Deterministic Size Discovery and Topology Recognition in Radio Networks with Short Labels

Adam Gańczorz
Institute of Computer Science, University of Wrocław, Poland

Tomasz Jurdziński
Institute of Computer Science, University of Wrocław, Poland

Mateusz Lewko
Institute of Computer Science, University of Wrocław, Poland

Andrzej Pelc
Département d'informatique, University of Québec en Outaouais, Gatineau, Canada

Abstract

We consider the fundamental problems of *size discovery* and *topology recognition* in radio networks modeled by simple undirected connected graphs. Size discovery calls for all nodes to output the number of nodes in the graph, called its size, and in the task of topology recognition each node has to learn the topology of the graph and its position in it.

We do not assume collision detection: in case of a collision, node v does not hear anything (except the background noise that it also hears when no neighbor transmits). The time of a deterministic algorithm for each of the above problems is the worst-case number of rounds it takes to solve it. Nodes have labels which are (not necessarily different) binary strings. Each node knows its own label and can use it when executing the algorithm. The length of a labeling scheme is the largest length of a label.

For size discovery, we construct a labeling scheme of length $O(\log\log\Delta)$ (which is known to be optimal, even if collision detection is available) and we design an algorithm for this problem using this scheme and working in time $O(\log^2 n)$, where n is the size of the graph. We also show that time complexity $O(\log^2 n)$ is optimal for the problem of size discovery, whenever the labeling scheme is of optimal length $O(\log\log\Delta)$. For topology recognition, we construct a labeling scheme of length $O(\log\Delta)$, and we design an algorithm for this problem using this scheme and working in time $O\left(D\Delta + \min(\Delta^2, n)\right)$, where D is the diameter of the graph. We also show that the length of our labeling scheme is asymptotically optimal.

2012 ACM Subject Classification Theory of computation → Distributed algorithms

Keywords and phrases size discovery, topology recognition, radio network, labeling scheme

Digital Object Identifier 10.4230/LIPIcs.DISC.2021.22

Related Version Due to the length requirements some proofs were omitted. For details see:
Full Version: `https://arxiv.org/abs/2105.10595` [13]

Funding *Adam Gańczorz*: Supported by the National Science Centre, Poland, grant 2017/25/B/ST6/02010.
Tomasz Jurdziński: Supported by the National Science Centre, Poland, grant 2017/25/B/ST6/02010.
Mateusz Lewko: Supported by the National Science Centre, Poland, grant 2017/25/B/ST6/02010.
Andrzej Pelc: Partially supported by NSERC discovery grant 2018-03899 and by the Research Chair in Distributed Computing at the Université du Québec en Outaouais.

1 Introduction

Information about the topology of the network or some of its parameters, such as size, often determines the efficiency and sometimes the feasibility of many network algorithms. For example, graph exploration with stop performed in rings with non-unique labels is impossible without knowing some upper bound on the size of the ring. On the other hand, optimal broadcasting algorithms in wireless networks with distinct labels are faster when the topology of the network is known [21]. Hence, the problems of *size discovery* and *topology recognition* are fundamental in network computing. Size discovery calls for all nodes to output the number of nodes in the underlying graph, called its size, and in the task of topology recognition each node has to output an isomorphic copy of the graph with its position in it marked. More formally, in topology recognition, every node v of the graph G modeling the network must output a graph G' and a node v' in this graph, such that there exists an isomorphism $f : G \to G'$, for which $f(v) = v'$.

The model. We consider size discovery and topology recognition in radio networks modeled by simple undirected graphs. Throughout this paper $G = (V, E)$ denotes the graph modeling the network, n denotes the number of its nodes, D its diameter, and Δ its maximum degree. We use square brackets to indicate sets of consecutive integers: $[i, j] = \{i, \ldots, j\}$ and $[i] = [1, i]$.

As usually assumed in the algorithmic literature on radio networks, nodes communicate in synchronous rounds, starting in the same round. In each round a node can either transmit the same 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 more than one neighbor of a node v transmits in a given round, there is a *collision* at v. Two scenarios concerning collisions were considered in the literature. The availability of *collision detection* means that node v can distinguish collision from silence which occurs when no neighbor transmits. If collision detection is not available, node v does not hear anything in case of a collision (except the background noise that it also hears when no neighbor transmits). In this paper we do not assume collision detection. The time of a deterministic algorithm for each of the above problems is the worst-case number of rounds it takes to solve it.

If nodes are anonymous then neither size discovery nor topology recognition can be performed, as no communication in the network is possible. Indeed, without any labels, in every round either all nodes transmit or all remain silent, and so no message can be received. Hence we consider labeled networks. A *labeling scheme* for a given network represented by a graph $G = (V, E)$ 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. Every node knows a priori only its label, and can use it as a parameter for the size discovery or topology recognition algorithm.

Our goal is to construct short labeling schemes for size discovery and topology recognition in arbitrary radio networks, and to design efficient deterministic algorithms for each of these tasks, using such schemes. Such short schemes in the context of radio networks were studied for size discovery in [19], and for topology recognition in [18]. In [19] the authors worked in the model with collision detection. They constructed labeling schemes of length $O(\log \log \Delta)$ and a size discovery algorithm using this scheme and working in time $O(Dn^2 \log \Delta)$. They also proved that labels of size $\Omega(\log \log \Delta)$ are necessary to solve the size discovery problem in this model. In [18], the authors studied topology recognition without collision detection, similarly as we do in the present paper, but restricted attention only to tree networks. They constructed labeling schemes of length $O(\log\log\Delta)$ and a topology recognition algorithm working for arbitrary trees, using these schemes. Moreover, they showed that labels of size $\Omega(\log\log\Delta)$ are necessary to solve the topology recognition problem for trees.

Solving distributed network problems with short labels can be seen in the framework of algorithms with *advice*. In this paradigm that has recently got growing attention, an oracle knowing the network gives advice to nodes not knowing it, in the form of binary strings, and a distributed algorithm cooperating with the oracle uses this advice to solve the problem efficiently. The required size of advice (maximum length of the strings) can be considered as a measure of the difficulty of the problem. Two variations are studied in the literature: either the binary string given to nodes is the same for all of them [17] or different strings may be given to different nodes [8, 7, 11, 12], as in the case of the present paper. If strings may be different, they can be considered as labels assigned to nodes by a labeling scheme. Such labeling schemes permitting to solve a given network task efficiently are also called *informative labeling schemes.* One of the famous examples of using informative labeling schemes is to answer adjacency queries in graphs [2].

Several authors have studied the minimum amount of advice (i.e., label length) required to solve certain network problems (see the subsection Related work). The framework of advice or labeling schemes permits us to quantify the amount of information used to solve a network problem, such as size discovery or topology recognition, regardless of the type of information that is provided. It should be noticed that the scenario of the same advice (label) given to all nodes would be trivial in the case of radio networks: no communication could occur, and hence the advice would have to contain the size of the network for size discovery, and would not help for topology recognition, as nodes would not be able to find their position in the network without communicating.

Our results. It turns out that the optimal length of labeling schemes, both for size discovery and for topology recognition, depends on the maximum degree Δ of the graph. For size discovery, we construct a labeling scheme of length $O(\log\log\Delta)$, which is optimal, in view of [19], and we design an algorithm for this problem using this scheme and working in time $O(\log^2 n)$, where n is the size of the graph. We also show that time complexity $O(\log^2 n)$ is optimal for the problem of size discovery, whenever the labeling scheme is of optimal length $O(\log\log\Delta)$. Hence, without collision detection we achieve the same optimal length of the labeling scheme, as was done in [19] with collision detection, and for this optimal scheme our size discovery algorithm is exponentially faster than that in [19].

For topology recognition, we construct a labeling scheme of length $O(\log\Delta)$, and we design an algorithm for this problem using this scheme and working in time $O\left(D\Delta + \min(\Delta^2, n)\right)$, where D is the diameter of the graph. We also show that the length of our labeling scheme is asymptotically optimal, by proving that topology recognition in the class of arbitrary radio networks requires labeling schemes of length $\Omega(\log\Delta)$. (In fact we prove a stronger result that this lower bound holds even in the model with collision detection.) If the optimal length of a labeling scheme sufficient to solve a problem is considered a measure of the difficulty of the problem, our result shows, in view of the labeling scheme of length $O(\log\log\Delta)$ for topology recognition in trees [18], that this task is exponentially more difficult in arbitrary radio networks than in radio networks modeled by trees.

Related work. There is a vast literature concerning distributed algorithms for various tasks in radio networks. These tasks include, e.g., broadcasting [5, 15], gossiping [5, 14] and leader election [6, 22]. In some cases [5, 14], the authors use the model without collision detection, in others [16, 22], the collision detection capability is assumed.

