is the binary representation of the integer i, corresponding to the term $L(3)$ at the respective nodes. Node v computes the concatenation s of the strings $b_1, b_2, \ldots, b_k$. This is the binary representation of the integer z_v, which will be used to compute the time slot in which node v will transmit in the next procedure. Moreover, each node w in T_v learns T_w because it knows the entire tree T_v with all id's. The stage, and hence the entire procedure, ends in round $t_1 = t_0 + 2m^2$.

Procedure `T-R`

The aim of this procedure is learning the topology of the tree by the root.

All heavy nodes and all light nodes whose parent is heavy transmit in this procedure. The procedure is executed in h epochs. The number of rounds reserved for an epoch is 2Δ. The first Δ rounds of an epoch are reserved for transmissions of heavy nodes and the last Δ rounds of an epoch are reserved for transmissions of light nodes whose parent is heavy. The epoch j starts in round $t_1 + 2(j-1)\Delta + 1$ and ends in round $t_1 + 2j\Delta$. All the nodes at level $h - i + 1$ which are either heavy nodes or light nodes with a heavy parent transmit in the epoch i. When a node v transmits in some epoch, it transmits a message $(\Lambda(v), T_v, C)$, where $C = t_v$, if v is a heavy node, and $C = 0$, if it is a light node. Below we describe the steps that a node performs in the execution of the procedure, depending on its label.

Let v be a node with $M(4) = 1$ and $M(6) = 1$, i.e., v is a light node whose parent is heavy. The node v transmit in this procedure if $L(4) \neq 0$. Let the level of v (learned in the execution of Procedure `Parameter Learning`) be l. Let the first component of the term $L(4)$ be the binary representation of the integer $c > 0$. The node v already knows the value z_v which it learned in the execution of Procedure `Slot Learning`. Knowing Δ, node v computes the list $\mathcal{S} = (T_1, T_2, \ldots, T_q)$ of trees (defined above) which unambiguously depends on Δ. The node v transmits the message $(\Lambda(v), T_{z_v}, 0)$ in round $t_1 + 2(h-l)\Delta + \Delta + (z_v - 1)\lceil \frac{1}{4}(\lfloor \log \Delta \rfloor + 1) \rceil + c$. We will show that node v is the only node among its siblings that transmits in this round.

Let v be a node with $M(3) = 1$ and $M(5) = 1$, i.e., v is a heavy node all of whose children are light. Let l be the level of v. All the children of v are light nodes with a heavy parent. They are at level $l - 1$. Let $u_1, u_2, \ldots, u_k$ be those children from which v received messages in the previous epoch. First, the node v partitions the nodes u_1, $u_2, \ldots, u_k$ into disjoint sets $R_1, R_2, \cdots, R_e$ such that all nodes in the same set have sent the message with same tree Q. For each such set R_d, $1 \leq d \leq e$, let Q_d be the tree sent by nodes from R_d. The node v got all pairs $(B(1), b_1), \ldots, (B(x), b_x)$, where $x = |R_d| < m$ and $B(i)$ is the binary representation of the integer i, corresponding to the term $L(4)$ at its children in R_d. Node v computes the concatenation s of the strings $b_1, b_2, \ldots, b_k$. Let y_d be the integer whose binary representation is s. After computing all y_d's, for $1 \leq d \leq e$, v computes the tree T_v, by attaching y_d copies of the the tree Q_d to v for $d = 1, \ldots, e$. The node v transmits the message $(\Lambda(v), T_v, t_v)$ in round $t_1 + 2(h-l)\Delta + t_v$. We will show that node v is the only node among its siblings that transmits in this round.