Many authors use the framework of algorithms with advice (or equivalently informative labeling schemes) to investigate the amount of information needed to solve a given network problem. In [10], the authors compare the minimum size of advice required to solve two information dissemination problems, using a linear number of messages. In [11], it is shown that advice of constant size permits to carry out the distributed construction of a minimum spanning tree in logarithmic time. In [2], optimal labeling schemes are constructed in order to answer adjacency queries in graphs. In [9], the advice paradigm is used to solve online problems.

In the model of radio networks, apart from the previously mentioned papers [19] studying size discovery and [18] studying topology recognition in the framework of short labeling schemes, other authors studied the tasks of broadcast and multi-broadcast [20, 8, 7, 23], and convergecast [4] in this framework. While [8, 7] assume that short labels are given to anonymous nodes, [20] adopts a different approach. The authors study radio networks without collision detection for which it is possible to perform centralized broadcasting in constant time, i.e., when the topology of the network is known and all nodes have different labels. They investigate how many bits of additional information (i.e., not counting the labels of nodes) given to nodes are sufficient for performing broadcast in constant time in such networks, if the topology of the network is not known to the nodes.

The task of topology recognition was also investigated in models other than the radio model: in [12] the authors use the LOCAL model, and in [24] the model used is the congested clique.

2 Preliminaries

As our algorithms use recent results showing that there exist constant-length labeling schemes for the broadcast problem, we recall these results, adjust them to our needs and introduce an auxiliary notion called the *broadcast tree*. Then, using the existence of an efficient broadcast algorithm with constant-length labels, we give a lemma allowing to "encode" a given message M in labels of the neighborhood of some path. This path is associated with an efficient algorithm that collects the whole message in a single node and broadcasts it to the whole network.

We use the constant-length labeling schemes for the broadcast problem [8, 7]. In this problem, a source node s has a message which must be communicated to all other nodes. First, we recall the result regarding fast broadcast with constant-length labels from [7].

▶ **Theorem 1** ([7]). *There exists a labeling scheme of length $O(1)$ and an algorithm* Executor *using it which solves the broadcast problem in time $O(D \log n + \log^2 n)$.*

We say that a node v is *informed* in some round t of a broadcast algorithm if v knows the broadcast message in round t. Otherwise, v is *uninformed* in round t. We assume that feedback messages in the execution of Executor are distinct from the broadcast message (this can be easily ensured by adding a special sign to the broadcast message).

▶ **Definition 2** (Broadcast Tree). *Let $G = (V, E)$ be a graph with the source node $s \in V$.*

For each node $v \in V \setminus \{s\}$, the parent of v, denoted $parent(v)$, is the first node which successfully transmits the broadcast message to v during the execution of Executor *in G. The* broadcast tree *$T_{\text{Executor}(G)}$ is the tree with the root s and the set of edges $(v, parent(v))$ for each $v \neq s$.*

The level *of a node $v \in V$, denoted level(v), in the broadcast tree $T_{\text{Executor}(G)}$ is equal to the natural number i such that v receives the broadcast message from parent(v) for the first time in round i of the execution of* Executor *on G.*

In our algorithms for the size discovery problem, we will use the following properties of the algorithm Executor from [7].

▶ **Lemma 3.** *The broadcast algorithm* Executor *described in Theorem 1 satisfies the following properties:*

(1) *Assume that the level of a node v in the broadcast tree $T_{\text{Executor}(G)}$ is equal to i and the level of parent(v) in this tree is $j < i$. Then, parent(v) has a child with level k for each $k \in [j+1, i]$ such that $k \mod 3 = 1$.*

(2) *The maximum value of the levels of nodes of the broadcast tree $T_{\text{Executor}(G)}$ is larger than $t - 3$, where t is the number of rounds of the execution of* Executor.

(3) *Each node v can determine the level of parent(v) and its own level in the broadcast tree $T_{\text{Executor}}(G)$.*

We say that an algorithm A solves the *acknowledged broadcast* problem in T rounds iff it solves the broadcast problem, and moreover, all nodes of the graph know after T rounds that broadcast has been completed. Below, we show that Executor can be easily transformed into an acknowledged broadcast algorithm.

▶ **Corollary 4.** *Algorithm* Executor *can be transformed into algorithm* ExecAck *such that*

- ExecAck *solves the acknowledged broadcast problem on every graph G, using $O(1)$-bit labels and working in at most $3t$ rounds, where $t = O(D \log n + \log^2 n)$ is the number of rounds of the execution of* Executor *on G.*
- *after an execution of* ExecAck *on G, each node knows the number of levels of the broadcast tree $T_{\text{Executor}(G)}$, as well as its own level and the level of its parent in $T_{\text{Executor}(G)}$.*

Now, we make an observation regarding the length of labels sufficient to store (in a distributed way) an arbitrary message M initially unknown to all nodes, and subsequently make the message M known to all nodes.

▶ **Lemma 5.** *Let* ExecAck *be the acknowledged broadcast algorithm which satisfies Corollary 4. Then, an arbitrary message M of size m (initially unknown to nodes) can be made known to all nodes of graph G in $O(t)$ rounds using $O(1 + m/t)$-bit labels, where $t = O(D \log n + \log^2 n)$ is the number of rounds of the execution of* ExecAck *on G.*

3 Optimal Labeling Schemes

In this section we construct labeling schemes of optimal length for the size discovery and for the topology recognition problems, together with algorithms using these optimal schemes for these tasks. For the task of size discovery we will later show, in Section 5, a faster algorithm, in fact the fastest possible algorithm to solve the size discovery problem using a labeling scheme of optimal length.

3.1 Size discovery

In Sections 3.1.1–3.1.3 we design a general labeling scheme of length $O(\log \log \Delta)$ and a size discovery algorithm using this scheme, based on the broadcast algorithm working with constant-length labels [7].

3.1.1 $\log \Delta$-degree subtrees of a tree

We prove a general lemma stating that, given a tree T with maximum degree Δ and 3 bits of "local memory" at each node, and given a binary string M of length $\lceil \log(n+1) \rceil + 2$, there exists a subtree T' of T with maximum degree $\lfloor \log \Delta \rfloor + 1$, such that M can be split into the 3-bit local memories of the nodes of T'.

▶ **Lemma 6.** *Let $T = (V, E)$ be a tree with maximum degree Δ, size n and the root $r \in V$. Let M be a binary string of length $\lceil \log(n+1) \rceil + 1$. Then, there exists a subtree T' of T rooted at r and an assignment of binary strings to nodes of T' such that:*

(a) *the number of children of each node of T' is at most $\lfloor \log \Delta \rfloor + 1$,*

(b) *the string assigned to each node of T' has length at most 3,*

(c) *the string assigned to the root of T' has length 2,*

and the concatenation of all strings assigned to nodes of T' is M.

3.1.2 Labeling scheme

In this section we describe the labeling scheme CompactLabels which combines the so-called Δ-learning primitive from [19], the construction of a broadcast tree with help of the algorithm Executor from Theorem 1, and the limited-degree subtree described in Lemma 6.

The labels will consist of the root bit and three disjoint blocks: Δ-*block*, *BroadcastTree-block* and *Message-block*.

Δ-block and the root bit. First, we choose a node r with the largest degree Δ in the graph G, and mark r using the one-bit flag root. That is, $\text{ROOT}_r = 1$ and $\text{ROOT}_v = 0$ for each $v \neq r$.

Then, we describe $O(\log \log \Delta)$-bit strings called Δ-blocks which will be used to learn the value of Δ by the node r. The root r is assigned the pair $(x, 0)$, where x is the binary representation of the integer $\lfloor \log \Delta \rfloor + 1$. This integer is the size of the binary encoding of Δ. Then we choose $\lfloor \log \Delta \rfloor + 1$ neighbors of r and assign them consecutive natural numbers in the range $[1, \lfloor \log \Delta \rfloor + 1]$. The Δ-block of the ith chosen node is equal to the pair (a_i, b_i) where a_i is the binary representation of $i \in [\lfloor \log \Delta \rfloor + 1]$ of length $\lfloor \log(\lfloor \log \Delta \rfloor + 1) \rfloor + 1$, with leading zeros, and b_i is the ith bit of the binary representation of Δ. The Δ-blocks of other nodes are equal to the tuple $(0, 0)$. As the value of $\log \Delta$ can be encoded on $O(\log \log \Delta)$ bits, the length of Δ-blocks is $O(\log \log \Delta)$.

BroadcastTree-block. We apply Corollary 4 to construct the broadcast tree T_{EXECUTOR} of the graph with root (source vertex) r and assign $O(1)$-bit labels to nodes, used by the acknowledged broadcast algorithm ExecAck working in $O(D \log n + \log^2 n)$ rounds (Corollary 4).

Message-block. Let M be the binary representation of the size n of the graph. The message-block is the concatenation of two blocks: index and message.

We apply Lemma 6 in order to construct a subtree T' of T_{EXECUTOR} such that the degree of T' is not larger than $\lfloor \log \Delta \rfloor + 1$ and each node of T' is assigned a substring of M of length at most 3. To each node v of T' apart from the root we assign a natural number k_v such that the numbers assigned to the children of any node are consecutive integers starting from 0. For a node v of T', the block index of fixed length $O(\log \log \Delta)$ is the binary representation of the integer $k_v \in [1, \lfloor \log \Delta \rfloor] + 1$ (with leading zeros), where v is chosen as the child with number k_v of its parent. The substring of length at most 3 assigned to each node of T' according to Lemma 6 is the block message of this node. The concatenation of these substrings of

M forms the string M and substrings are assigned to nodes of T' in post-order, i.e., the substring assigned to a given node v is situated in M after the substrings assigned to the children of v in T', and substrings are assigned to the children w in increasing order of k_w. The blocks INDEX and MESSAGE of nodes outside of tree T' are the string (0).

Conceptually, the label of a node is the concatenation of the root bit, the Δ-block, the Broadcast Tree-block and the Message-block. In order to mark separations between the different blocks, we use the standard trick: the bit 1 is coded as 10, the bit 0 is coded as 01 and separations are coded as 00. This does not change the complexity of the label length, hence our labeling scheme has length $O(\log\log\Delta)$.

3.1.3 A simple size discovery algorithm

In this section we describe a simple size discovery algorithm using the above labeling scheme and working in time $O(D\log n + \log^2 n)$. We will show in Section 5 how to improve this algorithm to get the optimal time $O(\log^2 n)$.

We first design the size discovery algorithm AuxiliarySD, which uses the labeling scheme described in the previous section. The algorithm AuxiliarySD is a composition of three procedures, `Δ-learning`, `Ack-broadcast`, and `Size-learning`, corresponding to the three blocks of the labels described above: *Δ-block*, *BroadcastTree-block* and *Message-block*.
Procedure `Δ-learning`.
This procedure lasts $\lfloor\log\Delta\rfloor + 1$ rounds. In the ith round, the node with Δ-block (a_i, b_i) for $i > 0$ transmits the message with two bits: 0 and b_i. In each round all nodes except the root r (i.e., the node with the ROOT bit equal to 1) ignore messages with the value of the first bit equal to 0 and r stores consecutive bits b_i. The root r also knows the value of $\lfloor\log\Delta\rfloor + 1$ stored in its label, so it will know the number of rounds of the Δ-Learning procedure.
Procedure `Ack-broadcast`
We execute algorithm ExecAck from Corollary 4 with the source vertex r, the labels from the BroadcastTree-block and the broadcast message equal to the binary string encoding the value of Δ. As we use the algorithm solving the acknowledged broadcast problem, all nodes know the number of the last round of this procedure.
Procedure `Size-learning`
The goal of this procedure is to learn the binary representation M of the size n of the graph that is distributedly stored in nodes of the subtree T' of the broadcast tree T_{Executor}, as described in the previous section.

Let L be the number of levels of the broadcast tree T_{Executor}. The procedure `Size-learning` is split into phases $1, \ldots, L$ such that the strings MESSAGE stored in nodes from the level $L - i + 1$ are transmitted to their parents in phase i, together with messages containing strings MESSAGE received by those nodes from their subtrees. Each phase lasts $\lfloor\log\Delta\rfloor + 1$ rounds. At the beginning of the phase i, each node v of T' of level $L - i + 1$ reconstructs its message from all messages received in previous rounds and its own string MESSAGE. As the labeling scheme uses post-order encoding, the node v first concatenates the messages of its children in T' in the order of numbers assigned to them and then adds its own string MESSAGE at the end of the string to get M_v, the part of M stored in its subtree. Assume that the node v of level $L - i + 1$ has INDEX that is the binary representation of the integer $k_v \in [1, \lfloor\log\Delta\rfloor + 1]$. Then, v transmits M_v in the round k_v of phase i.

▶ **Lemma 7.** *The algorithm AuxiliarySD solves the size discovery problem using a labeling scheme of length $O(\log\log\Delta)$, and it works in time $O(T_{\text{Executor}}(n, D) \cdot \log\Delta)$, where $T_{\text{Executor}}(n, D) = O(D\log n + \log^2 n)$ is the time of the broadcast algorithm Executor from Theorem 1.*

Now, we combine Lemma 7 with Lemma 5 to construct an improved version of AuxiliarySD, called GeneralSD. The goal is to get rid of the additional $\log \Delta$ multiplicative factor in the time complexity of a size discovery algorithm based on the broadcast algorithm Executor.

Given a graph G, we first compute the number t_G of rounds of the execution of Executor on G. If $t_G < \log n$, then we use the labeling scheme and the algorithm from Lemma 7. As the number of levels of the tree T_{Executor} is not larger than the number of rounds of the algorithm, the number of rounds of the size discovery algorithm is $O(t_G \log \Delta) = O(\log^2 n)$.

If $t_G \geq \log n$, we use the constant-length labeling scheme and the broadcasting algorithm ExecAck from Lemma 5, with the binary representation of the size n of the graph as the message M. Thus, we obtain an algorithm which solves the size discovery problem in time $O(D \log n + \log^2 n)$ using a labeling scheme of length $O\left(1 + \frac{\log n}{t_G}\right) = O(1)$.

Finally, in order to make the nodes of the graph aware of the chosen variant of the algorithm, we add one bit to all labels which contains the information whether or not $t_G < \log n$. Given the value of this bit, the nodes work according to instructions of the former or the latter algorithm described above. Thus, we obtain the following result.

▶ **Theorem 8.** *The algorithm GeneralSD solves the size discovery problem using a labeling scheme of length $O(\log \log \Delta)$, and works in time $O(D \log n + \log^2 n)$.*

3.2 Topology recognition

In this section we design a topology recognition algorithm working for any graph of maximum degree Δ using a labeling scheme of length $O(\log \Delta)$. We will later prove that this length is optimal. Our algorithm works in time $O\left(D\Delta + \min(n, \Delta^2)\right)$. First, in Section 3.2.1 we focus on constructing a BFS tree of the graph and efficient broadcast/gathering algorithms working in this tree. Then, in Section 3.2.2, the main topology recognition algorithm is presented, using the broadcast and gathering subroutines from Section 3.2.1.

3.2.1 Broadcast-gathering primitive

In this section we describe a labeling scheme of length $O(\log \Delta)$ and algorithms for the (acknowledged) *broadcast* and *gathering* problems using this scheme. Both algorithms work in time $O(D\Delta)$. We would like to emphasize that, although the broadcast problem can be solved more efficiently (cf. [7]) than the solution given here, our aim is to build the broadcast schedule associated with the gathering algorithm. Notice that the gathering problem requires a labeling scheme of lengths $\Omega(\log \Delta)$, e.g., if the communication graph's topology is a star.

Consider a graph $G = (V, E)$ of maximum degree Δ and diameter D, with a root node $r \in V$. In the gathering problem, each node $v \in V$ has some message M_v and the root r has to learn all messages M_v.

Denote by *layer*(v) the distance from r to a node v. We say that v is at layer i if *layer*$(v) = i$. Let V_i be the set of nodes at layer i. The neighborhood of a node v in G is denoted by $\mathcal{N}(v)$ and the set of neighbors of v at layer i is denoted by $\mathcal{N}_i(v)$. We fix an arbitrary strict total ordering on nodes denoted by $\prec$. For each node v, its label $\mathcal{L}(v)$ is equal to the tuple $(r_v, \mathit{leaf}(v), a_v, b_v, g_v, \Delta)$, where r_v is the 1-bit flag indicating whether v is the root node r, *leaf*(v) is the bit indicating whether v has any neighbors on higher layers than *layer*(v), a_v is the bit used for acknowledged broadcast, while both b_v and g_v, called *broadcast-label* and *gather-label*, are special integer values used for broadcast and gathering respectively, described in more detail below. More precisely, the label $\mathcal{L}(v)$ is the concatenation of bits r_v, *leaf*(v), a_v, and binary representations of integers b_v, g_v and Δ.