Let v be a node with $M(3) = 1$ and $M(5) = 0$, i.e., v is a heavy node who has at least one heavy child. Let $u_1, \ldots, u_{k_1}$ be the light children of v from which v received a message in the previous epoch, and let $u'_1, \ldots, u'_{k_2}$ be the heavy children of v from which v received a message in the previous epoch. The node v computes the tree T_v rooted at v as follows. It first attaches trees rooted at its light children, using the messages it received from them, in the same way as explained in the previous case. Then, it attaches trees rooted at its heavy children. These trees are computed from the code β in the message from each of the heavy children of v. Let u' be the unique heavy child of v for which the term $L(5) = 1$. The node v computes t_v which is equal to the term C in the message it received from the node u'. The node v transmits the message $(\Lambda(v), T_v, t_v)$ in round $t_1 + 2(h-l)\Delta + t_v$. We will show that node v is the only node among its siblings that transmits in this round.

Procedure `Final`

The aim of this procedure is for every node of the tree to learn the topology of the tree and to place itself in the tree. The procedure starts in round $t_1 + 2h\Delta + 1$ and ends in round $t_1 + 2h\Delta + h$. In round $t_1 + 2h\Delta + 1$, the root r transmits the message that contains the tree T_r. In general, every node v transmits a message exactly once in

Procedure `Final`. This message contains the sequence $(T_r, T_{w_p}, \ldots, T_{w_1}, T_v)$, where w_i is the ancestor of v at distance i. In view of the fact that every node v already knows T_v at this point, after receiving a message containing the sequence $(T_r, T_{w_p}, \ldots, T_{w_1})$ in round j, a node v transmits the sequence $(T_r, T_{w_p}, \ldots, T_{w_1}, T_v)$ in round $j+1$, if its level is less than h.

A node v outputs the tree T_r, and identifies itself as one of the nodes in T_r for which the subtrees rooted at their ancestors in each level starting from the root are isomorphic to the trees in the sequence $(T_r, T_{w_p}, \ldots, T_{w_1}, T_v)$. (Notice that there may be many such nodes). The procedure ends in round $t_1 + 2h\Delta + h$, when all nodes place themselves in T_r and output T_r.

Theorem 2. *Upon completion of Algorithm* `Tree Topology Recognition`*, all nodes of a tree correctly output the topology of the tree and place themselves in it. The algorithm uses labels of length $O(\log \log \Delta)$ and works in time $O(D\Delta)$, for trees of maximum degree Δ and diameter D.*

4.2 The lower bound

In this section, we prove that any topology recognition algorithm using a labeling scheme of length $O(\log \log \Delta)$ must use time at least $\Omega(D\Delta^{\epsilon})$, for any constant $\epsilon < 1$, on some tree of diameter $D \geq 4$ and maximum degree $\Delta \geq 3$. We split the proof of this lower bound into three parts, corresponding to different ranges of the above parameters, as the proof is different in each case.

Case 1: Δ bounded, D unbounded. In this case we need to show a lower bound $\Omega(D)$.

Lemma 1. *Let $D \geq 4$ be any integer, let $\Delta \geq 3$ be any integer constant and let $c > 1$ be any real constant. For any tree T of maximum degree Δ consider a labeling scheme LABEL(T) of length at most $c \log \log \Delta$. Let $TOPO$ be any algorithm that solves topology recognition for every tree T of maximum degree Δ using the labeling scheme LABEL(T). Then there exists a tree T of maximum degree Δ and diameter D for which $TOPO$ must take time $\Omega(D)$.*

Case 2: Δ unbounded, D bounded. In this case, we need to show a lower bound $\Omega(\Delta^{\epsilon})$, for any constant $\epsilon < 1$. The following lemma proves a stronger result.

Lemma 2. *Let $\Delta \geq 3$ be any integer, let $D \geq 4$ be any integer constant, and let $c > 0$ be any real constant. For any tree T of maximum degree Δ, consider a labeling scheme LABEL(T) of length at most $c \log \log \Delta$. Let $TOPO$ be an algorithm that solves topology recognition for every tree of maximum degree Δ and diameter D using the labeling scheme LABEL(T). Then there exists a tree T of maximum degree Δ and diameter D for which $TOPO$ must take time $\Omega(\frac{\Delta}{(\log \Delta)^c})$.*

Case 3: unbounded Δ and D. Let $\Delta \geq 3$, $D \geq 4$ be integers. We first assume that D is even. The case when D is odd will be explained later. It is enough to prove the lower bound for $D \geq 6$. Let $h = \lfloor \frac{D}{6} \rfloor$ and $g = \frac{D}{2} - h$. Then $2h \leq g \leq 2h+2$. Let P be a line of length g with nodes $v_1, v_2, \cdots, v_{g+1}$, where v_1 and v_{g+1} are the endpoints of P. We construct from P a class of trees called *sticks* as follows.

Let $x = (x_1, x_2, \cdots, x_g)$ be a sequence of integers, with $0 \leq x_i \leq \Delta - 2$. Construct a tree P_x by attaching x_i leaves to the node v_i for $1 \leq i \leq g$. Let $\mathcal{P}$ be the set of all sticks constructed from P. Then $|\mathcal{P}| = (\Delta-1)^g$. Let $\mathcal{P} = \{P_1, P_2, \cdots, P_{(\Delta-1)^g}\}$.

Let S be a rooted tree of height h, with root r of degree $\Delta - 1$, and with all other non-leaf nodes of degree Δ. The nodes in S are called *basic nodes*. Let $Z = \{w_1, w_2, \cdots, w_z\}$, where $z = (\Delta-1)^h$, be the set of leaves of S. Consider a sequence $y = (y_1, y_2, \cdots, y_z)$, for $1 \leq y_i \leq (\Delta-1)^g$. We construct a tree T_y from S by attaching to it the sticks in the following way: each leaf w_i is identified with the node v_1 of the stick P_{y_i}, for $1 \leq i \leq z$. We will say that the stick P_{y_i} is *glued* to node w_i. The diameter of each tree T_y is D. For odd D, do the above construction for $D-1$ and attach one additional node of degree 1 to one of the leaves.

Let $\mathcal{T}(\Delta, D)$ be a maximal set of pairwise non-isomorphic trees among the trees T_y. Then, $|\mathcal{T}(\Delta, D)| \geq \frac{((\Delta-1)^g)^z}{z!} \geq \frac{((\Delta-1)^g)^z}{z!} \geq (\Delta-1)^{h(\Delta-1)^h}$.

Consider any time $\tau > 0$. For any tree $T \in \mathcal{T}(\Delta, D)$, consider any labeling scheme $L(T)$ and let $\mathcal{A}$ be any algorithm that solves topology recognition in every tree $T \in \mathcal{T}(\Delta, D)$ in time τ, using the labeling scheme $L(T)$. The following lemma gives an upper bound on the number of basic nodes that can belong to a history of the root r.

Lemma 3. *Let B be the number of basic nodes of level i that can reach r within time τ, according to algorithm $\mathcal{A}$. Then $B \leq \frac{\tau^i}{i!}$ if $\tau \geq i$, and $B = 0$, otherwise.*

The next lemma gives a lower bound on the time of topology recognition for the class $\mathcal{T}(\Delta, D)$.

Lemma 4. *Let $\epsilon < 1$ be any positive real constant, and let $c > 1$ be any real constant. For any tree $T \in \mathcal{T}(\Delta, D)$, consider a labeling scheme LABEL(T) of length at most $c \log\log \Delta$. Then there exist integers $\Delta_0, D_0 > 0$ such that any algorithm that solves topology recognition for every tree $T \in \mathcal{T}(\Delta, D)$, where $\Delta \geq \Delta_0$ and $D \geq D_0$, using the scheme LABEL(T), must take time $\Omega(D\Delta^\epsilon)$ for some tree $T \in \mathcal{T}(\Delta, D)$.*

Proof. Wee first do the proof for even D. Consider an algorithm *TOPO* that solves topology recognition for every tree $T \in \mathcal{T}(\Delta, D)$ in time $\tau \leq (\frac{D}{6} - 1)\Delta^\epsilon \leq h\Delta^\epsilon$ with a labeling scheme $LABEL(T)$ of length at most $c \log\log \Delta$. For a scheme of this length, there are at most $2^{c \log\log \Delta + 1} = 2(\log \Delta)^c$ different possible labels. According to Lemma 3, for $1 \leq i \leq h$ the number of basic nodes of level i, that reach r within time τ is at most $\frac{\tau^i}{i!}$, if $\tau \geq i$, otherwise there are no such nodes.