The idea of the broadcast/gather labels and algorithms. For the broadcast problem, we split the nodes of each layer i into subsets $X_0, \ldots, X_{\Delta-1}$, where X_j is a maximal set of nodes of layer i not belonging to $\bigcup_{k<j} X_k$ such that the sets of neighbors of X_j at layer $i+1$ are pairwise disjoint, exluding the neighbors of nodes from $\bigcup_{k<j} X_k$. For $v \in X_j$ we set $b_v = j$. We observe that all nodes from X_j can transmit in the same round in order to deliver the broadcast message to all their neighbors at layer $i+1$, excluding the neighbors of the nodes from $\bigcup_{k<j} X_k$. This fact makes possible to deliver the broadcast message from layer i to layer $i+1$ in Δ rounds. For each node v at layer $i+1$, the first node which successfully transmits the broadcast message to v becomes $\mathit{parent}(v)$ and v becomes its child.

For the gathering problem, we assign distinct gather-labels in the range $[0, \Delta-1]$ to the children of each node. This assignment is required to satisfy also the following additional restriction. If u at layer $i+1$ is a neighbor of v at layer i and u is not a child of v, then the gather-label of u is not assigned to any child of v. With this restriction, the nodes at level i with a given value of the gather-label can transmit successfully messages to their parents in the same round. Thanks to this property, the values/messages stored in the nodes from the layer $i > 0$ can be gathered in their parents at layer $i-1$ in $O(\Delta)$ rounds.

More details, formal statement of the above mentioned properties and the final broadcast and gathering algorithms are described in the appendix, Section A.1.

3.2.2 Topology recognition algorithm

In order to solve the topology recognition problem, we choose an arbitrary node as the root r and assign to each node v the label which is the concatenation of two binary strings: the label $\mathcal{L}(v)$ used by our broadcast-gathering primitive and the binary representation of a natural number $\mathcal{C}(v)$ called the *color* of v which is the color of v in some fixed distance-two vertex coloring of the graph G with the set of colors equal to $[1, \Delta^2]$. Given these labels, we execute the following four-stage algorithm TopRec:

Stage 1. First, the acknowledged broadcast AckBrBFS is executed with the modification that each node v transmits the concatenation of the consecutive substrings g_u of the labels of nodes of the path from the root r to v, including g_v. This concatenation will be denoted by $\mathrm{ID}(v)$.

Stage 2. Then, in the block of Δ^2 rounds, each node v such that $\mathcal{C}(v) = i \in [1, \Delta^2]$ transmits in the round i of the block, sending the string $\mathrm{ID}(v)$, and each node stores received messages in its local memory.

▶ Remark. If $\Delta^2 > n$ then $\log \Delta \geq \frac{1}{2} \log n$. In this case, in order to reduce the number of rounds of Stage 2 to $\min(\Delta^2, n)$, we extend the labeling by assigning to each node a unique identifier in the range $[1, n]$ of length $O(\log n) = O(\log \Delta)$. Then, Stage 2 consists of n rounds, where the node with the identifier i transmits in round i of the stage.

Stage 3. Next, we execute the gathering algorithm GatherBFS, where the message M_v of a node v is equal to the set of IDs received by v in Stage 2, together with its own ID. The set of these messages, over all nodes v, permits each node to reconstruct the topology of the graph and situate itself in it.

Stage 4. Finally, we execute the broadcast algorithm BroadcastBFS, where the message M is equal to the set of messages of all nodes gathered at the root r in Stage 3.

▶ **Theorem 9.** *The algorithm TopRec solves the topology recognition problem on every graph, using a labeling scheme of length $O(\log \Delta)$. It works in time $O(D\Delta + \min(n, \Delta^2))$.*

4 Lower Bound on Lengths of Labels for Topology Recognition

In this section, we show that any algorithm for topology recognition in the class of general graphs requires a labeling scheme of length $\Omega(\log \Delta)$, and hence our algorithm TopRec from the previous section uses a labeling scheme of optimal length. The optimal length of a labeling scheme for topology recognition is thus exponentially larger than that sufficient to solve the easier problem of size discovery. In fact, we will prove a stronger result: the above lower bound holds even in the model with collision detection. Hence, till the end of this section we assume that collision detection is available.

The proof is divided into two parts. We first show the lower bound $\Omega(\log n)$ on the length of labeling schemes for topology recognition in graphs of degree $2^{\Omega(\log n)}$. Second, we generalize this result for graphs of arbitrary degree Δ, to get the lower bound $\Omega(\log \Delta)$.

In our proof, we exploit the fact that each node in the graph G has to learn its degree. Intuitively, if the set $\{ deg(v) \mid v \in V \}$ of node degrees in G is larger than the set $\{ \mathcal{L}(v) \mid v \in V \}$ of node labels then some two nodes $u, v \in V$, such that $deg(u) \neq deg(v) \wedge \mathcal{L}(u) = \mathcal{L}(v)$ have to learn their degrees through their *communication histories* $\mathcal{H}$. We will construct a family of graphs such that the labels have to be of length $\Omega(\log \Delta)$ so that nodes with the same labels and different degrees could have different communication histories.

We now formally define a *communication history* of a node for a given topology recognition algorithm executed on a given graph. Intuitively, this is a record of what the node learns during the execution of the algorithm (assuming collision detection).

▶ **Definition 10** (Communication history). *Let $G = (V, E)$ be a graph and let $\mathcal{L}$ denote a labeling scheme on G. Consider a topology recognition algorithm $\mathcal{A}$ executed on G and let i be a round of this algorithm. For each node $v \in V$, we define $\mathcal{H}_i(v)$ inductively such that $\mathcal{H}_i(v)$ denotes the communication history of v at the end of the ith round of $\mathcal{A}$.*

- *$\mathcal{H}_0(v) = \emptyset$, for all v.*
- *If*
  - *v transmits in the ith round, or*
  - *v listens and more than one neighbor of v transmits in the ith round (a collission),*

  we append the special character #, denoting "no message", to v's history, i.e., $\mathcal{H}_i(v) = [\mathcal{H}_{i-1}(v), \#]$,
- *If v listens and none of its neighbors transmits in the ith round (silence), we append ε to v's history, i.e., $\mathcal{H}_i(v) = [\mathcal{H}_{i-1}(v), \varepsilon]$.*
- *If v listens and successfully receives a message $m \in \{0,1\}^*$ in the ith round (exactly one neighbor of v transmits m), we append m to v's history, i.e., $\mathcal{H}_i(v) = [\mathcal{H}_{i-1}(v), m]$.*

$\mathcal{H}(v)$ denotes the communication history of v at the end of the last round of the execution of $\mathcal{A}$ on G.

4.1 The Lower Bound for Graphs with Large Degrees

In this section we define a family $\mathcal{G}$ of graphs of arbitrarily large size n with maximum degree $\Delta = 2^{\Omega(\log n)}$, such that a labeling scheme of length $\Omega(\log n)$ is necessary to solve the topology recognition problem on each suffciently large graph from this family.

Let n be a sufficiently large natural number such that $n^{1/2}$ is an even natural number. We construct an n-node graph $G_n = (V, E)$. The graph is composed of $n^{1/2}$ components, each component is composed of $n^{1/2}$ nodes. Let C_i denote the set of nodes from the ith component. Every node is connected with every node from a different component, i.e., for every C_i, C_j $(i \neq j)$ and every pair of nodes $u \in C_i, v \in C_j$ we have $(u, v) \in E$. Let $C(v)$ denote the component to which a node v belongs.

We now describe the set of edges connecting nodes of a component C_i composed of $k = n^{1/2}$ nodes. We divide nodes of C_i into two sets A and B, each of size $k/2$. Let $a_j \in A$ denote the jth node from A, and let $b_j \in B$ denote the jth node from B. Then, we connect a_j with the first j nodes from B: $(a_j, b_1), (a_j, b_2), \ldots, (a_j, b_j) \in E$. This concludes the construction of G_n. Observe that each G_n is connected. Note also that for any component, the set of different degrees of nodes in this component has size $k/2$. The degree of each node is in the range $[n - k + 1, n - k/2]$, since each component has size $n^{1/2}$. Note that nodes of different degrees that have equal labels and equal communication histories cannot learn that they have different degrees. The family $\mathcal{G}$ consists of all graphs G_n, such that $n^{1/2}$ is an even natural number.