Denote by q the total number of basic nodes that reach r within time τ. If $\tau \geq h$, then $q \leq \sum_{i=1}^{h} \frac{\tau^i}{i!} \leq h\frac{\tau^h}{h} = h\frac{(h\Delta^\epsilon)^h}{h!}$. We know that $\log(h!) = h\log h - \frac{h}{\ln 2} + \frac{1}{2}\log h + O(1) \geq h\log h - \frac{h}{\ln 2}$. Since $\ln 2 > \frac{1}{2}$, we have $\log(h!) > h\log h - 2h$. Therefore, $h! > \frac{h^h}{2^{-2h}}$, and hence $q \leq h\Delta^{h\epsilon}2^{2h}$. If $\tau < h$, then $q \leq \sum_{i=1}^{\tau} \frac{\tau^i}{i!} \leq \tau\Delta^{\tau\epsilon}2^{2\tau} \leq h\Delta^{h\epsilon}2^{2h}$. Therefore, $q \leq h\Delta^{h\epsilon}2^{2h}$, for all $\tau > 0$.

The number of different unlabeled sticks is at most $(\Delta-1)^{2h+2}$. Nodes of each such stick can be labeled with labels of length at most $\lfloor c\log\log\Delta\rfloor$ in at most $(2(\log\Delta)^c)^{(2h+2)\Delta}$ ways, because each stick can have at most $(2h+2)\Delta$ nodes. Therefore, the number of different labeled sticks is at most $p=(\Delta-1)^{2h+2}\,(2(\log\Delta)^c)^{(2h+2)\Delta}$.

The history of the root r of a tree $T\in\mathcal{T}(\Delta,D)$ may include some nodes from a stick in T only if the basic node at level h to which this stick is glued is a node in the history. The maximum information that the root can get from a basic node v at level h, but not from any other node at this level, is the information about the whole labeled stick glued to v.

The number of possible histories $H(TOPO,\tau)$ of the node r is at most the product of the number of possible labelings of the basic nodes in $H(TOPO,\tau)$ and the number of possible gluings of labeled sticks to them. Since there are at most q basic nodes in $H(TOPO,\tau)$, there are at most $(2(\log\Delta)^c)^q$ possible labelings of these nodes. Since there are at most p labeled sticks to choose from, the number of possible gluings of labeled sticks to the basic nodes in $H(TOPO,\tau)$ is at most p^q. Therefore, the number of possible histories $H(TOPO,\tau)$ of the node r is at most $2^q(\log\Delta)^{cq}p^q=(2p(\log\Delta)^c)^q$. Let $X=(2p(\log\Delta)^c)^q$. We have $\log X=q(\log p+1+c\log\log\Delta)=q+q\log p+qc\log\log\Delta$. Also, $\log p=(2h+2)\log(\Delta-1)+(2h+2)\Delta(1+c\log\log\Delta)$. Therefore, $\log X=q(1+\log p+c\log\log\Delta)$
$=q\,(1+(2h+2)\log(\Delta-1)+(2h+2)\Delta(1+c\log\log\Delta)+c\log\log\Delta)\le 5qc(2h+2)\Delta\log\Delta\le 5h\Delta^{h\epsilon+1}2^{2h}c(2h+2)\log\Delta$. Also, $\log|\mathcal{T}(\Delta,D)|\ge h(\Delta-1)^h\log(\Delta-1)$. Now, for any Δ and for sufficiently large h, we have $5h\Delta^{h\epsilon+1}2^{2h}c(2h+2)<\frac{1}{2}h\Delta^h$. Therefore, $5h\Delta^{h\epsilon+1}2^{2h}c(2h+2)\log\Delta<\frac{1}{2}h\Delta^h\log\Delta<h(\Delta-1)^h\log(\Delta-1)$, for sufficiently large Δ and sufficiently large h.