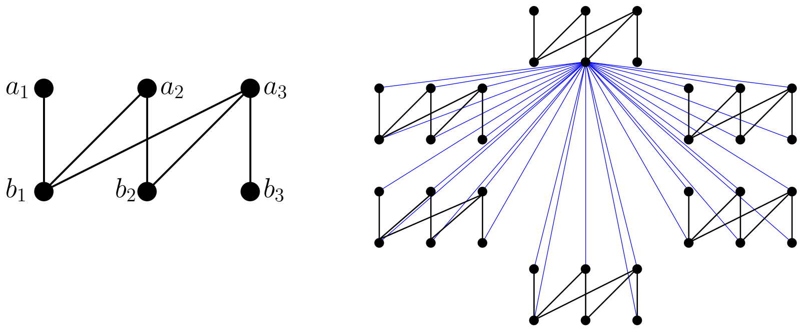

Figure 1 An illustration of the graph G_{36} from the family $\mathcal{G}$. The left picture depicts one component, the right picture depicts the whole graph. Blue edges connect nodes from different components. For clarity, only some of the inter-component edges are presented (each pair of nodes from different components should be connected by an edge).

Observe that, for each natural number n, there is at most one graph of size n in the family $\mathcal{G}$. Below, we make a simple but useful observation regarding lack of information received by most of the nodes in graphs from $\mathcal{G}$ if at least two nodes transmit in a round.

▶ **Fact 11.** *If at least two nodes $u \neq v$ transmit in the ith round then, for every node w such that $C(w) \neq C(u) \wedge C(w) \neq C(v)$, w does not hear any message in the ith round.*

We now define the *canonical history*, for a given graph G from the family $\mathcal{G}$ and a given topology recognition algorithm $\mathcal{A}$ executed on G. The canonical history is a specific communication history, which will be common to many nodes of G in the early stages of the execution of a topology recognition algorithm. It has the property that a node with the canonical history receives a message in a round i iff every node in the whole graph receives this message in this round.

▶ **Definition 12** (Canonical history). *Fix a graph G from the family $\mathcal{G}$, with a given labeling scheme $\mathcal{L}$, and a topology recognition algorithm $\mathcal{A}$ executed on G. With respect to G, $\mathcal{L}$ and $\mathcal{A}$, we define the canonical history $\widehat{\mathcal{H}}_i$ at the end of the ith round as follows:*

- $\widehat{\mathcal{H}}_0 = \emptyset$.
- $\widehat{\mathcal{H}}_i = [\widehat{\mathcal{H}}_{i-1}, m]$, *if there is exactly one node in G that transmits the message m in the ith round.*
- $\widehat{\mathcal{H}}_i = [\widehat{\mathcal{H}}_{i-1}, \#]$, *if at least two nodes of G transmit in round i,*
- $\widehat{\mathcal{H}}_i = [\widehat{\mathcal{H}}_{i-1}, \varepsilon]$, *if no node of G transmits in round i.*

We then define the set of components sharing the canonical history.

▶ **Definition 13.** *Fix a graph G from the family $\mathcal{G}$, with a given labeling scheme $\mathcal{L}$, and a topology recognition algorithm $\mathcal{A}$ executed on G. We say that a node v has the canonical history $\widehat{\mathcal{H}}_i$ if $\mathcal{H}_i(v) = \widehat{\mathcal{H}}_i$ at the end of the ith round of the execution of $\mathcal{A}$ on G. A component C_j has the canonical history at the end of the ith round if every node from that component has the canonical history at the end of this round.*

Let $\widehat{\mathcal{C}}_i$ denote the set of components of G that have the canonical history $\widehat{\mathcal{H}}_i$ at the end of the ith round:

$$\widehat{\mathcal{C}}_i = \{\, C_j \mid \forall_{v \in C_j} \mathcal{H}_i(v) = \widehat{\mathcal{H}}_i \,\}.$$

High-level idea of the proof of the lower bound. A node with the canonical history does not have any information that distinguishes it from other nodes with the canonical history, apart from its label. In topology recognition, nodes with different degrees must make different decisions (as each node has to situate itself in the graph), and there are at least $\frac{1}{2}\sqrt{n}$ different degrees in each component of a graph $G_n \in \mathcal{G}$. The communication history of all nodes at the beginning is equal to the canonical history. We show that, in every round, nodes from at most two components can change their histories from the canonical to a non-canonical one. Using this fact we show that some nodes change their histories from the canonical one to non-canonical after at least $\frac{1}{2}\sqrt{n}$ rounds. We also show that a node makes such a change (from the canonical to non-canonical history) in round i iff at least one node from its component (call it a *trigger*) transmits a message in that round and that nodes from at most one other component transmit messages in that round. Using these properties we show that, for each label l, at most two nodes with the label l can be triggers. On the other hand, we need a trigger in each component, so $\frac{1}{4}\sqrt{n}$ different labels are needed which implies that the length of the labeling scheme is $\Omega(\log n)$.

▶ **Corollary 14.** *Any algorithm solving the topology recognition problem in all graphs from the family $\mathcal{G}$ requires a labeling scheme of length $\Omega(\log n)$ on G_n.*

4.2 The Lower Bound for Graphs with Arbitrary Degrees

In this section we show that, for all positive integers $\Delta < n$, there exists a graph $H_{\Delta,n}$ of maximum degree $\Theta(\Delta)$ and of size $\Theta(n)$, such that any algorithm solving the topology recognition problem in $H_{\Delta,n}$ requires a labeling scheme of length $\Omega(\log \Delta)$. In order to prove this lower bound we use the family $\mathcal{G}$ of graphs G_n of maximum degrees $2^{\Omega(\log n)}$, where n is the size of the graph, constructed in Section 4.1.

Choose arbitrary positive integers $\Delta < n$. We construct a $\Theta(n)$-node graph $H_{\Delta,n}$ from $\Theta(n/\Delta)$ isomorphic copies of a graph G_k from $\mathcal{G}$, for some $k = \Theta(\Delta)$. Let k be the smallest natural number such that $k \geq \Delta$ and $\sqrt{k}$ is a natural even number. Let G'_Δ denote the graph obtained by the following slight modification of the graph G_k: add a new *special* node s connected to all other nodes of G_k. Let $G'^{(i)}_\Delta$, for all $i = 1, 2, \ldots, \lceil \frac{n}{\Delta} \rceil$, denote the ith isomorphic copy of the graph G'_Δ. All these copies are pairwise disjoint.

The graph $H_{\Delta,n}$ contains the above $\lceil \frac{n}{\Delta} \rceil$ disjoint isomorphic copies $G'^{(i)}_\Delta$ of G'_Δ, for all $i = 1, 2, \ldots, \lceil \frac{n}{\Delta} \rceil$. Moreover, we ensure connectivity of $H_{\Delta,n}$ in the following way. Let s_i denote the special node of $G'^{(i)}_\Delta$. We connect every s_i with s_{i+1} for each $i = 1, 2, \ldots, \lceil \frac{n}{\Delta} \rceil - 1$ by an edge, as well as $s_{\lceil \frac{n}{\Delta} \rceil}$ with s_1. Thus the special nodes of all graphs $G'^{(i)}_\Delta$ are connected in a ring. As the graph G'_Δ is connected, the graph $H_{\Delta,n}$ is connected as well. Moreover, $H_{\Delta,n}$ has maximum degree $\Theta(\Delta)$ and size $\Theta(n)$.

▶ **Theorem 15.** *For all sufficiently large positive integers $\Delta < n$, there exists a graph of size $\Theta(n)$ and of maximum degree $\Theta(\Delta)$ such that any topology recognition algorithm for this graph requires a labeling scheme with more than $\Delta^{1/4}$ distinct labels.*

The following corollary of Theorem 15 is the main result of this section.

▶ **Corollary 16.** *For all positive integers $\Delta < n$, there exists a graph of maximum degree $\Theta(\Delta)$ and of size $\Theta(n)$, such that any algorithm solving the topology recognition problem on this graph requires a labeling scheme of length $\Omega(\log \Delta)$.*

The above corollary shows that algorithm TopRec solves the topology recognition problem using a labeling scheme of optimal length.

5 Fast Algorithm for Size Discovery Problem

We finally design an algorithm for size discovery that is much faster than algorithm GeneralSD and also solves the size discovery problem using a labeling scheme of optimal length $O(\log \log \Delta)$. We also prove that this algorithm is the fastest possible among size discovery algorithms using such an optimal labeling scheme.

In Section 5.1, we describe a size discovery algorithm with time complexity $O(\log^2 n)$ using a labeling scheme of optimal length $O(\log \log \Delta)$. In Section 5.2, we show that this time complexity is asymptotically optimal for labeling schemes of length $O(\log \log \Delta)$.

5.1 The algorithm

In this section, we design the algorithm FastSD, which solves the size discovery problem in time $O(\log^2 n)$ using a labeling scheme with asymptotically optimal length $O(\log \log \Delta)$. Actually, if the diameter of the graph is $\Omega(\log n)$, constant-length labels are sufficient.