It follows that, for sufficiently large h and Δ, we have $\log X<\log|\mathcal{T}(\Delta,D)|$. Therefore, there exist integers Δ_0 and D_0 such that $X<|\mathcal{T}(\Delta,D)|$, for all $\Delta\ge\Delta_0$ and $D\ge D_0$. Hence, for $\Delta\ge\Delta_0$ and $D\ge D_0$, there exist two trees T_1 and T_2 in $\mathcal{T}(\Delta,D)$ whose roots have the same history. Therefore, the root r in T_1 and the root r in T_2 output the same tree as the topology, within time τ. This is a contradiction, which proves the lemma for even D. For odd D, the same proof works with D replaced by $D-1$. $\square$

Lemmas 1, 2, and 4 imply the following theorem.

Theorem 3. *Let $\epsilon<1$ be any positive real number. For any tree T of maximum degree $\Delta\ge 3$ and diameter $D\ge 4$, consider a labeling scheme of length $O(\log\log\Delta)$. Then any topology recognition algorithm using such a scheme for every tree T must take time $\Omega(D\Delta^\epsilon)$ for some tree.*

5 Time for small maximum degree Δ or small diameter D

In this section we solve our problem for the remaining cases of small parameters Δ and D, namely, in the case when $\Delta\le 2$ or $D\le 3$. We start with the case of small diameter D.

5.1 Diameter $D = 3$

Theorem 4. *The optimal time for topology recognition in the class of trees of diameter $D = 3$ and maximum degree $\Delta \geq 3$, using a labeling scheme of length $\Theta(\log\log\Delta)$, is $\Theta(\frac{\log\Delta}{(\log\log\Delta)})$.*

5.2 Diameter $D = 2$

We now consider the case of trees of diameter 2, i.e., the class of stars. Since there is exactly one star of a given maximum degree Δ, the problem of topology recognition for $D = 2$ and a given maximum degree Δ is trivial. A meaningful variation of the problem for $D = 2$ is to consider all trees (stars) of maximum degree *at most* Δ, for a given Δ.

Theorem 5. *The optimal time for topology recognition in the class of trees of diameter $D = 2$ (i.e., stars) and maximum degree at most Δ, where $\Delta \geq 3$, using a labeling scheme of length $\Theta(\log\log\Delta)$, is $\Theta(\frac{\log\Delta}{(\log\log\Delta)})$.*

5.3 Maximum degree $\Delta = 2$

We finally address the case of trees of maximum degree $\Delta = 2$, i.e., the class of lines. Since there is exactly one line of a given diameter D, the problem of topology recognition for $\Delta = 2$ and for a given diameter D is trivial. A meaningful variation of the problem for $\Delta = 2$ is to consider all trees (lines) of diameter *at most* D, for a given D.

We first propose a topology recognition algorithm for all lines of diameter at most D, where $D \geq 4$, using a labeling scheme of length $O(1)$ and working in time $O(\log D)$.

Algorithm `Line-Topology-Recognition`

Let T be a tree of maximum degree 2 and diameter at most D, i.e., a line of length at most D. Let $v_1, v_2, \ldots, v_{k+1}$, for $k \leq D$, be the nodes of T, where v_1 and v_{k+1} are the two endpoints. At a high level, we partition the line into segments of length $O(\log k)$ and assign labels, containing (among other terms) couples of bits, to the nodes in each segment. This is done in such a way that the concatenation of the first bits of the couples in a segment is the binary representation of the integer k, and the concatenation of the second bits of the couples in a segment is the binary representation of the segment number. In time $O(\log k)$, every node learns the labels in each segment, and computes k and the number $j \geq 0$ of the segment to which it belongs. It identifies its position in this segment from the round number in which it receives a message for the first time. Then a node outputs the line of length k with its position in it.

The following lemma gives a lower bound on the time of topology recognition for lines, matching the performance of Algorithm `Line-Topology-Recognition`.