In Section 5.1.1, we generalize the recent labeling schemes for broadcast to the multi-source broadcast problem. Then, in Section 5.1.2, we give the idea of our new size discovery algorithm, using the multi-source broadcast algorithm. Finally, in Section A.2, we give details of our algorithm.

5.1.1 Multi-source broadcast

For the purpose of faster size discovery, we need a generalization of the result from [7] regarding the multi-source broadcast problem. Let $G = (V, E)$ be a graph and let a non-empty set $S \subset V$ be the set of *sources*. Assuming that all nodes from S know the broadcast message M, the goal of *multi-source broadcast* is to deliver the message M to all nodes of G. Let the *distance* from a node $v \in V$ to $S \subset V$ be the minimum of distances from v to s in G, over all $s \in S$. Moreover, let the diameter D_S of G with respect to S be the maximum of distances from v to S, over all $v \in V$.

We construct the multi-source broadcast algorithm MBroadcast through a simple modification of the broadcast algorithm Executor from [7]. The labeling scheme for the (single-source) broadcast algorithm from [7] is based on the probabilistic broadcast algorithm from [3]. As one can see in Section 6.3 of [7] (Lemmas 9 and 10), the probabilistic analysis depends merely on the length of a shortest path from a node which knows the broadcast message initially to a considered node v. More precisely, the only differences with respect to Executor concern the definition of the sets FRONTIER_1 and DOM_1, i.e., the set of uninformed nodes that have an informed neighbor before round 1 and the set of informed nodes that is a minimal dominating set of FRONTIER_1. As there is exactly one informed node s at the beginning in the broadcast problem, we have $\mathrm{DOM}_1 = \{s\}$ and $\mathrm{FRONTIER}_1$ is equal to the set of neighbors of s, in the labeling scheme used by Executor. For the multi-source broadcast problem, we

- set $\mathrm{FRONTIER}_1 = \{v \mid (s, v) \in E \text{ and } v \notin S\}$,
- choose a subset of S that is a minimum dominating set of $\mathrm{FRONTIER}_1$ and let this chosen subset of S be DOM_1.

Then, the analysis from [7] works also for the multi-source broadcast problem and it implies a multi-source broadcast algorithm MBroadcast using a constant-length labeling scheme and working in time $O(D_S \log n + \log^2 n)$.

▶ **Corollary 17.** *Algorithm MBroadcast solves the multi-source broadcast problem using a labeling scheme of length $O(1)$, and works in time $O(D_S \log n + \log^2 n)$, where S is the set of sources.*

Given the above multi-source broadcast algorithm, we can describe our faster size discovery algorithm. For this purpose we introduce a few auxiliary notions.

Let T_{BFS} be a BFS tree of the graph, where the root of the tree is an arbitrary node s. Moreover, let V_i be the set of nodes at distance i from the root, called the layer i of T_{BFS}. Thus, in particular, $V_0 = \{s\}$.

5.1.2 The idea of the $O(\log^2 n)$-time size discovery algorithm

If the diameter D of the graph G is $O(\log n)$, we simply apply GeneralSD which gives the desired $O(\log^2 n)$ time bound. Otherwise, we conceptually split the graph G into subgraphs of small diameter and then distribute information about the value of n in these subgraphs separately. Still, we have to assure that collisions caused by edges connecting different subgraphs do not prevent successful execution of our subroutines in the considered subgraphs. For brevity, we introduce the notation $\lg n = \lceil \log(n+1) \rceil$, i.e., $\lg n$ is the length of the binary representation of the natural number n. We say that a layer V_i is *green* iff $\lfloor i/\lg n \rfloor$ is an even number; i.e., the layers $[j\lg n, (j+1)\lg n - 1]$ are green for each even number $j \geq 0$. The nodes from layers $[j\lg n, (j+1)\lg n - 1]$ for even j form the jth *stripe* X_j. Thus, we have the 0th, 2nd, 4th stripe and so on. Moreover, two consecutive stripes X_j and X_{j+2} are "separated" by nodes from the layers $V_{(j+1)\lg n}, V_{(j+1)\lg n+1}, \ldots, V_{(j+2)\lg n-1}$ which do not belong to any stripe. Nodes from each stripe are called *green*. Moreover, nodes from the last layer $V_{(j+1)\lg n-1}$ of the jth stripe for each even j are called *super-green*.

Our algorithm FastSD (an abbreviation of Fast Size Discovery) either executes GeneralSD (if $D = O(\log n)$) with appropriate labels or it consists of two stages:

Stage 1: In each stripe separately and in parallel, we execute an algorithm which ensures that, at the end of the stage, all super-green nodes from the stripe (i.e., nodes from the last layer $(j+1)\lg n + 1$ of the stripe j) learn the value of n at the end of the stage. We perform this task in two phases. For the first phase, we choose a minimal set U_j of nodes from the first layer of the stripe (called a minimal BFS-cover) such that each super-green node in the stripe j is reachable from this set U_j, i.e., there is a BFS-path from a node of U_j to this node. Then, for each element of $u \in U_j$ we choose a path of length $\lg n$ such that it is a shortest path from u to the last layer $(j+1)\lg n - 1$ of the stripe j. Moreover, we guarantee that those paths are conflict-free, i.e., there are no edges connecting nodes on different layers of different paths from the chosen family of paths. For each such path, the value of n is then encoded in 1-bit parts along the nodes of the path. In Phase 1 we "collect" these 1-bit parts in appropriate nodes from the BFS-cover U_j and, in Phase 2, we use the multi-source broadcast algorithm MBroadcast with U_j as the the set of sources and $D = \log n$ in order to broadcast the size n to super-green nodes (i.e., the nodes on the layer $(j+1)\lg n - 1$). As the consecutive stripes are "separated" by $\log n$ layers, we can execute Phase 1 separately in each stripe without the risk that interferences between nodes from different stripes cause any problem.

Stage 2: Execute the multi-source broadcast algorithm MBroadcast (see Corollary 17) with S equal to the set of all super-green nodes, i.e., S composed of the nodes from the layers $(j+1)\lg n - 1$, where j is even and $(j+1)\lg n - 1$ is not larger than the height of the BFS tree and $D_S = 2\lg n$. The broadcast message in the execution of MBroadcast is equal to the binary encoding of n, known to the nodes from S after Stage 1. As each node of the graph is at distance $\leq 2\lg n$ from some super-green node, all nodes learn the value of n in Stage 2.

5.2 Lower bound on time complexity of size discovery

We leverage a powerful technical result from [1], originally proved as a part of a proof of the $\Omega(\log^2 n)$ lower bound on time complexity of broadcast in radio networks, to show that time complexity of FastSD is asymptotically optimal.

Let $G = (U \cup V, E)$ be a bipartite graph, where U, V are the parts of the bipartition of G, i.e., $U \cap V = \emptyset$, and there are no edges between nodes inside U nor inside V, such that $|U| = |V| = n$ and there is no isolated vertex in $U \cup V$. A *bipartite broadcast schedule* for G is a sequence $U_1, \ldots, U_k$ of subsets of U such that, if one executes a k-round radio network algorithm in G with the set of transmitters in the ith round equal to U_i for each $i \in [k]$, then every node from V receives (at least) one message during this execution of the algorithm. The number of subsets k is called the *size* of the bipartite broadcast schedule for G.

▶ **Theorem 18** ([1]). *There exists a constant $c > 0$ such that, for each sufficiently large natural number n, there exists a bipartite graph G with sides of size n such that the size of each bipartite broadcast schedule for G is larger than $c \log^2 n$.*

Equipped with Theorem 18, we are ready to prove the lower bound $\Omega(\log^2 n)$ on the time complexity of the size discovery problem with short labels.

▶ **Theorem 19.** *The time complexity of any algorithm solving the size discovery problem on all graphs with size at most n, using labeling schemes of length smaller than $\frac{1}{4}\log n$, is $\Omega(\log^2 n)$.*

Since $O(\log\log\Delta)$ is $o(\log n)$ for any graph, and in view of the lower bound $\Omega(\log\log\Delta)$ from [19] on the length of labeling schemes permitting size discovery, Theorems 25 and 19 give the following corollary.

▶ **Corollary 20.** *Algorithm FastSD solves the size discovery problem on all graphs using a labeling scheme of asymptotically optimal length $O(\log\log\Delta)$, and it works in time $O(\log^2 n)$, which is aymptotically optimal for size discovery algorithms using asymptotically optimal labeling schemes.*

6 Conclusion

We constructed labeling schemes of asymptotically optimal length for size discovery and topology recognition in arbitrary radio networks without collision detection. We also designed algorithms to solve these problems using these optimal schemes.