Lemma 5. *Let $D \geq 3$ be any integer, and let $c > 0$ be any real constant. For any line T, consider a labeling scheme LABEL(T) of length at most c. Let $TOPO$ be any* *algorithm that solves topology recognition for every line of diameter at most D using the labeling scheme LABEL(T). Then there exists a line of diameter at most D, for which $TOPO$ must take time $\Omega(\log D)$.*

In view of the performance of Algorithm `Line-Topology-Recognition` and of Lemma 5, we have the following result.

Theorem 6. *The optimal time for topology recognition in the class of trees of maximum degree $\Delta = 2$ (i.e., lines) of diameter at most D, using a labeling scheme of length $O(1)$, is $\Theta(\log \boldsymbol{D})$.*

6 Conclusion

We established a tight bound $\Theta(\log \log \Delta)$ on the minimum length of labeling schemes permitting topology recognition in trees of maximum degree Δ, and we proved upper and lower bounds on topology recognition time, using such short schemes. These bounds on time are almost tight: they leave a multiplicative gap smaller than any polynomial in Δ. Closing this small gap is a natural open problem. Another interesting research topic is to extend our results to the class of arbitrary graphs. We conjecture that such results, both concerning the minimum length of labeling schemes permitting topology recognition, and concerning the time necessary for this task, may be quite different from those that hold for trees.

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. M. Chrobak, L. Gasieniec, W. Rytter, Fast broadcasting and gossiping in radio networks, Journal of Algorithms 43 (2002):177189.
3. R. Cohen, P. Fraigniaud, D. Ilcinkas, A. Korman, D. Peleg, Label-guided graph exploration by a finite automaton, ACM Transactions on Algorithms 4 (2008).
4. D. Dereniowski, A. Pelc, Drawing maps with advice, Journal of Parallel and Distributed Computing 72 (2012), 132–143.
5. Y. Emek, P. Fraigniaud, A. Korman, A. Rosen, Online computation with advice, Theoretical Computer Science 412 (2011), 2642–2656.
6. P. Fraigniaud, C. Gavoille, D. Ilcinkas, A. Pelc, Distributed computing with advice: Information sensitivity of graph coloring, Distributed Computing 21 (2009), 395–403.
7. P. Fraigniaud, D. Ilcinkas, A. Pelc, Communication algorithms with advice, Journal of Computer and System Sciences 76 (2010), 222–232.
8. P. Fraigniaud, D. Ilcinkas, A. Pelc, Tree exploration with advice, Information and Computation 206 (2008), 1276–1287.
9. P. Fraigniaud, A. Korman, E. Lebhar, Local MST computation with short advice, Theory of Computing Systems 47 (2010), 920–933.
10. E. Fusco, A. Pelc, Trade-offs between the size of advice and broadcasting time in trees, Algorithmica 60 (2011), 719–734.

11. E. Fusco, A. Pelc, R. Petreschi, Topology recognition with advice, Information and Computation 247 (2016), 254-265.
12. L. Gasieniec, A. Pagourtzis, I. Potapov, T. Radzik, Deterministic communication in radio networks with large labels. Algorithmica 47 (2007), 97-117.
13. L. Gasieniec, D. Peleg, Q. Xin, Faster communication in known topology radio networks, Distributed Computing 19 (2007), 289-300.
14. C. Gavoille, D. Peleg, S. Pérennes, R. Raz. Distance labeling in graphs, Journal of Algorithms 53 (2004), 85-112.
15. 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.
16. D. Ilcinkas, D. Kowalski, A. Pelc, Fast radio broadcasting with advice, Theoretical Computer Science, 411 (2012), 1544–1557.
17. M. Katz, N. Katz, A. Korman, D. Peleg, Labeling schemes for flow and connectivity, SIAM Journal of Computing 34 (2004), 23–40.
18. A. Korman, S. Kutten, D. Peleg, Proof labeling schemes, Distributed Computing 22 (2010), 215–233.
19. 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.
20. N. Nisse, D. Soguet, Graph searching with advice, Theoretical Computer Science 410 (2009), 1307–1318.
21. D. Peleg, Distributed computing, a locality-sensitive approach, SIAM Monographs on Discrete Mathematics and Applications, Philadelphia 2000.