In the case of size discovery, we showed that our algorithm using the labeling scheme of optimal length $O(\log\log\Delta)$ has also asymptotically optimal time among size discovery algorithms using optimal schemes for this problem. The main open problem left by our research is the following: What is the time of the fastest topology recognition algorithm using a labeling scheme of optimal length $O(\log\Delta)$?

References

1 Noga Alon, Amotz Bar-Noy, Nathan Linial, and David Peleg. A lower bound for radio broadcast. *J. Comput. Syst. Sci.*, 43(2):290–298, 1991.

2 Stephen Alstrup, Haim Kaplan, Mikkel Thorup, and Uri Zwick. Adjacency labeling schemes and induced-universal graphs. *SIAM J. Discret. Math.*, 33(1):116–137, 2019.

3 Reuven Bar-Yehuda, Oded Goldreich, and Alon Itai. On the time-complexity of broadcast in multi-hop radio networks: An exponential gap between determinism and randomization. *J. Comput. Syst. Sci.*, 45(1):104–126, 1992.

4 Gewu Bu, Zvi Lotker, Maria Potop-Butucaru, and Mikael Rabie. Lower and upper bounds for deterministic convergecast with labeling schemes. Manuscript, HAL Id: hal-02650472, 2020.

5 Marek Chrobak, Leszek Gasieniec, and Wojciech Rytter. Fast broadcasting and gossiping in radio networks. *J. Algorithms*, 43(2):177–189, 2002. doi:10.1016/S0196-6774(02)00004-4.

6 Artur Czumaj and Peter Davies. Exploiting spontaneous transmissions for broadcasting and leader election in radio networks. *J. ACM*, 68(2):13:1–13:22, 2021. doi:10.1145/3446383.

7 Faith Ellen and Seth Gilbert. Constant-length labelling schemes for faster deterministic radio broadcast. In Christian Scheideler and Michael Spear, editors, *SPAA '20: 32nd ACM Symposium on Parallelism in Algorithms and Architectures*, pages 213–222. ACM, 2020.

8 Faith Ellen, Barun Gorain, Avery Miller, and Andrzej Pelc. Constant-length labeling schemes for deterministic radio broadcast. In C. Scheideler and P. Berenbrink, editors, *31st Symposium on Parallelism in Algorithms and Architectures, SPAA 2019*, pages 171–178. ACM, 2019.

9 Yuval Emek, Pierre Fraigniaud, Amos Korman, and Adi Rosén. Online computation with advice. *Theor. Comput. Sci.*, 412(24):2642–2656, 2011. doi:10.1016/j.tcs.2010.08.007.

10 Pierre Fraigniaud, David Ilcinkas, and Andrzej Pelc. Communication algorithms with advice. *J. Comput. Syst. Sci.*, 76(3-4):222–232, 2010. doi:10.1016/j.jcss.2009.07.002.

11 Pierre Fraigniaud, Amos Korman, and Emmanuelle Lebhar. Local MST computation with short advice. *Theory Comput. Syst.*, 47(4):920–933, 2010. doi:10.1007/s00224-010-9280-9.

12 Emanuele G. Fusco, Andrzej Pelc, and Rossella Petreschi. Topology recognition with advice. *Inf. Comput.*, 247:254–265, 2016. doi:10.1016/j.ic.2016.01.005.

13 Adam Ganczorz, Tomasz Jurdzinski, Mateusz Lewko, and Andrzej Pelc. Deterministic size discovery and topology recognition in radio networks with short labels. *CoRR*, abs/2105.10595, 2021. arXiv:2105.10595.

14 Leszek Gasieniec, Aris Pagourtzis, Igor Potapov, and Tomasz Radzik. Deterministic communication in radio networks with large labels. *Algorithmica*, 47(1):97–117, 2007.

15 Leszek Gasieniec, David Peleg, and Qin Xin. Faster communication in known topology radio networks. *Distributed Comput.*, 19(4):289–300, 2007. doi:10.1007/s00446-006-0011-z.

16 Mohsen Ghaffari, Bernhard Haeupler, and Majid Khabbazian. Randomized broadcast in radio networks with collision detection. In Panagiota Fatourou and Gadi Taubenfeld, editors, *PODC*, pages 325–334. ACM, 2013. doi:10.1145/2484239.2484248.

17 Christian Glacet, Avery Miller, and Andrzej Pelc. Time vs. information tradeoffs for leader election in anonymous trees. *ACM Trans. Algorithms*, 13(3):31:1–31:41, 2017.

18 Barun Gorain and Andrzej Pelc. Short labeling schemes for topology recognition in wireless tree networks. In S. Das and S. Tixeuil, editors, *24th International Colloquium, SIROCCO 2017*, volume 10641 of *Lecture Notes in Computer Science*, pages 37–52. Springer, 2017.

19 Barun Gorain and Andrzej Pelc. Finding the size of a radio network with short labels. In P. Bellavista and V. K. Garg, editors, *Proc. of the 19th International Conference on Distributed Computing and Networking, ICDCN 2018*, pages 10:1–10:10. ACM, 2018.
20 David Ilcinkas, Dariusz R. Kowalski, and Andrzej Pelc. Fast radio broadcasting with advice. *Theor. Comput. Sci.*, 411(14-15):1544–1557, 2010. doi:10.1016/j.tcs.2010.01.004.
21 Dariusz R. Kowalski and Andrzej Pelc. Optimal deterministic broadcasting in known topology radio networks. *Distributed Computing*, 19(3):185–195, 2007. doi:10.1007/s00446-006-0007-8.
22 Dariusz R. Kowalski and Andrzej Pelc. Leader election in ad hoc radio networks: A keen ear helps. *J. Comput. Syst. Sci.*, 79(7):1164–1180, 2013. doi:10.1016/j.jcss.2013.04.003.
23 Colin Krisko and Avery Miller. Labeling schemes for deterministic radio multi-broadcast. *CoRR*, abs/2104.08644, 2021. arXiv:2104.08644.
24 Pedro Montealegre, Sebastian Perez-Salazar, Ivan Rapaport, and Ioan Todinca. Graph reconstruction in the congested clique. *J. Comput. Syst. Sci.*, 113:1–17, 2020.

A Appendix

A.1 Details of the Broadcast-gathering primitive

Assignment of broadcast-labels and gather-labels

Assignment of the values b_v and construction of a BFS-tree. For each layer i, we assign the values b_v as follows. First, we take any maximal subset X_0 of V_i such that the sets of neighbors at layer $i+1$ of elements of this set are pairwise disjoint. All nodes from $\mathcal{N}_{i+1}(v)$ will be called *children* of v, for each $v \in X_0$. Then, we construct sets $X_1, X_2, \ldots, X_{\Delta-1}$ as follows. Assume that the sets $X_0, \ldots, X_{j-1}$ are already constructed for $j > 0$. We define X_j as a maximal subset of nodes v at layer i such that

(a) $v \notin X_0 \cup \cdots \cup X_{j-1}$,

(b) for each pair of nodes $v \neq u$ such that $v, u \in X_j$, we have

$$\mathcal{N}_{i+1}(v) \cap \mathcal{N}_{i+1}(u) \setminus \left(\bigcup_{k=0}^{j-1} \bigcup_{x \in X_k} \mathcal{N}_{i+1}(x) \right) = \emptyset.$$

If a node $v \in X_j$ has a neighbor u in V_{i+1}, such that no node from X_k is in $\mathcal{N}_i(u)$ for all $k < j$, we say that v is the *parent* of u and u is a *child* of v. Observe that the set of edges determined by this parent-child relationship will form a BFS tree of the graph, denoted T_{BFS}. As we show in the full version of the paper, all nodes at layer i belong to $X_0 \cup \cdots \cup X_\Delta$. We set $b_v \leftarrow k$ for each node $v \in X_k$, where the sets $X_0, \ldots, X_\Delta$ are as above.

For the acknowledged broadcast, we choose some arbitrary leaf node with the highest value of *layer*(v) and a path P of length *layer*(v), such that P starts at r and ends at v in the tree T_{BFS} obtained through parent-child relationships described above. The value of a_u is set to 1 for each node u on the path P while the value a_w is set to 0 for each node w outside of P.

Assignment of the values g_v. As mentioned before, the bit g_v of each node v will be used in our gathering algorithm. The value g_r of the root node r is equal to 0. Then we assign values g_v layer by layer, assigning the values to nodes at layer i in phase i. In the ith phase, we order nodes at layer $i-1$ according to the ordering determined by the values of b_v, from the smallest to the largest and, among nodes with the same value of b_v, according to the ordering $\prec$. Let v_j be the jth node at layer $i-1$ in this order. We give the value g_u to each *child* u of v_j in the BFS-tree $T_{\rm BFS}$, as follows. For a node v at layer $i-1$, we order its children at layer i (i.e., its neighbors at layer i which have not been assigned gather-labels yet) according to $\prec$, and assign to each of them the smallest non-negative integer not assigned earlier to any neighbor of v at layer i.

Below, we state some properties relevant for the gathering algorithm.

▶ **Lemma 21.** *The integers g_v have the following properties:*
(a) $g_v \in [0, \Delta - 1]$,
(b) $g_v \neq g_u$ *for each* $u \neq v$ *such that* $parent(u) = parent(v)$,
(c) *if* $parent(v) \neq parent(u)$ *and* $g_u = g_v$ *for some* $u \neq v$ *at the same layer, then* u *and* $parent(v)$ *are not connected by an edge.*

Since the label $\mathcal{L}(v)$ of any node is a concatenation of three bits and three representations of integers at most Δ, our labeling scheme has length $O(\log \Delta)$.

The properties of the labeling scheme facilitate the construction of a broadcast algorithm, acknowledged broadcast algorithm and a gathering algorithm, called BroadcastBFS, AckBrBFS and GatherBFS respectively (see the full version of the paper), with the following properties.

▶ **Lemma 22.** *The algorithms BroadcastBFS, AckBrBFS and GatherBFS solve the broadcast problem, the acknowledged broadcast problem and the gathering problem, respectively, using a labeling scheme of length $O(\log \Delta)$, and work in time $O(D\Delta)$.*

A.2 Details of algorithm FastSD

We say that a path $P = (x_1, \ldots, x_k)$ in a graph G with a fixed source node s is a *BFS-path* if, for each consecutive nodes x_i and x_{i+1} on P, the layer of x_{i+1} is by one larger than the layer of x_i. In other words, each edge of the path increases the distance from the source node s by one. Thus, in particular, we do not use edges connecting nodes from the same layer. We say that v is *BFS-reachable* from u if there exists a BFS-path from u to v.

In order to implement Stage 1 of FastSD, we first build a *BFS-cover* of stripe X_j by the nodes from the first layer $V_{j \log n}$ of the stripe, for each even j (see Figure 2).

▶ **Definition 23** (BFS-cover)**.** *For an even natural number j, the set of nodes $U_j \subseteq V_{j \log n}$ is a BFS-cover of the stripe $X_j = \bigcup_{i=j lg\ n}^{(j+1)lg\ n-1} V_i$ if, for each node $v \in V_{(j+1)lg\ n-1}$, there exists a BFS-path $u_0, u_1, \ldots, u_{lg\ n-1} = v$ such that $u_i \in V_{j lg\ n+i}$ for each $i \in [0, lg\ n-1]$, and $u_0 \in U_j$. In other words, the path $u_0, \ldots, u_{lg\ n-1}$ starts at some node from the given BFS-cover U_j and each edge of the path goes to the layer with larger index.*

We say that a BFS-cover U_j of the stripe X_j is a *minimal BFS-cover* of the stripe X_j if no proper subset U' of U_j is a BFS-cover of X_j. A set of BFS-paths $P_1, \ldots, P_k$ is *conflict-free* if, for each $i \neq j$, there is no edge (x_i, x_j) in the graph such that $x_i \in P_i$, $x_j \in P_j$, x_i and x_j belong to different layers.

▶ **Lemma 24.** *Let $U_j = \{u_1, \ldots, u_k\}$ be a minimal BFS-cover of the stripe X_j. Then, there exists a set of* conflict-free *BFS-paths $\{P_1, \ldots, P_k\}$ such that, for each $i \in [k]$:*
- *P_i starts at u_i,*
- *the final node of P_i is a super-green node of the stripe X_j,*

Let $M = M[1]M[2] \cdots M[\lg n]$ be the binary representation of n. In order to facilitate implementation of Stages 1 and 2 of FastSD, we construct labels of nodes in a given (green) stripe X in the following way (see Figure 2):

1. A minimal BFS-cover $U = \{u_1, \ldots, u_k\}$ of the stripe X is chosen, along with the set of conflict-free paths $\mathcal{P} = \{P_1, \ldots, P_k\}$ of length lg n, such that P_i starts at u_i and ends at a node in the last layer of X (i.e., at a supergreen node).
2. The set $X_{\text{BFS}} \subseteq X$ is determined such that $x \in X_{\text{BFS}}$ if x is BFS-reachable from some element of the BFS-cover U. (Observe that, although each node from the last layer of a stripe is BFS-reachable from U by the definition of a BFS-cover, it might be the case that nodes on smaller layers of the stripe are not reachable from U.)
3. The label of each node v is a concatenation of the following strings:
   a. a 4-bit string composed of flags REACH$_v$, SUPER-GREEN$_v$, COVER$_v$, PATHS$_v$ indicating whether v is green and it belongs to X_{BFS} of its stripe X, whether v is super-green, whether v belongs to the chosen BFS-cover U of its stripe, and whether v belongs to one of the chosen cover-free paths $\mathcal{P} = \{P_1, \ldots, P_k\}$ of its stripe, respectively;
   b. M_v: a one-bit string defined as follows. If v is the ith node of a path P_l from the above set of conflict-free paths, then M_v is equal to $M[i]$; otherwise, the value of M_v is 0;
   c. B_v: the constant-length label assigned to v by the labeling scheme for the multi-source broadcast algorithm MBroadcast, with the graph $G' = (V', E')$, where $V' = X_{\text{BFS}}$, E' is the set of edges of G connecting nodes from X_{BFS}, and the set of sources S is equal to U.
   d. S2$_v$: the constant-length label assigned to v by the labeling scheme for the multi-source broadcast algorithm MBroadcast, with the communication graph G, where the set of sources S is equal to the set of all super-green nodes (i.e., such nodes v that SUPER-GREEN$_v = 1$).

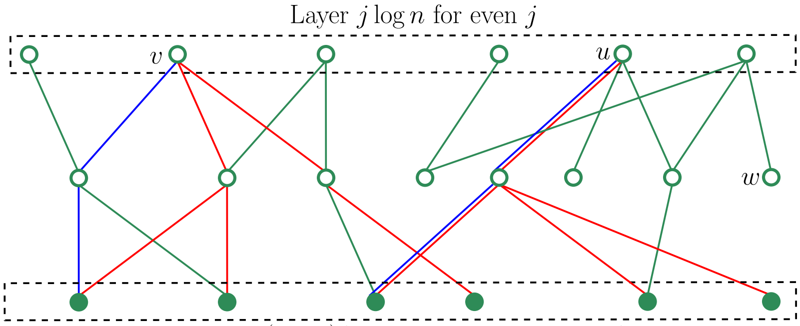

Figure 2 An example illustrating the notions and terms from the construction of FastSD. The edges denote paths of length $\frac{1}{2}$lg n. The set $\{u, v\}$ is a minimal BFS-cover – the red edges show that each node from the layer $(j+1)$lg $n - 1$ is reachable from $\{u, v\}$ by a BFS-path. The blue edges form a set of conflict-free BFS-paths described in Lemma 24. The node w is an example of a node which is not active in Phase 2 of Stage 1, since $w \notin X_{\text{BFS}}$. However, w is at distance ≤ 2lg n from some super-green node from the layer $(j-1)$lg $n - 1$ (stripe $j-2$) and therefore w receives a message with the size of the graph in Stage 2.

Using the above described labels, **Stage 1** for a stripe X is implemented as follows:

Phase 1: collecting the value of n at all nodes from the BFS-cover U.

- Round 1: each super-green node v from $\mathcal{P}$ (i.e., each v such that SUPER-GREEN$_v = 1$ and PATHS$_v = 1$) sends M_v.
- Rounds $2, 3, \ldots,$ lg n: each node v with PATHS$_v = 1$ which received a message M in the previous round sends the concatenation of its bit M_v and the received message M.

Phase 2: broadcast of the value of n inside X_{BFS}.
Using the labels B_v, all nodes with $\textsc{reach}_v = 1$ execute the multi-source broadcast algorithm MBroadcast, where the set of sources S is equal to U and the graph is $G' = (V', E')$, $V' = X_{\mathrm{BFS}}$ (determined by the flags $\textsc{cover}_v$) and E' is the set of edges of G connecting nodes from X_{BFS}.

Stage 2 is an execution of the multi-source broadcast algorithm MBroadcast on the whole graph G using the labels $\mathrm{S2}_v$ and thus with the source set S consisting of all super-green nodes (i.e., the nodes v with $\textsc{super-green}_v = 1$).

▶ **Theorem 25.** *Algorithm FastSD solves the size discovery problem in time $O(\log^2 n)$ using a labeling scheme of length $O(\log\log\Delta)$.*